Short description

Has two functionalities:

- Sample images in balanced classes for each individual according to BOVIDS' prediction. It can be chosen whether images are sampled uniformly at random or if the script searches explicitly for "hard" examples, thus images in which the prediction score is small or single frame and multiple frame predict different classes. We suggest mining about at least 50% of images randomly, for as many videos as possible. To this end, run the script twice. If too few hard examples are present, the script will sample some images randomly to balance classes.
- Can furthermore extract those images from the FINAL_STORAGE_CUTOUT and save them such that they can be manually approved.
- Attention: The script will create a .csv file which is used to store the prediction by the single frame network, multiple frame network and, later on, by the human (evaluation of those images). The .csv file is created per individual and the filename corresponds to the individual code. If the script is run multiple times, you need to change the destination (OUTPUT_BASE) as otherwise those .csv files will be overwritten.

Requirements

- packages: numpy, pandas
- predicted nights by BOVIDS (folder /raw_csv/ in the total classification task in FINAL_STORAGE_PREDICTION_FILES/species/zoo/individualnumber/total/raw_csv/.

Step 1 – open spyder:

- Terminal / shell:
    - conda activate bovids
    - spyder

Step 2 – adjust parameters:

- KI_AUSWERTUNG = Path to the prediction by BOVIDS. Corresponds to FINAL_STORAGE_PREDICTION_FILES from global_configuration.py during prediction. [string]
- KI_CUTOUT = Path to the "cut out" folder of BOVIDS. Corresponds to FINAL_STORAGE_CUTOUT from global_configuration.py during prediction. [string]

- OUTPUT_BASE = Destination (folder). Will create a subfolder per individual automatically. [string]
- RANDOM_IMAGES = True, if random images should be sampled in balanced classes. False, if explicitely hard examples and conflicts should be chosen. [boolean]
- IMAGES_PER_INDIVIDUAL_AND_CLASS = Number of images per individual and class that are sampled. [integer]
- INDIVIDUALS = List of individual codes from which the images should be sampled. [list]

Further variables (need normally no adjustment):

- MAX_CONFIDENCY_RANDOM_IMAGES = Bound on the prediction score to count as a "hard" example. [float]
- MAXIMUM_NUMBER_CONFLICT_PER_INDIVIDUAL_AND_CLASS = Maximum number of images that are conflicts between both action classifiers. The other "hard" examples will be those with a low score. [integer]

Step 3 – run the script:

- ***get_samples():*** The .csv files are created (image paths are sampled).
- ***move_images():*** Given the .csv files from the last step, the necessary images are copied into the destination folder.
- Both steps are conducted sequentially: ***get_samples_and_copy_images()***