OUTDATED SCRIPTS: Ignore the python/scripts in this guide, but it is an annotation of the steps in Deep Lab Cut.

June 21st release of DLC User Guide [HERE](https://www.nature.com/articles/s41596-019-0176-0.epdf?author_access_token=YdgZNNupsH23JNKsogan2tRgN0jAjWel9jnR3ZoTv0PafSJfogKvzMN9Yibs0_Cn5RrZDjtijUWL61FeNFC2vYig0gH-hrXbchDCvAxDsQhlBRStoRflJdbJePPUfES7jig_npStoBnzkKR817z8Tw%3D%3D)

**To use DeepLabCut:**

**\*You will have to complete steps 1-3 every time that you wish to use DeepLabCut\***

**As of June 2019: Please see HPC notes to open and run DeepLabCut**

1. Open terminal and activate your virtual environment from the installation
   1. Ex. $ source activate dlc-macOS\_CPU
2. Once in the virtual environment, enter ipython or pythonw by typing either 'ipython' or 'pythonw'
   1. *Why:* ipython/pythonw are more advanced terminal interfaces that allow you to do more with your python-based software
   2. *Check:* once you enter into the new terminal, you should have new command prompts
      1. For ipython: 'In [1]:'
      2. For pythonw: '>>>'
      3. To exit either, use the command 'ctrl + z'
      4. \*For MacOS users, it is recommended that you use pythonw for DLC
3. Import DeepLabCut in ipython
   1. *How:*Type >>> import DeepLabCut
   2. *Why:* this will enable the DLC commands and interface that will allow you to analyze video. Without importing DLC, you will not be able to run any commands.
   3. *Check:* If DLC is imported correctly, you should be prompted with a new ipython/pythonw command
   4. *Error:*Module not found but DLC is installed? Make sure that you entered into your dlc conda environment before opening ipython

**For all subsequent steps, Deeplabcut MUST be imported into your terminal in order for the commands to work. You will have to do this each time that you start a new terminal**

1. **Create a New Project in DLC:**
   1. *Definition:*Projects are the training and analysis space for your videos. You will load, label, train and analyze videos within this space. One project should be used to train and analyze similar videos (I.e. one project can be dedicated to videos of the same organism with approximately the same frame shape. If you were studying a different organism, or using a new perspective, you would want to create a new project)
   2. *How:*1st define your variables:
      1. >>> task='name\_of\_your\_project'
      2. >>> user='your-name"
      3. >>>video='pathname\_of\_your\_video'
      4. >>>working\_directory='pathname\_to\_desired\_folder'

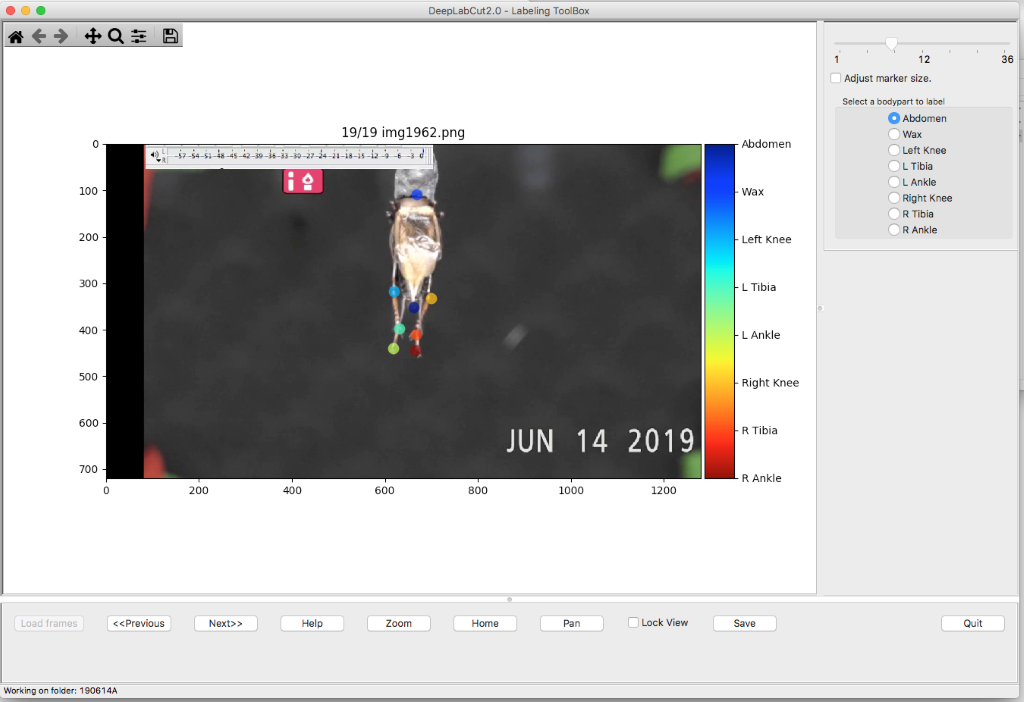
2nd, type: >>>deeplabcut.create\_new\_project (task, user, [video], working\_director)

* 1. Path to your videos should be the full path to the video you wish to analyze. You can include as many (or as few) videos as you wish, just separate by commas
  2. The working directory should be the folder that you wish to place the project in – it will be the site of all further steps and analysis and the destination for all files created while performing DLC analysis
  3. *Check:*Once you have created your new project, you should see a new folder labeled "Project Title, Name of Experimenter, Date you created project" that contains the following subdirectories:
     1. Config.yaml
     2. Dlc-models
     3. Labeled-data
     4. Training-datasets
     5. Videos
  4. For more information and descriptions of the subdirectories read Part A of [this document](https://github.com/AlexEMG/DeepLabCut/blob/master/docs/functionDetails.md#A-create-a-new-project)

1. **Save the Configuration File Path (will make your life much easier)**
   1. *Definition:*In the code output from creating a project, you will be prompted with a configuration file path that will follow the format of the working\_directory\_file\_name/project\_file\_name/config.yaml
   2. *How:*>>>config\_path = 'working\_directory\_folder\_name/project\_file\_name/config.yaml'
      1. You can choose to store it as any word, but all deeplabcut tutorials/information store it as 'config\_path'
      2. If you're not sure what your config path should be, open the config file, find the project path, and add '/config.yaml'.
   3. *Why:*This path is required in every subsequent command. Saving the path in this form will greatly simplify all future commands.
      1. note: you will have to re-store the path as 'config\_path' each time you start in a new terminal
   4. *Error:*Config file not found? Make sure that there are no misspellings in your pathname and that you ended it with '/config.yaml'
2. **Add New Videos to your Project (optional step)**
   1. *Definition:* This function exists if you would like to add more videos to a project after you have created it
   2. *How*:  >>>deeplabcut.add\_new\_videos(config\_path,[`full path of video 4', `full path of video 5'])
3. **Configure the Project Parameters**
   1. *Definition:*The config.yaml file in your project folder is a configuration file that allows you to define your own label points (I.e. abdomen, head, left knee etc.) for your project, as well as  parameters of the frames that you wish to edit. Box 1 defines each part of the configuration file.
   2. *How:*Open the config.yaml file in any text editor and edit to include your desired information.  Key: do this step before you begin extracting frames for labeling and SAVE the edited configuration file
   3. WHAT DO WE WANT TO HAVE AS OUR LABELS AND WHY?
   4. BOX I: Glossary of parameters in the project configuration file (config.yaml) 
      The config.yaml file sets the various parameters for generation of the training set file and evaluation of results. 
      The meaning of these parameters is defined here, as well as referenced in the relevant step. 
      Parameters set during the project creation: 
      task: Name of the project (e.g. mouse-reaching). (do not edit) 
      scorer: Name of the experimenter (do not edit) 
      date: Date Of creation Of the project, (do not edit) 
      project_path: Full path Of the project; edit this if you need to move the project to a cluster/server/another computer or a different directory on your computer 
      video_sets: A dictionary with the keys as the full path of the video file and the values, crop as the cropping parameters used during frame extraction 
      (use the function add_new_videos to add more videos to the project; if necessary the paths can be edited manually, and the crop values are designed to be edited manually). 
      Important parameters to edit after project creation: 
      bodyparts: List containing names of the points to be tracked. The default is set to hand, Fingerl, Finger2, Joystick. Do not change after labeling frames (and saving 
      You can add additional labels later, if needed. 
      numframes2pick: This is an integer that specifies the number of frames to be extracted from a video or a segment of video. The default is set to 20. 
      colormap: It specifies the colormap used for plotting the labels in images Or videos in many steps, 
      Matplotlib colormaps are possible (https://matplotlib.org/examples/color/colormaps_reference.html). 
      dotsize: Specifies the marker size When plotting the labels in images or videos The default is set to 12. 
      alphavalue: Specifies the transparency of the plotted labels. The default is set to 0.5. 
      iteration: This keeps the count of the number of iterations used to create the training dataset The first iteration starts with O and thus the default value is set to O. 
      Do not change this manually, 
      If you are extracting frames from long videos: 
      start: Start point of interval to sample frames from when extracting frames. Value in relative terms of video length, i.e. is the full video. The default is set to 0. 
      stop: Same as start, but the end of the interval. Default is 1. 
      Related to the Neural Network Training: 
      TrainingFraction: This is a two digit floating-point number in the range 10-1] to split the dataset into training and testing dataset. The default is set to 0.95. 
      resnet: This specifies which pre-trained model to use. The default is set to 50 (user can choose 50 or 101, see also Mathis et al, 2018). 
      used during video analysis: 
      batch_size: This specifies how many frames to process at once during inference (For tuning Of this parameter see Mathis & Warren 201B). 
      snapshotindex: This specifies which checkpoint to use to evaluate the network. The default set to -1. use "all' to evaluate all the checkpoints. 
      Snapshots refer to the stored TensorFlow configuration. which holds the weights of the feature detectors. 
      p-cutoff: This specifies the threshold Of the likelihood and helps distinguishing likely body parts from uncertain ones. The default is Set to 0.1. 
      cropping: Specifies if the analysis video needs to be cropped. The default is set to False. 
      ,y2: These are the cropping parameters used for cropping novel video(s). The default is set to the frame size Of the video 
      used during refinement steps: 
      move2corner•. In some (rare) cases the predictions from DeepLabCut will be Outside Of the image (due to the location refinement shifts). 
      This binary parameter makes sure that those points are mapped to a user defined point within the image so that the label can be manually moved to the correct 
      location. The default is set to True. 
      corner2move2: This is the target location, if move2comer is True. The default is set to (50.50). 

1. **Extract Frames from the videos**
   1. *Definition:*This command selects still frames from the videos that will be used to label the features on the video (like legs, abdomen, head etc.). Frames can be selected in a uniform distribution, via a 'kmeans' algorithm that selects frames with a lot of movement, or manually.
   2. *How:*>>>deeplabcut.extract\_frames(config\_path, 'automatic', 'uniform/kmeans/manual')
   3. For more information and descriptions of the different methods to select frames, read Part C of [this document](https://github.com/AlexEMG/DeepLabCut/blob/master/docs/functionDetails.md#C-data-collection)

1. **Label Frames in the videos**
   1. *Definition:*This command opens up a Graphical User Interface that allows you to load and label points on the extracted frames
   2. *How:*>>>deeplabcut.label\_frames(config\_path)
      1. Click on 'Load Frames' and select the images from the video folder
      2. Control+Click on where you want to place your marker
      3. If you want to move your marker once placed, left click and drag the marker to the desired location.
      4. Select 'Save' once all images are labeled
      5. Select 'Quit' once all images are labeled
      6. If you have more frames to label, select "yes" to labeling another data set. If you are done labeling data sets, select "no".
   3. *Why:*These labels will be used to train the AI in order for it to track the body parts of the cricket (or whatever organism). The labels should be as accurate and consistent on the body parts as possible. If a body part is not visible, do not mark it.
   4. For more information on how to label frames read Part D of [this document](https://github.com/AlexEMG/DeepLabCut/blob/master/docs/functionDetails.md#D-label-frames)



1. **Check Labels on the frames**
   1. *How:*>>>deeplabcut.check\_labels(config\_path)
   2. *What/Why:*This creates a library of images of the labeled frames. If you are not happy with your label placement, repeat step 9 and move the markers that are not correctly placed by clicking and dragging the points.
2. **Setup your Terminal interface to train the project**
   1. *How:*Exit iPython (Ctrl+Z) and re-enter without GUIs activated
      1. $ Export DLClight="True"
      2. $iPython
      3. In [1] import deeplabcut
   2. *Why:*The training commands cannot work with GUIs, so you have to do the training steps in a more basic version of DLC
   3. *Check:* If the correct version of DLC is loaded, you should receive the message "DLC loaded in light mode; you cannot use the (re)labeling GUI!"
3. **Create the Training Dataset**
   1. *How:* >>>deeplabcut.create\_training\_dataset(config\_path, num\_shuffles=1)
   2. *What/Why:*This command allows the program to compile the labeled images into a training set and a test set which the computer will use to train the system and then determine if it correctly placed the markers on the cricket's legs.
   3. *Check:*This command should create two folders in the dlc-models folder of your project; a 'test' and a 'train' folder.
   4. For more information on the parameters of the training dataset, read Part F of [this document](https://github.com/AlexEMG/DeepLabCut/blob/master/docs/functionDetails.md#F-create-training-dataset)
4. **Train the Network:**
   1. *How:*>>>deeplabcut.train\_network(config\_path, maxiters=xxxxxxx)
   2. *What/Why:*This command uses the labeled frames and videos and its AI to learn how to correctly label the parts of the crickets. It does this by repeatedly running through iterations of the training until the error in its labeling is minimized. This process will take a considerable amount of time. Per the DLC website, you should only need ~200,000 iterations for good results, but the program will not stop until 1,000,000 iterations. This is controlled by the number put in the 'maxiters=xx'. \*\* Update this once we determine the optimal number of iterations\*\*
   3. *Check:* You should see a "learning\_stats.csv" and a log folder within the train folder of dlc-models of your project. DO NOT OPEN THESE FOLDERS until the training is complete
   4. **Check the status of the training network**
      1. *How:*In the regular terminal, cd into the train subdirectory and use the tail command to get learning\_stats updates
         * Ex. $ cd /users/jeschole/190617-Julie-2019-06-17/dlc-models/iteration-0/190617Jun17-trainset95shuffle1/train
         * $ tail -f learning\_stats.csv
      2. *Why:*This allows you to see the number of iterations that the training is on without opening (and therefore closing out) the training files
      3. *Check:*You should see a live running list of numbers. The numbers on the far left are the number of iterations that have been completed by the software.
5. **Evaluate the results of the trained network:**
   1. *Why:*Check the stats from your training to see how well the training went (I.e. the degree of error) and creates a directory in your project folder labeled 'evaluation-results' with the original labeled frames with both your labels and the computer's labels to see how accurately the computer labeled the sites. If the computer is wildly off on any of the frames, you can remove those labels later on in the training.
   2. *How: >>>deeplabcut.evaluate\_network(config\_path, plotting=True)*
   3. *Check:*You should see a new subdirectory 'evaluation-results' in your project folder, and this should contain resnet files and a folder containing the computer-labeled images. Look at these images to determine how well the computer did labeling.
6. **Analyze the trained network:**
   1. *Why:*This command allows you to analyze a full video and extract data from it. It can be the video that you used to train the network with or a completely new video. From this command, you can do multiple types of analyses/results visualization.
   2. *How:*>>>deeplabcut.analyze\_videos(config\_path, ['full\_path\_of\_video\_or\_video\_folder'], shuffle=1, save\_as\_csv=True, videotype='.avi')
   3. *Check:*If you labeled csv=true, this step should provide a file with the coordinates for each of the labeled parts over time that can be imported into outside analytical software (R, Matlab etc.)
   4. *Note:* I have found it necessary to import videos from outside of the project folder (I.e. don't use videos from the /project-folder/videos subdirectory) - I think this is because my imported videos in that folder have 0GB of information – the fix has been to copy outside videos into that folder instead.
7. **Get data output from DLC:**
   1. DLC has ways to look at/semi-evaluate the data using DLC commands
      1. >>>deeplabcut.create\_labeled\_video(config\_path, [video\_path]) creates a video file that has labels that track all labeled parts of the cricket
      2. >>>deeplabcut.filterpredictions(config\_path, [video\_path], videotype='.avi') unclear what this does but supposedly very helpful
      3. >>>deeplabcut.plot\_trajectories(config\_path, [video\_path]) creates a subdirectory called plot\_poses in the video folder that contains graphs/plots to start to visualize/interpret the data.