Week 14 Document Submission

Jacob Dallaire

November 22, 2024

1. Paper

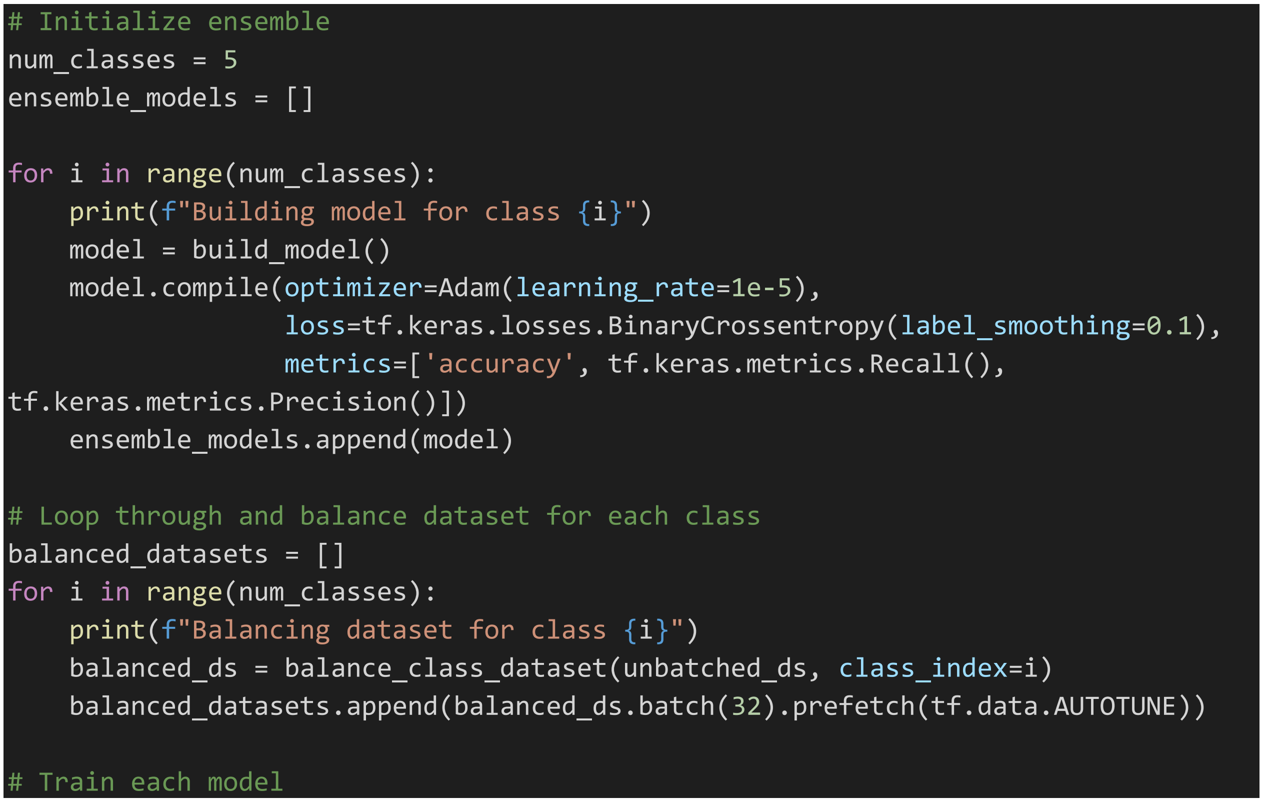
Bartosz Krawczyk, Mikel Galar, Michał Woźniak, Humberto Bustince, Francisco Herrera, Dynamic ensemble selection for multi-class classification with one-class classifiers, Pattern Recognition,Volume 83, 2018, Pages 34-51,ISSN 0031-3203, https://doi.org/10.1016/j.patcog.2018.05.015.

SUMMARY

Classification problems with multiple classes can be complex because as the number of classes increases, it becomes harder to define clear boundaries between them. To make these problems easier to solve, researchers often break them down into simpler binary classification tasks, which can be handled by various classifiers. Two common strategies for this are One-vs-One (OVO), where a classifier is trained for every pair of classes, and One-vs-All (OVA), where one classifier is trained for each class against all others, helping to improve classification performance.

2. Scripts

New script to use a one vs all series of classifiers. Not yet tested.



```
# Initialize ensemble
num_classes = 5
ensemble_models = []

for i in range(num_classes):
    print(f"Building model for class {i}")
    model = build_model()
    model.compile(optimizer=Adam(learning_rate=1e-5),
                  loss=tf.keras.losses.BinaryCrossentropy(label_smoothing=0.1),
                  metrics=['accuracy', tf.keras.metrics.Recall(),
tf.keras.metrics.Precision()])
    ensemble_models.append(model)

# Loop through and balance dataset for each class
balanced_datasets = []
for i in range(num_classes):
    print(f"Balancing dataset for class {i}")
    balanced_ds = balance_class_dataset(unbatched_ds, class_index=i)
    balanced_datasets.append(balanced_ds.batch(32).prefetch(tf.data.AUTOTUNE))

# Train each model
```

```
history_list = []
for i, model in enumerate(ensemble_models):
    print(f"Training model for class {i}")
    binary_dataset = balanced_datasets[i]

    # Early stopping callback
    early_stopping_callback = tf.keras.callbacks.EarlyStopping(
        monitor='loss',
        patience=5,
        restore_best_weights=True
    )

    # Train the model
    history = model.fit(
        binary_dataset,
        epochs=10,
        #class_weight={0: 1, 1: class_weight_dict[i]},  # Class weights for binary classification
        callbacks=[early_stopping_callback]
    )
    history_list.append(history)
```

3. Documentation

The beginning of the week was spent trying to improve the multi-class classifier’s accuracy with little success. After speaking with Bri about some of the challenges being faced by this approach, I am attempting to implement one vs all classification.

Results from the multi-class classifier showed predictions have some preference to the correct class but often have difficulty identifying between at least one other class. Equestris, the smallest class, in particular was a frequent misclassification.

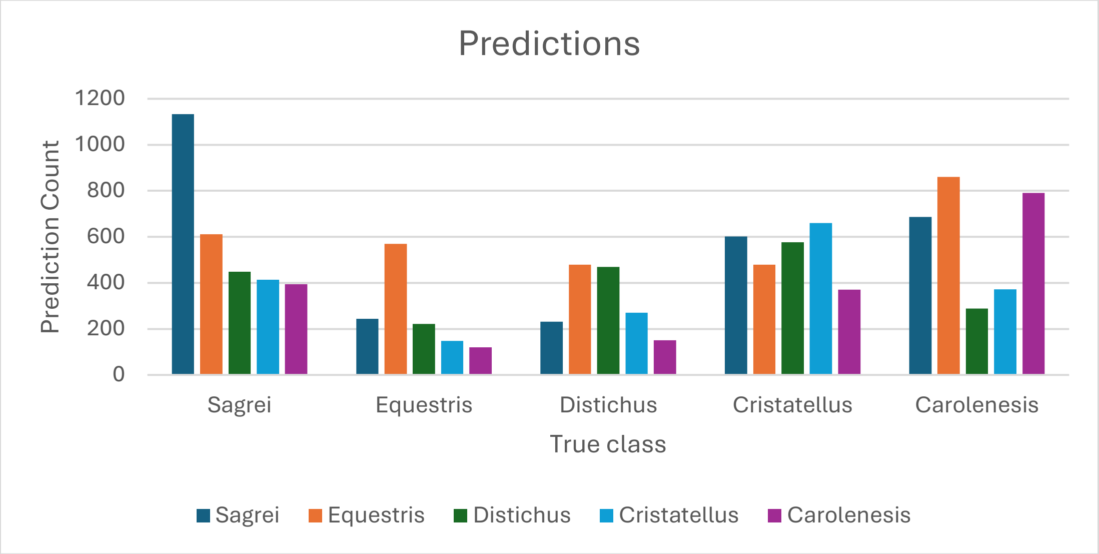


Figure 1 classification predictions

4. Next Weeks Proposal

I will be testing the implementation of the one vs all classifier as well I will test using some different base model to do transfer learning on. Currently I am using mobile net V2.

Weekly Report

Philip Woolley

2024-11-22

Time Log Reponse:

- Retrained segmentation model with additional data. Started final project documentation
- Retrain model with additional data and new metric. Continue final project documentation. Change website structure to match request from Bree
- Is there anything blocking you? - Access to "Appearance" tab on wordpress needed to rearrange navigation menu

1 Abstract

Abstract The biggest challenge to improve the diagnosis and therapies of Craniomaxillofacial conditions is to translate algorithms and software developments towards the creation of holistic patient models. A complete picture of the individual patient for treatment planning and personalized healthcare requires a compilation of clinician-friendly algorithms to provide minimally invasive diagnostic techniques with multimodal image integration and analysis. We describe here the implementation of the open-source Craniomaxillofacial module of the 3D Slicer software, as well as its clinical applications. This paper proposes data management approaches for multisource data extraction, registration, visualization, and quantification. These applications integrate medical images with clinical and biological data analytics, user studies, and other heterogeneous data.

Summary This paper is a writeup describing a 3D slicer module for working with dental, cranial, and maxillofacial data. It shows a breakdown of the various capabilities of the module, as well as implementation details listed in the related github repository. As the final output of my project may take the form of a slicer module, it is important to understand how these modules are commonly presented. This paper does not provide much in terms of technical detail or instructions of how the tool is used, which is information that I would like to include as it will make the process of continuing development much easier for the next student. There is no real information given about the design or implementation of any of the tools that make up the module.

Citation Bianchi, J. et al. (2020). 3D Slicer Craniomaxillofacial Modules Support Patient-Specific Decision-Making for Personalized Healthcare in Dental Research. In: Syeda-Mahmood, T., et al. Multimodal Learning for Clinical Decision Support and Clinical Image-Based Procedures. CLIP ML-CDS 2020 2020. Lecture Notes in Computer Science(), vol 12445. Springer, Cham. https://doi.org/10.1007/978-3-030-60946-7$_5$

2 Scripts and Code Blocks

by applying the inverse of the two transformations and cropping according to the bounding box, we can transform the initial model into an axis-aligned model of only the head.

3 Documentation

The VisualizeModelResults.ipynb notebook is used for creating and viewing images of model output on validation data. Users provide a pretrained model and validation dataset, and this notebook inferences all of the images in the dataset and allows the user to review the output segmentations against the ground truth manual segmentations.

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.

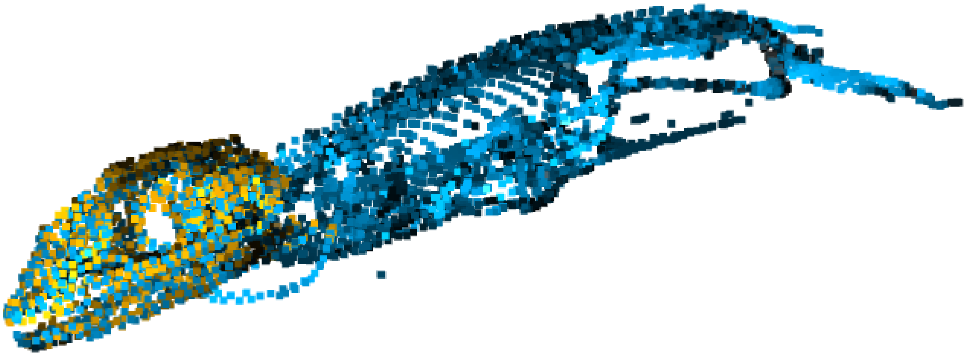
https://www.morphosource.org/projects/0000C1059?locale=enpage=11sort=publication_status_s List of available MicroCT Datasets of anolis lizards that will be used for this project. When infrastructure for data storage is ready I will prepare documentation detailing the downloading and storage process.

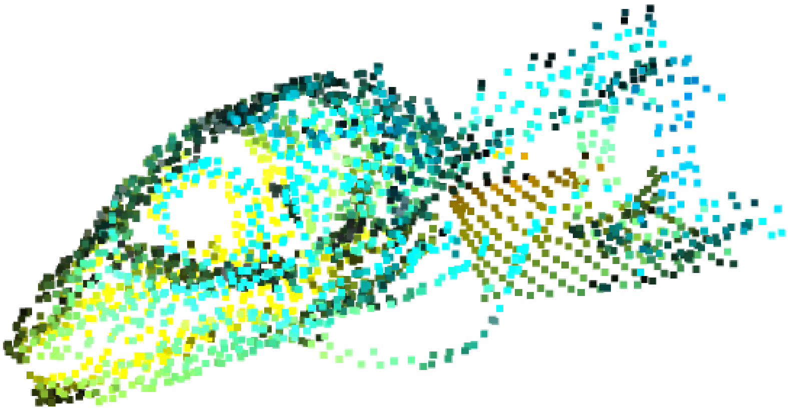
https://slicermorph.github.io/ Documentation for SlicerMorph, an extension of the 3D slicer tool commonly used by Biologists. This is used for loading stacks of .tiff images as a volume in 3d slicer.

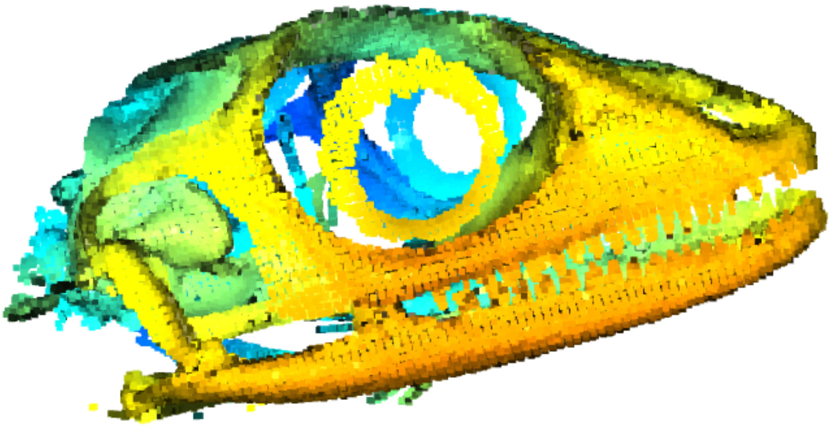
https://github.com/jmhuie/SlicerBiomech Documentation for the Dental Dynamics module, which is a 3D slicer extension for calculating tooth stress from jaw segmentations. the outputs from my segmentation pipeline will need to be compatible with this module for analysis.

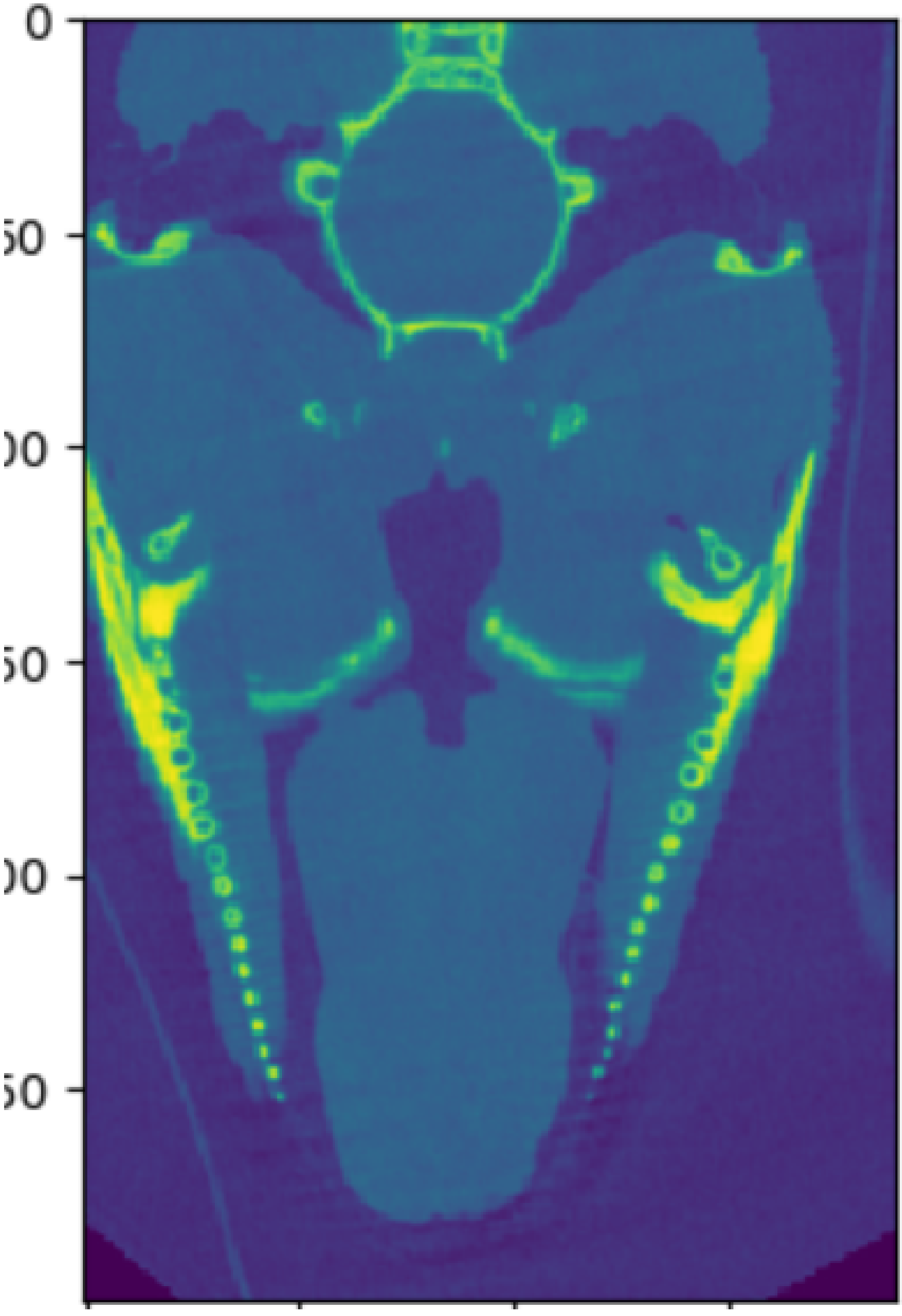
4 Script Validation (Optional)

5 Results Visualization








0
50
00
50
00
50

6 Proof of Work

Please see results visualization section

7 Next Week's Proposal

- Continue segmenting training data for ML panoptic segmentation model
- Continue project documentation
- Reformat blog page as requested by Bree

Week14 report

Ruiqing Wang | CiChild CV team

▪ What progress did you make in the last week?

1. Refine the dataset and keep training networks on SLEAP
2. Finish wrap up summary on DLC and SLEAP progress and protocol
3. Attend Cichild group meeting with current work on Sleap
4. Review papers on SLEAP and pose estimation
5. Help assembling paper report submissions and address submission situation.

▪ What are you planning on working on next?

1. Keep training network on SLEAP and refine dataset
2. Meet with Cichild CV team to discuss current progress

▪ Is anything blocking you from getting work done?

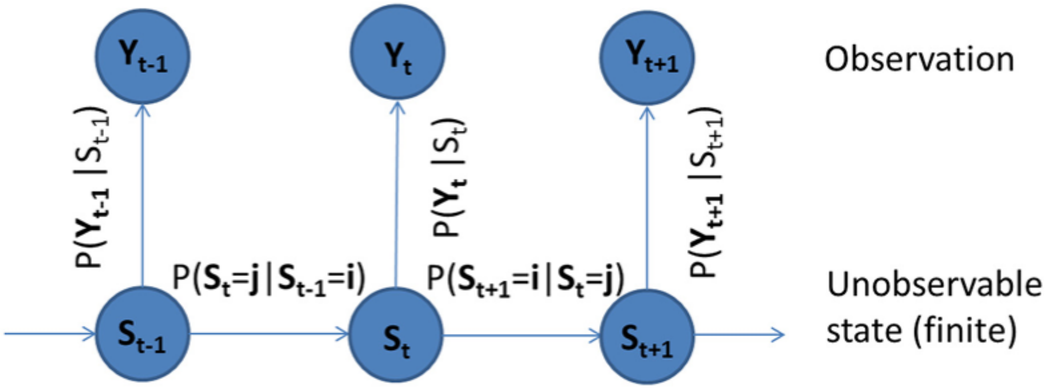
N/A

Paper abstract

Paper: Machine learning for inferring animal behavior from location and movement data
https://www.sciencedirect.com/science/article/pii/S1574954118302036

Abstract: Movement ecology has significantly progressed due to advancements in animal-attached technologies, such as GPS transmitters and tri-axial accelerometers, which facilitate the collection of high-resolution location and locomotion data. The integration of sophisticated statistical and machine learning techniques, including state space models and hidden Markov models, allows researchers to infer behavioral patterns from movement data, accounting for measurement errors and estimating true animal locations. This study emphasizes the utility of both supervised and unsupervised learning algorithms in behavioral classification, particularly highlighting the challenges posed by the lack of synchronized behavioral observations in many species, which often necessitates the use of unsupervised methods for analyzing cryptic or less-studied animals.

Review Method:
A Hidden Markov Model (HMM) for analyzing animal behavior consists of a set of hidden behavioral states (e.g., foraging and migrating) that transition according to probabilities P(St=j|St−1=i)P(S_t = j | S_{t-1} = i), forming a Markov chain. Observations such as step lengths and turning angles are generated from state-specific probability distributions (e.g., gamma or Weibull for step lengths, wrapped Cauchy for turning angles). Using these distributions and transition probabilities, algorithms like the Viterbi algorithm assign hidden states to observed data. HMMs enable segmentation of movement paths into distinct behavioral states, providing insights into transitions and ecological patterns while leveraging Bayesian or frequentist methods for parameter estimation. Despite limitations like handling observational errors and complexities with multiple states, HMMs remain a powerful tool for understanding animal movement and behavior.

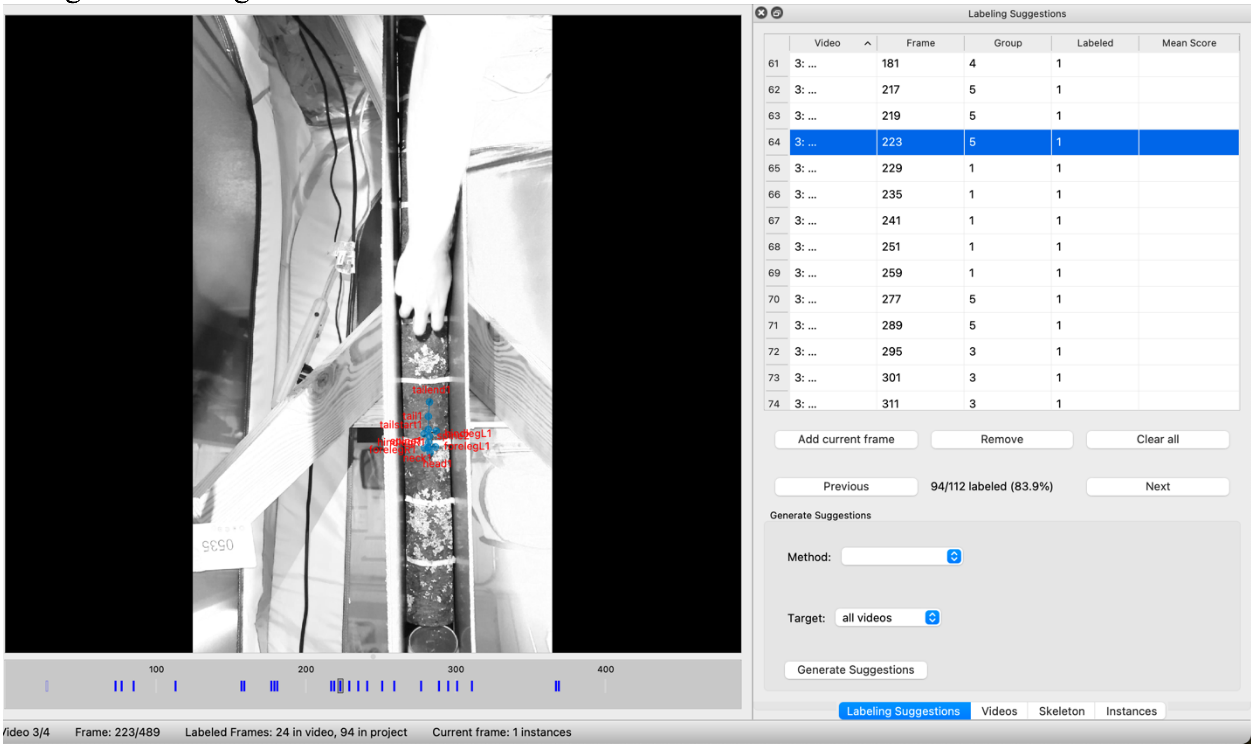


Scripts and Code Blocks

This week I mainly trained the network and modify data annotations and did writings.

Documentation

For SLEAP dataset, there exist a incompatible problems in data annotations. I refined the dataset several times and delete frames which didn't have the label and re-train the networks.
In the new dataset, I have labeled 94 frames, and according to the past experience, it should be enough for training on networks.



Labeling Suggestions

	Video	Frame	Group	Labeled	Mean Score
61	3: ...	181	4	1	
62	3: ...	217	5	1	
63	3: ...	219	5	1	
64	3: ...	223	5	1	
65	3: ...	229	1	1	
66	3: ...	235	1	1	
67	3: ...	241	1	1	
68	3: ...	251	1	1	
69	3: ...	259	1	1	
70	3: ...	277	5	1	
71	3: ...	289	5	1	
72	3: ...	295	3	1	
73	3: ...	301	3	1	
74	3: ...	311	3	1	

94/112 labeled (83.9%)

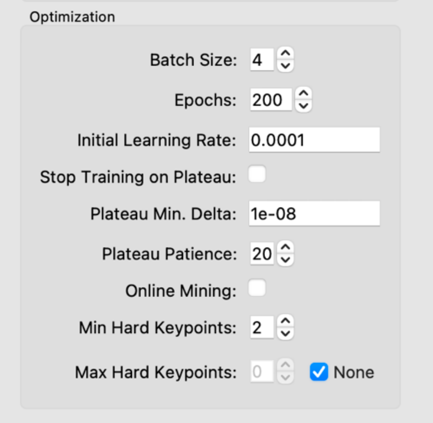
Generate Suggestions

Method:

Target: all videos

/ideo 3/4 Frame: 223/489 Labeled Frames: 24 in video, 94 in project Current frame: 1 instances

On setting the parameters, this time I forced the system to run 200 epochs without stopping early, here is the overview of the epochs loss.



Optimization

Batch Size: 4

Epochs: 200

Initial Learning Rate: 0.0001

Stop Training on Plateau: □

Plateau Min. Delta: 1e-08

Plateau Patience: 20

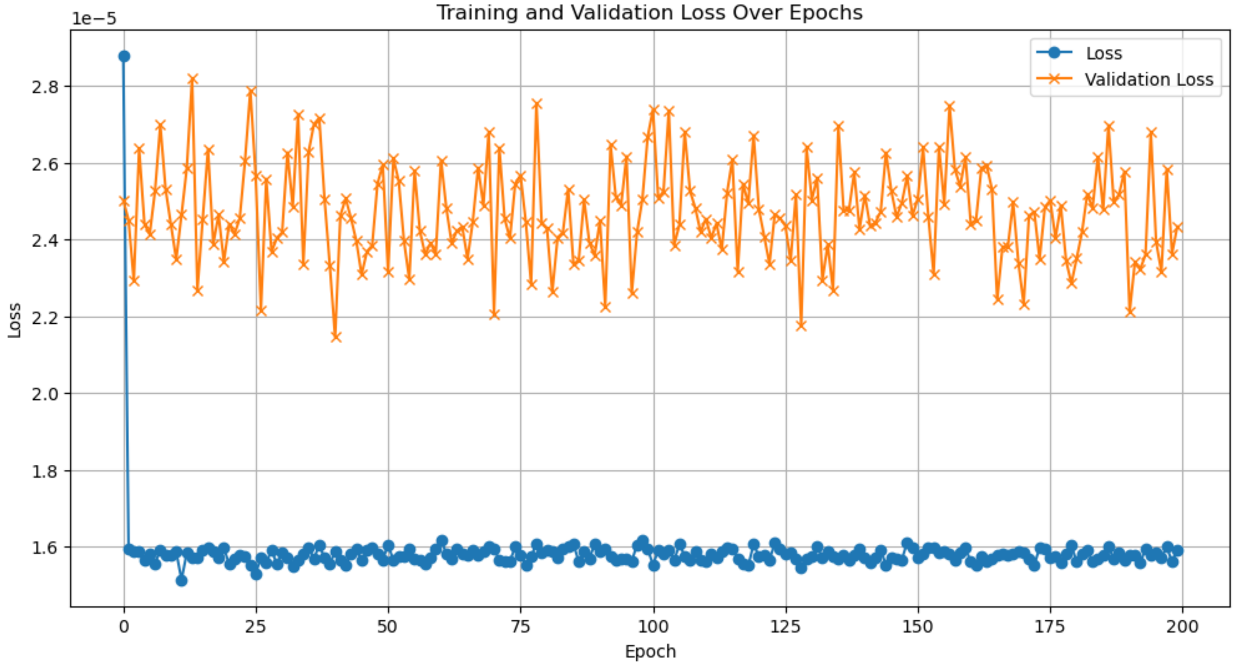
Online Mining: □

Min Hard Keypoints: 2

Max Hard Keypoints: 0 ☑ None

Results Visualization and Code Validation

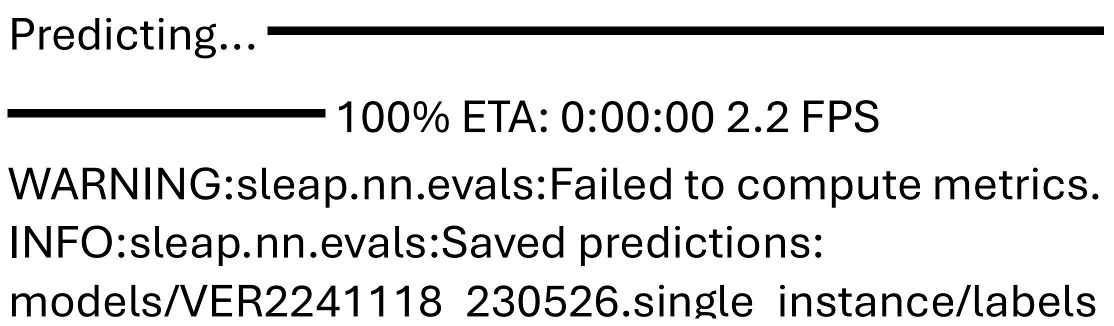
Here is the epochs loss overview:



The plot shows a comparison of training and validation loss over 200 epochs. While the training loss remains consistently low, the validation loss is highly fluctuating and significantly higher than the training loss. This indicates potential overfitting, where the model learns to perform well on the training data but generalizes poorly to unseen data. The early stopping is verified from the last trying, so next step I may increase the dataset, get more labeled data and start training again.

Proof of Work and code validation

So here is the output of my training networks:



```
Predicting... ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
━━━━━━━━━━━━━━━ 100% ETA: 0:00:00 2.2 FPS
WARNING:sleap.nn.evals:Failed to compute metrics.
INFO:sleap.nn.evals:Saved predictions:
models/VER2241118_230526.single_instance/labels
```

That’s the same output from last run, which indicated the overfitting problems, which suggested that we may need more dataset or simplify the model architecture.

The other part I was doing is the writing on the summary of SLEAP and DLC network training and video production, here is the screenshot:

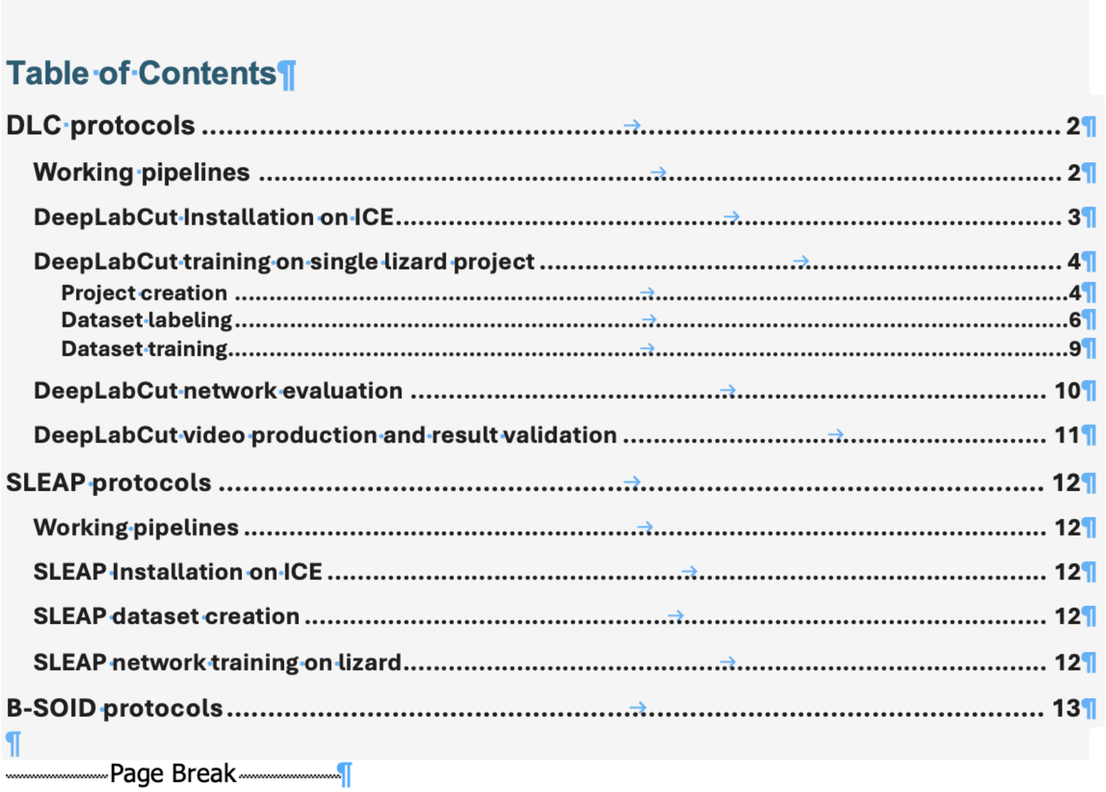


Table of Contents

Page Break

Next Week's Proposal

1. Keep training network on SLEAP
2. Label more dataset on lizard videos
3. Meet with Cichild CV team to discuss current progress

Week 14 Document Submission

Lizard X-RAY Landmark Group

Mercedes Quintana

What progress did you make in the last week?

- Addressed comments on lizard cv website
- Analyzed model outputs

What are you planning on working on next?

- Create new toe dataset to better classify toes
- Help Ayush create the website
- Continue to update the lizard cv website

Is anything blocking you from getting work done?

- Nope

Abstracts:

URL: https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1011194

Deep learning approaches to landmark detection in tsetse wing images

Morphometric analysis of wings has been suggested for identifying and controlling isolated populations of tsetse (Glossina spp), vectors of human and animal trypanosomiasis in Africa. Single-wing images were captured from an extensive data set of field-collected tsetse wings of species Glossina pallidipes and G. m. morsitans. Morphometric analysis required locating 11 anatomical landmarks on each wing. The manual location of landmarks is time-consuming, prone to error, and infeasible for large data sets. We developed a two-tier method using deep learning architectures to classify images and make accurate landmark predictions. The first tier used a classification convolutional neural network to remove most wings that were missing landmarks. The second tier provided landmark coordinates for the remaining wings. We compared direct coordinate regression using a convolutional neural network and segmentation using a fully convolutional network for the second tier. For the resulting landmark predictions, we evaluate shape bias using Procrustes analysis. We pay particular attention to consistent labelling to improve model performance. For an image size of 1024 × 1280, data augmentation reduced the mean pixel distance error from 8.3 (95% confidence interval [4.4,10.3]) to 5.34 (95% confidence interval [3.0,7.0]) for the regression model. For the segmentation model, data augmentation did not alter the mean pixel distance error of 3.43 (95% confidence interval [1.9,4.4]). Segmentation had a higher computational complexity and some large outliers. Both models showed minimal shape bias. We deployed the regression model on the complete unannotated data consisting of 14,354 pairs of wing images since this model had a lower computational cost and more stable predictions than the segmentation model. The resulting landmark data set was provided for future morphometric analysis. The methods we have developed could provide a starting point to studying the wings of other insect species. All the code used in this study has been written in Python and open sourced.

Summary: This paper uses a two-step method to landmark Tsetse fly wings. The first step removes incomplete wings and the second places landmarks. This method could be used to quickly landmark many flies.

Scripts and Code Blocks:

I analyzed the model outputs using code that I have written previously.

Documentation:

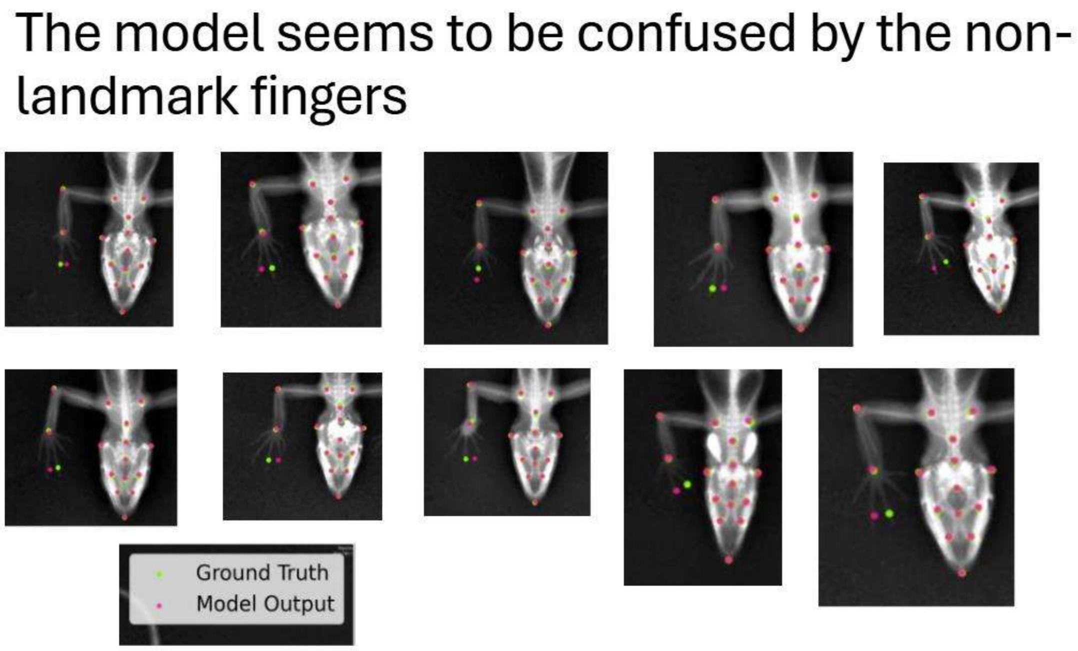
I have no new code to document as of now.

Script Validation:

I have no validation steps now.

Results Visualization / Proof of Work:

Here are some results I shared at the bi-weekly meeting:



The model seems to be confused by the non-landmark fingers

Next Week Proposal:

Next week I am going to continue to make sure the website is updated. I am going to get the lizard to show on the web app and continue to make it more user friendly. I am going to create a dataset of the lizard feet to help the classifier be more accurate.