Week 5 Document Submission

Jacob Dallaire

September 21, 2024

1. Paper

Patton, P. T., Cheeseman, T., Abe, K., Yamaguchi, T., Reade, W., Southerland, K., Howard, A., Oleson, E. M., Allen, J. B., Ashe, E., Athayde, A., Baird, R. W., Basran, C., Cabrera, E., Calambokidis, J., Cardoso, J., Carroll, E. L., Cesario, A., Cheney, B. J., … Bejder, L. (2023). A deep learning approach to photo–identification demonstrates high performance on two dozen cetacean species. *Methods in Ecology and Evolution*, 14(10), 2611–2625. https://doi.org/10.1111/2041-210X.14167

SUMMARY

2. The introduction of Patton et al. (2023) highlights the significance of noninvasive photo-identification (photo-id) in studying various aspects of animal ecology, particularly in cetaceans, while addressing the resource-intensive nature of the matching process required to identify individuals across images. The authors propose a multi-species automated photo-id model leveraging deep learning techniques, specifically convolutional neural networks, to enhance efficiency and accuracy in identifying individuals from large datasets, thereby facilitating ecological research and conservation efforts. This model not only improves identification across species with shared characteristics but also addresses the limitations of single-species approaches, particularly for rare or less-documented species, by utilizing transfer learning principles.

3. Scripts

Function to load images and extract label - load_dataset_with_labels()
Function to process images into manageable size for the model - load_and_preprocess_image()
Function to split dataset into testing and training - train_test_split()

New train.py script to train and save the model. ~20 hours training time. 2,422,597 total parameters.

4. Validation

Model summary:

Layer (type)	Output Shape	Param #
mobilenetv2_1.00_224 (Functional)	(None, 8, 8, 1280)	2257984
global_average_pooling2d (GlobalAveragePooling2D)	(None, 1280)	0
dense (Dense)	(None, 128)	163968
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 5)	645

==========

Total params: 2,422,597
Trainable params: 2,026,053
Non-trainable params: 396,544

5. Documentation

This week was spent creating a model to use transfer learning with the MobileNetV2model. The MobileNetV2 model was wrapped with extra layers that were used to tune the overall model for our dataset. Training time is 20 hours and with a failure at the end of the first training two rounds of training were performed. Accuracy, Recall, and precision were all ~95% on the training data. Evaluation with testing data to determine if the model is overfitted is necessary.

6. Next Weeks Proposal

Evaluation of the model on unseen testing data will be performed. This will determine if the model has over fit the training data. Investigation on accuracy, precision, and recall of each individual class will also be evaluated.

Weekly Report

Philip Woolley

2024-09-20

Time Log Reponse:

- What Progress did you make in the last week? - Performed full segmentation for another case. Developed script to convert segmented images into dataset for use with ML libraries.
- What are you planning on working on next? - Continue segmenting training data. Trial run Mask2Former on existing segmented images.
- Is there anything blocking you? - None at this time

1 Abstract

Abstract

Background

Algorithmic cellular segmentation is an essential step for the quantitative analysis of highly multiplexed tissue images. Current segmentation pipelines often require manual dataset annotation and additional training, significant parameter tuning, or a sophisticated understanding of programming to adapt the software to the researcher's need. Here, we present CellSeg, an open-source, pre-trained nucleus segmentation and signal quantification software based on the Mask region-convolutional neural network (R-CNN) architecture. CellSeg is accessible to users with a wide range of programming skills.

Results

CellSeg performs at the level of top segmentation algorithms in the 2018 Kaggle Data Challenge both qualitatively and quantitatively and generalizes well to a diverse set of multiplexed imaged cancer tissues compared to established state-of-the-art segmentation algorithms. Automated segmentation post-processing steps in the CellSeg pipeline improve the resolution of immune cell populations for downstream single-cell analysis. Finally, an application of CellSeg to a highly multiplexed colorectal cancer dataset acquired on the CO-Detection by indEXing (CODEX) platform demonstrates that CellSeg can be integrated into a multiplexed tissue imaging pipeline and lead to accurate identification of validated cell populations.

Conclusion

CellSeg is a robust cell segmentation software for analyzing highly multiplexed tissue images, accessible to biology researchers of any programming skill level.

Summary

This paper proposes an automatic cell segmentation software, developed using mask R-CNN for cell segmentation and a series of preprocessing and postprocessing steps. What the authors of this paper do particularly well is adapt their software for users with or without programming experience. This is an important aspect of final user design for my project, as this pipeline should be usable by biology experts with little programming experience as well as easily extensible by future researchers. The authors also do a good job of visually comparing the performance of their solution to state of the art models. The structure of the paper focuses more on the application than the technical implementation details, which is also a nod to the intended audience. Streamlining writing to not focus on technical details could be a good strategy for my project also.

Citation

Lee, M.Y., Bedia, J.S., Bhate, S.S. et al. CellSeg: a robust, pre-trained nucleus segmentation and pixel quantification software for highly multiplexed fluorescence images. BMC Bioinformatics 23, 46 (2022). https://doi.org/10.1186/s12859-022-04570-9

2 Scripts and Code Blocks

This week, I created the DataPreprocess.ipynb notebook. It is a draft of the process to convert slicer volume files (.nrrd and .seg.nrrd) into a HuggingFace dataset for use with the pretrained Mask2Former model. This involves several steps. For each frame of the volume in the sagittal view, convert the frame to a PIL image and process the corresponding mask frame. The mask frames are processed by separating each individual segment in them. Each segment is given a unique segment identifier, and bounding box information is recorded. Each mask frame corresponds to a list of information about the segments in it, of the form: [{'area': 27025, 'bbox': [0, 0, 115, 235], 'category_id': 0, 'id':

1}]. The category IDs for this dataset are [0: background, 1: lower jaw, 2: lower teeth, 3: all other bone]

```
VOLS_DIR = r"C:\Users\pgmw9\Documents\Georgia Tech\HAAG\Lizard-Auto-Segmentation\vols"
MASK_VOLS_DIR = r"C:\Users\pgmw9\Documents\Georgia Tech\HAAG\Lizard-Auto-Segmentation\masks"
segno = 1
vols = glob.glob(VOLS_DIR + r"\*.nrrd")
print(vols)

for volpath in vols:

    filename = os.path.basename(volpath).split(".")[0]
    maskpath = MASK_VOLS_DIR + "/" + filename + "_with UJ.seg.nrrd"
    mask = iio.v3.imread(maskpath)
    vol = iio.v3.imread(volpath)
    for i in range(vol.shape[1]):
        segments_info = []
        imframe = vol[:, i, :]
        maskframe = mask[:, i, :].astype(np.int64)
        seglist = np.unique(maskframe)
        for j in seglist:
            msk = iio.v3.imread(maskpath)[:, i, :]
            t = [msk == j]
            msk[msk != j] = 0
            msk[tuple(t)] = 255
            maskframe[tuple(t)] = segno
            area = np.count_nonzero(msk)
            points = cv2.findNonZero(msk)
            bR = cv2.boundingRect(points)
            catid = 2
            if j == 1:
                catid = 1
            elif j == 0:
                catid = 0
            elif j == mask.max():
                catid = 3
            segments_info.append({'area': area, 'bbox': [bR[0], bR[1], bR[2], bR[3]], 'category_id': catid, 'id': segno})
            segno = segno + 1
        imlist.append(Image.fromarray(imframe))
        print(np.unique(maskframe))
        masklist.append(maskframe)
        seglist2.append(segments_info)
```

3 Documentation

The DataProcess.ipynb notebook is used for converting slicer volume files (.nrrd and .seg.nrrd) into a HuggingFace dataset for use with the pretrained Mask2Former model. Volumes should be added to the "vols" folder, and segmentation volumes should be added to the "masks" folder. T

https://www.morphosource.org/projects/0000C1059?locale=enpage=11sort=publication_status_s List of available MicroCT Datasets of anolis lizards that will be used for this project. When infrastructure for data storage is ready I will prepare documentation detailing the downloading and storage process.

https://slicermorph.github.io/ Documentation for SlicerMorph, an extension of the 3D slicer tool commonly used by Biologists. This is used for loading stacks of .tiff images as a volume in 3d slicer.

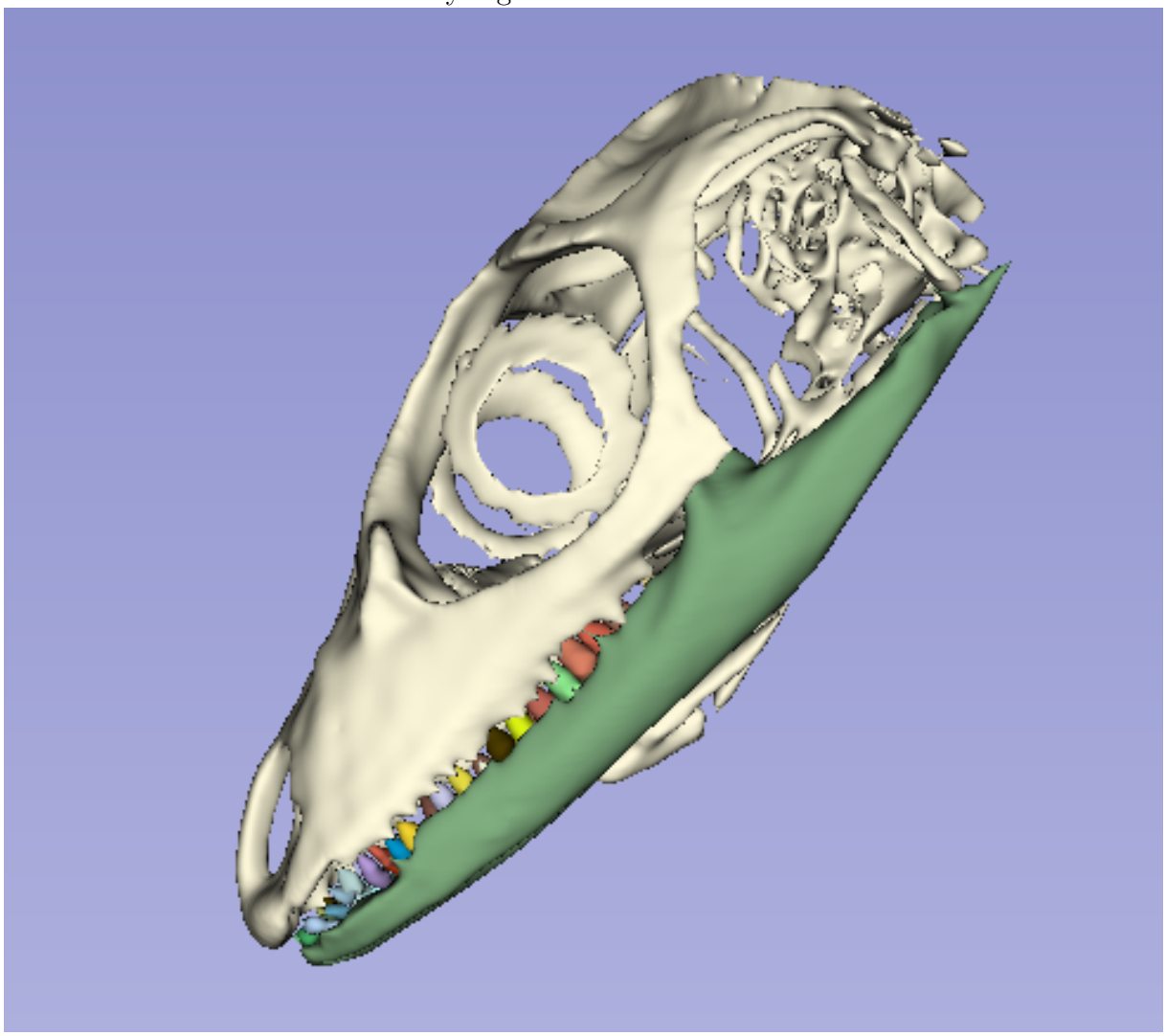
https://github.com/jmhuie/SlicerBiomech Documentation for the Dental Dynamics module, which is a 3D slicer extension for calculating tooth stress from jaw segmentations. the outputs from my segmentation pipeline will need to be compatible with this module for analysis.

4 Script Validation (Optional)

The code in DataPreprocess.ipynb is a proof of concept that will be rewritten into a .py script, full validation to come at that time.

5 Results Visualization

Here is a 3D model of the second fully segmented scan.



6 Proof of Work

See Code Blocks and Results Visualization sections

7 Next Week's Proposal

- Continue segmenting training data for ML panoptic segmentation model
- Develop training script for Mask2Former
- Keep up with any required blog posts for webmaster role

Week5 report

Ruiqing Wang | Lizard CV team

Time slot response:

- What progress did you make in the last week?
  1. Met with Dr. Stroud and discuss about further methods and resources
  2. Talked with Bree about current challenge and potential solution
  3. Run through the project again and confirm the problems
  4. Check my PACE allocation and evaluate performance
  5. Review papers on DeepLabCut
  6. Help assembling paper report submissions and address submission situation.
- What are you planning on working on next?
  1. Set up python and slurm code in project running on PACE
  2. Try google colab and check if this environment works
- Is anything blocking you from getting work done?

Trying to get network trained on new environment

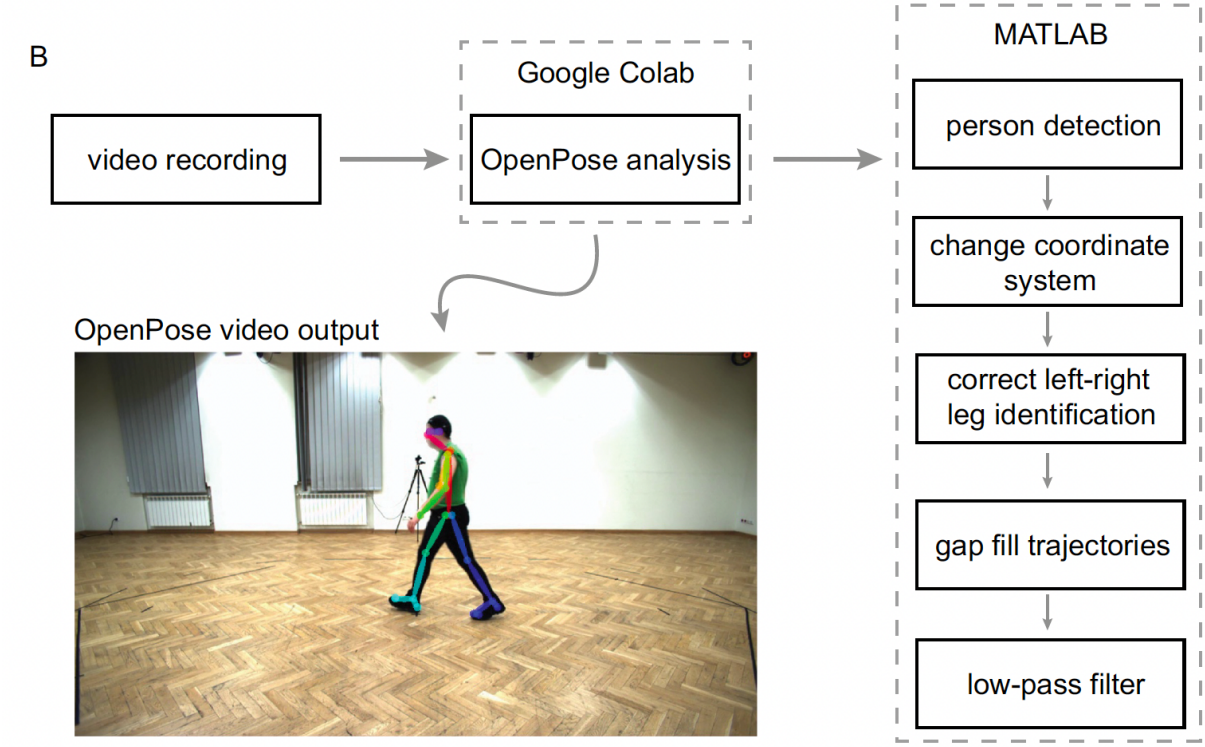
Paper abstract

Two-dimensional video-based analysis of human gait using pose estimation

Summary: The study introduces a novel approach utilizing two-dimensional video-based pose estimation through OpenPose, demonstrating its efficacy by comparing spatiotemporal and kinematic gait parameters against standard motion capture data. The results indicate that this method can achieve high accuracy in measuring gait parameters, with mean absolute errors that are minimal, thus providing a user-friendly and accessible alternative for gait analysis in both clinical and research settings.

Methodology: This paper highlights using MATLAB pose analysis including a. person detection b. Changing coordinate systems c. correcting left-right leg identification d. Gap-filling trajectories e. Applying a low-pass filter. There are parameters gaits they uses by step time, stance time, swing time, and double support time) between motion capture and OpenPose systems are minimal. They computed Pearson and intra-class correlation coefficients to assess the agreement between methods. In summary, they evaluate the accuracy and reliability of using OpenPose for video-based gait analysis by comparing it to established motion capture technology.

The project design is very well set up and may be helpful to my project:



This image illustrates a workflow for video-based human gait analysis. It starts with a video recording, which is then processed using OpenPose analysis in Google Colab. The output is a video showing a person walking, with their body segments highlighted in different colors. This data is then further processed in MATLAB. The workflow demonstrates an automated approach to analyzing human gait using computer vision and data processing techniques which could be helpful in future research.

Scripts and Code Blocks

To check my current PACE allocation and set up environment:

```
Gathering storage and job accounting information for user: rwang753

  ** Please note that the information and display format of this tool **
  ** is subject to change and should *not* be used for scripting.    **


================================================================================
                                        Welcome to the ICE Cluster!
================================================================================
 * Your Name (as PACE knows it)          :   Ruiqing Wang
 * UserID                                :   3288402
 * Username                              :   rwang753
 * Your Email (for PACE contact)         :

================================================================================
                                                ICE Storage
================================================================================
Filesystem                                                Usage (GB)    Limit
Home:/home/hice1/rwang753                                        0.0     30.0   0.0%
Scratch:/storage/ice1/0/2/rwang753                               0.0    300.0   0.0%
```

I set up my testing job.sub and try to submit it to hive system:

```
#!/bin/bash
#SBATCH -JSlurmCPlusExample             # Job name
#SBATCH --account=rwang753    #tracking account
#SBATCH -N1 -n4          # Number of nodes and cores per node required
#SBATCH --mem-per-cpu=1G                 # Memory per core
#SBATCH -t15                                # Duration of the job (Ex: 15 mins)
#SBATCH -phive                      # queue name(where job is submitted)
#SBATCH -oReport-%j.out                  # Combined output and error messages file
#SBATCH --mail-type=BEGIN,END,FAIL       # Mail preferences
#SBATCH --mail-user=rwang753@gatech.edu # E-mail address for notifications
cd $SLURM_SUBMIT_DIR                     # Change to working directory

#/home/hice1/rwang753/scratch/week5

echo "TASKS_PER_NODE=" $SLURM_TASKS_PER_NODE
echo "NNODES=" $SLURM_NNODES
echo "NTASKS" $SLURM_NTASKS
echo "JOB_CPUS_PER_NODE" $SLURM_JOB_CPUS_PER_NODE
echo $SLURM_NODELIST

module load gcc/12.3.0   #remain to be changed
module load mvapich2

mpicxx main.cpp -o mpi_main
mpirun ./mpi_main
```

Documentation

For sbatch work submission, here is the link I used for reference:
https://gatech.service-now.com/technology?id=kb_article_view&sysparm_article=KB0042003

Results Visualization

All my current code samples were stored in rwang753/home/hice1/rwang753/scratch

Proof of Work:

Currently I have my cpu test and gpu test code uploaded to PACE. However, I am facing the issue with partition and account error which I can't fix. I have reached to support team and hopefully to figure this out.

```
[rwang753@login-ice-2 week5]$ sbatch job.sub
sbatch: error: invalid partition specified: hive
sbatch: error: Batch job submission failed: Invalid partition name specified
[rwang753@login-ice-2 week5]$ vim job.sub
[rwang753@login-ice-2 week5]$ ls
gpu_test.py  gpu_test.sub  job.sub  main.cpp
[rwang753@login-ice-2 week5]$
```

The gpu_test shown below:

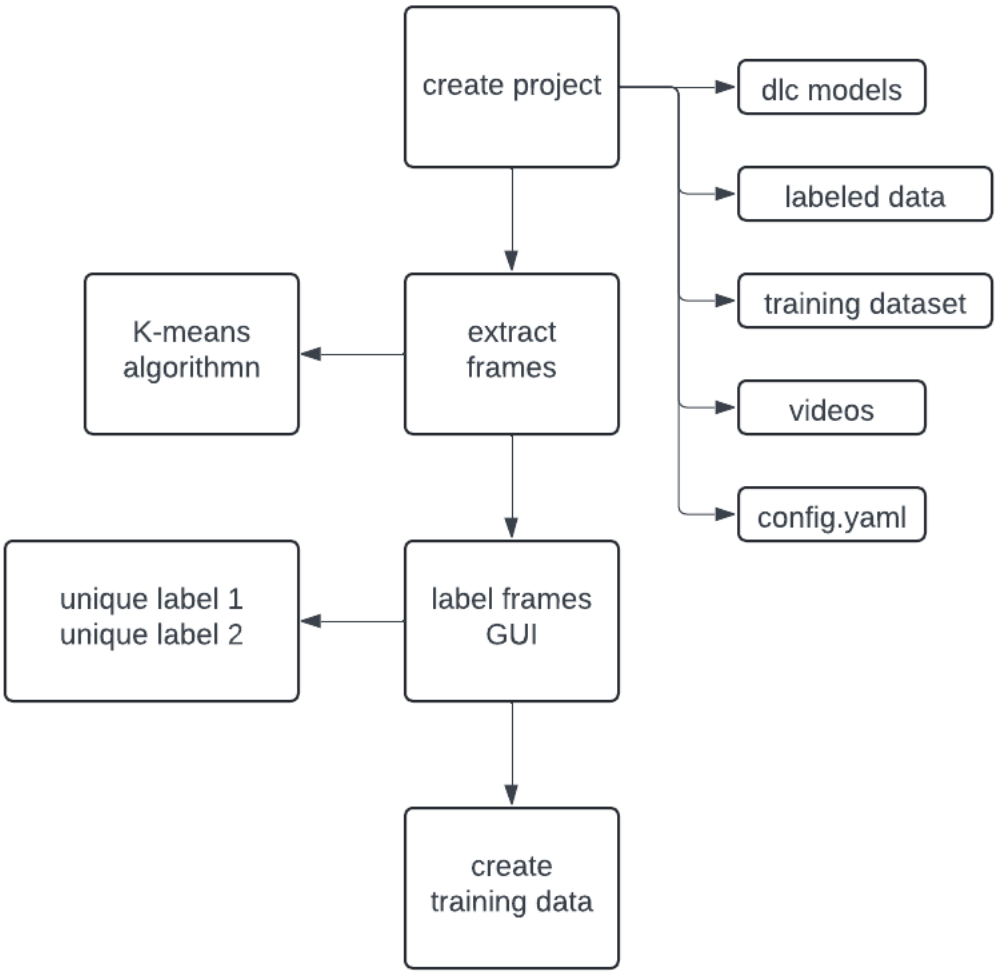
```
#!/bin/bash
#SBATCH -GPUExample_RW # Job name
#SBATCH -Anlytle3 # Charge account
#SBATCH -N1 --gres=gpu:1 # Number of nodes and GPUs required
#SBATCH --gres-flags=enforce-binding # Map CPUs to GPUs
#SBATCH --mem-per-gpu=12G # Memory per gpu
#SBATCH -t15 # Duration of the job (Ex: 15 mins)
#SBATCH -phive-gpu # Partition name (where job is submitted)
#SBATCH -oReport-%j.out # Combined output and error messages file
#SBATCH --mail-type=BEGIN,END,FAIL # Mail preferences
#SBATCH --mail-user=rwang753@gatech.edu # e-mail address for notifications
cd /home/hice1/rwang753/scratch/week5 # Change to working directory created in

/home/hice1/rwang753/scratch/week5

module load tensorflow-gpu/2.9.0 # Load module dependencies
srun python $TENSORFLOWGPUROOT/gpu_test.py gpu 1000 # Run test example
~
~
~
```

And I got similar error:

```
[rwang753@login-ice-2 week5]$ sbatch gpu_test.sub
sbatch: error: Invalid GRES specification (with and without type
identification)
```

My current plan is to solve current partition issue, clearly there is something wrong with partition name which related to the account. I am reaching out to the support team, and at the same time, I am looking at Google Colab to check if I can get my work done there.

For DeepLabCut project, here is the detailed code flow that I am trying to follow in colab:



Next Week's Proposal

1. Solve PACE issue with account and partition
2. Set up my code project on Google Colab

Week 4 Document Submission

Lizard X-RAY Landmark Group

Mercedes Quintana

What progress did you make in the last week?

- Continued to work on website
- Reworked visualizing scripts to work with updated version of ml-morph
- Created a new visualizing script for an individual lizard
- Finalized PACE access

What are you planning on working on next?

- Rework grid search script to work with new ml-morph library
- Gridsearch possible model hyperparameters to create the best model
- Meet with Dr. Stroud to share results
- Continue to update the website

Is anything blocking you from getting work done?

- Nope

Abstracts:

URL: https://www.sciencedirect.com/science/article/abs/pii/S1077314219301444

Face alignment using a 3D deeply-initialized ensemble of regression trees

Face alignment algorithms locate a set of landmark points in images of faces taken in unrestricted situations. State-of-the-art approaches typically fail or lose accuracy in the presence of occlusions, strong deformations, large pose variations and ambiguous configurations. In this paper we present 3DDE, a robust and efficient face alignment algorithm based on a coarse-to-fine cascade of ensembles of regression trees. It is initialized by robustly fitting a 3D face model to the probability maps produced by a convolutional neural network. With this initialization we address self-occlusions and large face rotations. Further, the regressor implicitly imposes a prior face shape on the solution, addressing occlusions and ambiguous face configurations. Its coarse-tofine structure tackles the combinatorial explosion of parts deformation. In the experiments performed, 3DDE improves the state-of-the-art in 300W, COFW, AFLW and WFLW data sets. Finally, we perform cross-dataset experiments that reveal the existence of a significant data set bias in these benchmarks.

Summary: 3DDE uses a cascade of ensembles of regression trees from the output of a CNN to improve upon previous methods that stumble on large deviations in posing and occlusions.

Scripts and Code Blocks:

Thanks to Dr. Porto, we have a much higher accuracy to work from than the beginning of the semester. He revised the ml-toolbox to remove the first shape finding step that was not needed for our images. I revised the landmark_skew.py and created a new script visual individual performance.py to look at individual lizards and their predictions.

Found in landmark_skew.py:

Read in test tps data for model and ground truth -> calculate common feature on each X-Ray for a conversion factor to millimeters -> calculate difference between ground truth and model output -> convert to millimeters -> find displacement from landmark -> find direction of error -> estimate a kernel density function of error -> display 3 graphs with matplotlib (kernel density function, histogram of length error and rose plot of angle error.

Found in visual_individual_performance.py:

Read in test xml for model and ground truth -> Find specific lizard -> Plot over image

Documentation:

Landmark_skew.py:

1. Read in data (found in XRAY Lizard Github) for ground and model output
2. Choose a specific landmark to visualize
3. Find the differences between the ground truth and the model output in pixels.
4. Find the difference angles of error
5. Convert and store millimeter conversion using each individual staple.
6. Estimate kernel density function
7. Display with matplotlib

visual_individual_performance.py:

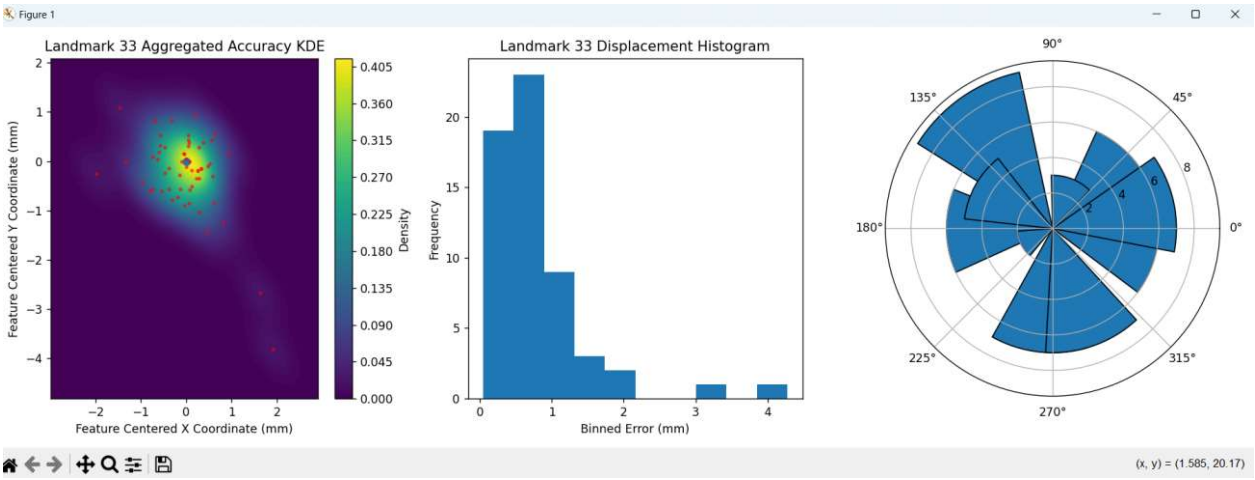
1. Read in test.xml and output.xml with number of image to visualize
2. Code will read in and display lizard image with output and ground truth

Script Validation:

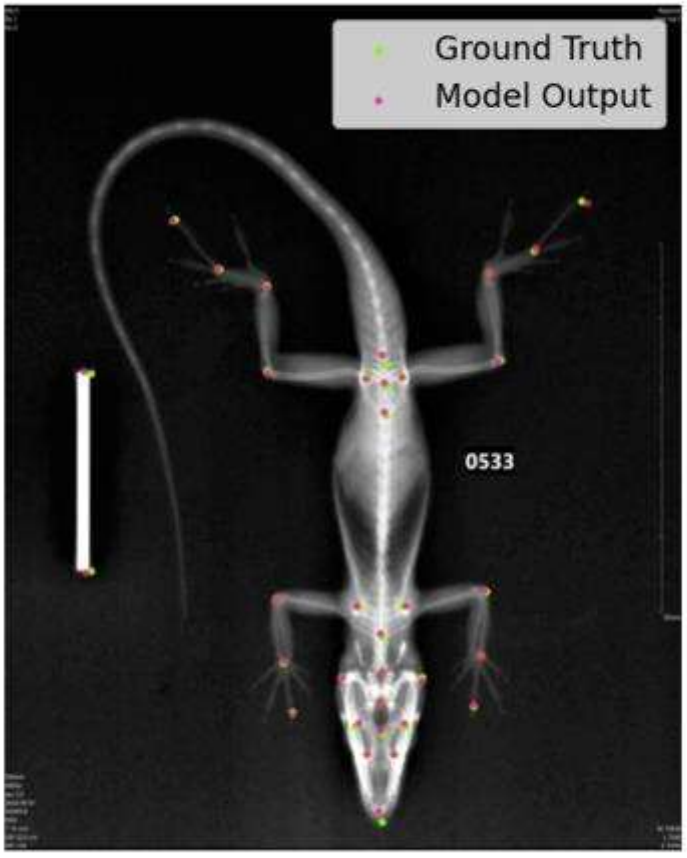
I have no validation steps now.

Results Visualization / Proof of Work:

This is the error visualized from Landmark 33, one of the worst performing areas on the lizard



Here is an example of the output of individual lizard with the great accuracy from Dr. Porto.



Next Week Proposal:

I plan to keep working on the website to keep it updated with the new meetings and work done. I plan to rework my grid search script to work with the updated toolbox and run a

more thorough grid search on PACE. Lastly I plan to rework the above code to have the options to either view one output or save all outputs to a folder.