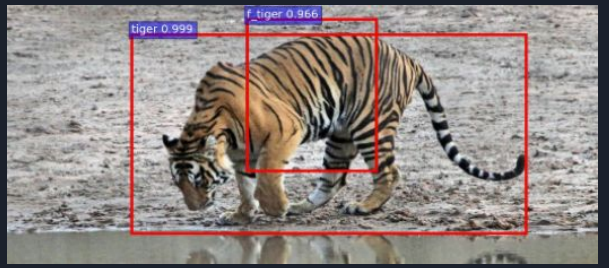
W210 Final Presentation

Safari Sleuths

By: Austin Jin, Brittany Dougall, Jackson Argo

Collaborators: Colorado Reed, Medhini Narasimhan, & Jacob Yeung from Berkeley AI Research Climate Initiative



Agenda

- Overview
- Datasets
- Solution Architecture
- Model
- Object Detection
- Individual Detection
- Results
- Web UI

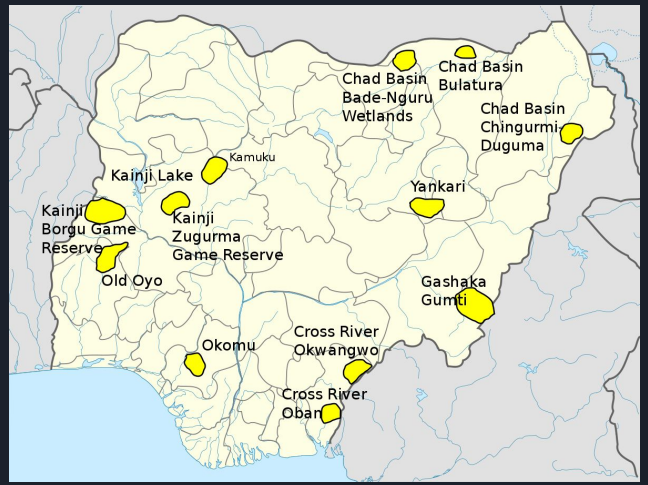
Overview

Using computer vision and machine learning techniques to make corrections to predictions of individual animals

- Poaching continues to drive population loss of endangered species, even on protected land
- Knowing species counts can aid conservationists in measuring natural changes vs poaching impact, deciding where to deploy resources, etc.
- Manually counting and identifying individual animals is time consuming and can be improved through automation

Who Could This Help?

BAIR Partner
Nigeria National Park Service



Other Potential
Partners

WWF

AFRICAN PARKS
A
P

Datasets

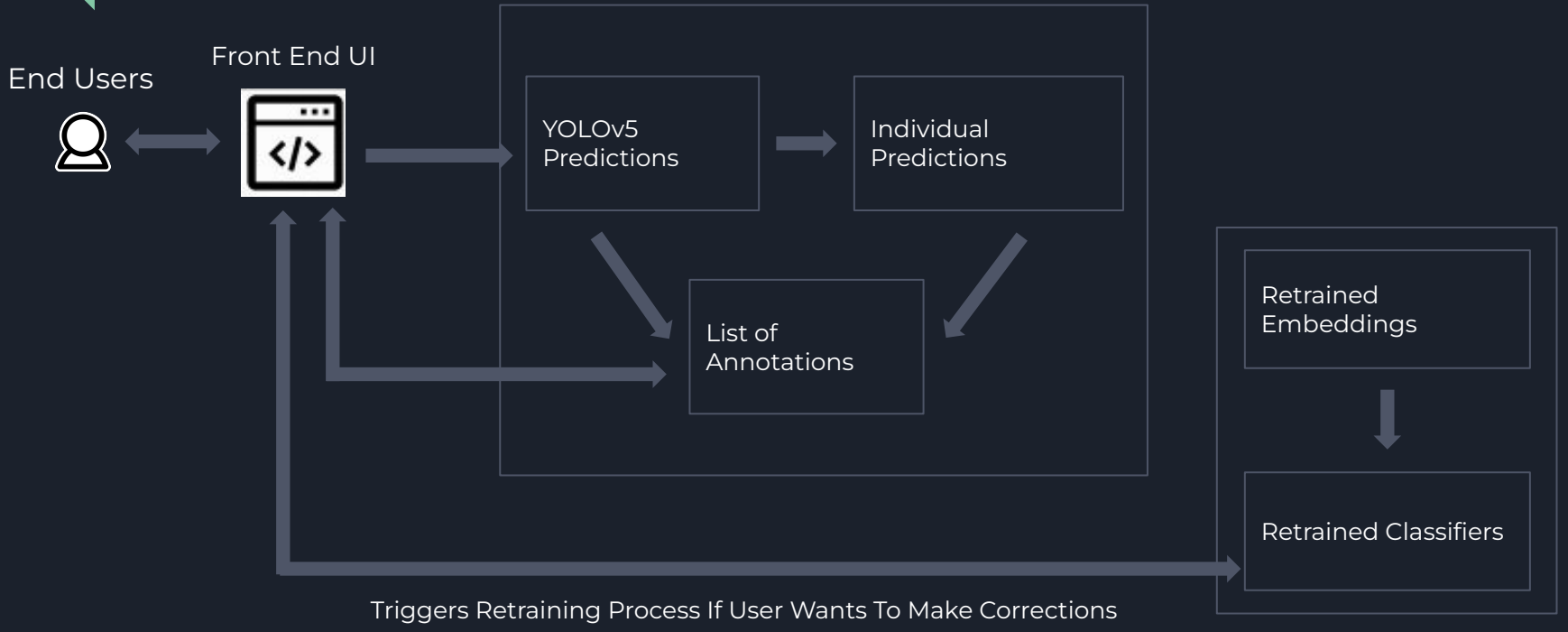
- Hyena & Leopard: Botswana Predator Conservation Trust
- Giraffe: Wild Nature Institute
- Coco annotations for bounding boxes & individual ids

Species	Train	Validation	Test
Hyena (Crocuta crocuta)	256 animals 2,506 images	162 animals 310 images	154 animals 312 images
Leopard (Panthera pardus)	430 animals 5,473 images	193 animals 673 images	186 animals 673 mages
Giraffe (Giraffa tippelskirchi)	151 animals 520 images	38 animals 52 images	39 animals 62 images

Annotations Fields Used

Field	Definition	Use
image id	Unique image identifier	train/validation/test splits
height	Image height in pixels	Coco bounding box conversion to Yolo format for Yolov5 training
width	Image width in pixels	Coco bounding box conversion to Yolo format for Yolov5 training
bbox	Coco bounding box coordinates based on original image size	Image cropping for individual recognition pipeline
individual_ids	A list of all of the images in which an individual has appeared	training/validation/test splits
name	Individual animal identifier	training/validation/test splits

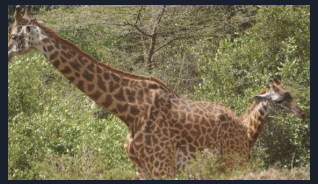
End-to-End Process



YOLOv5 Object Detection

Train Model

{“class”: 0, “x center”: 0.548, “y center”: 0.539, “width”: 0.461, “height”: 0.508}



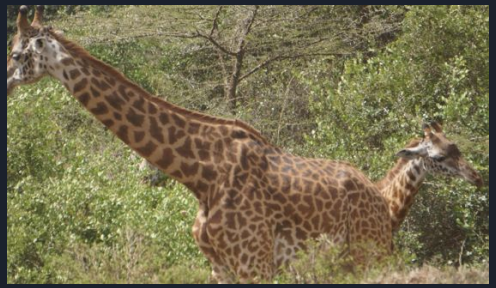
Custom classification yaml

Inference

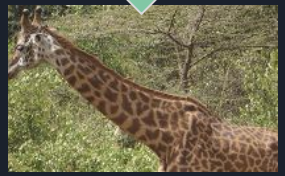
Why YOLOv5?

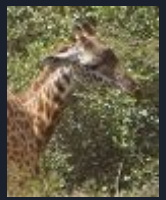
- Need object classification & localization to extract multiple individuals for individual classification
- Fast - predictions can be returned to the user quickly
- Can be retrained or fine-tuned for custom class prediction (species specific)

Object Detection to Individual Recognition



Yolov5

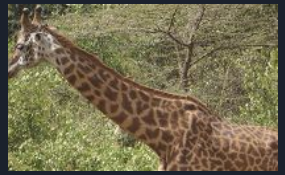




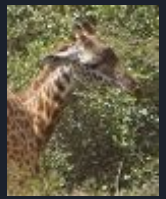
SimCLR

KNN
Classifier

Giraffe A

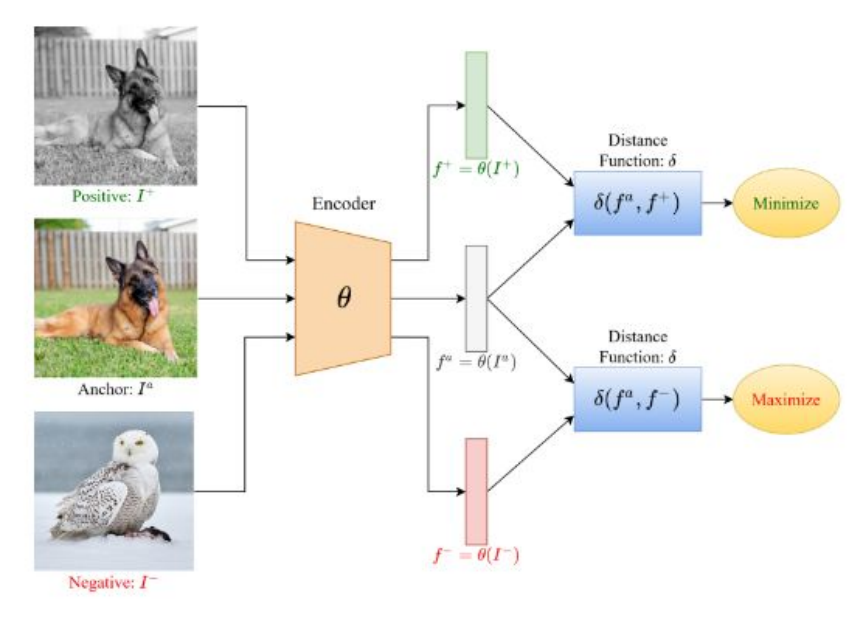


Giraffe B



```
tensor([[0.2992, 0.0716, 0.1516,  ..., 0.4713, 0.3572, 0.0891],
[0.4534, 0.2210, 0.1828,  ..., 0.5872, 0.1490, 0.0088]])
```

Individual Detection
Embedding + Classifiers



SimCLR v1 feature extraction

- Able to learn highly multi-class embeddings
- Generalizable embeddings

KNN Individual Classifiers

- Insufficient data to train classifier CNN
- Best performance compared to other commonly used classifiers
- Prior studies

YOLOv5 Object Detection Results

	Training	Validation
Bounding Box Loss	0.024	0.024
Object Loss	0.011	0.004
Class Loss	0.002	0.0003

Test Metrics	Hyena (Crocuta crocuta)	Leopard (Panthera pardus)	Giraffe (Giraffa tippelskirchi)
Accuracy	99.4%	99.6%	100%
Precision	0.985	0.994	1
Recall	0.988	0.993	1
F1-Score	0.987	0.993	1

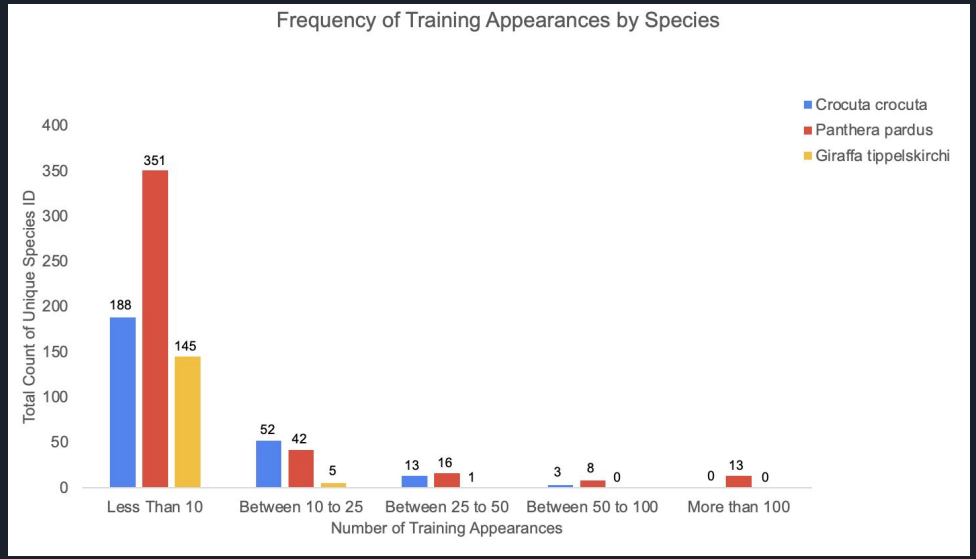
Training

- Freezing of select layers + addition of unlabeled images (10% of train set)

Findings

- Only 2% of images fail to be classified
- Only 9 incorrect classifications
- 92% of predicted boxes have confidence scores >50%

Individual Detection Results



Test Metrics	Crocuta crocuta	Panthera pardus	Giraffa tippelskirchi
Top-1 Accuracy	42.6%	62.0%	58.1%
Weighted Precision	0.441	0.613	0.737
Weighted Recall	0.426	0.620	0.581

Demo

Q&A

Image Sources

1. Nigerian National Parks. https://upload.wikimedia.org/wikipedia/commons/thumb/4/43/Nigerian_National_Parks.svg/300px-Nigerian_National_Parks.svg.png
2. African Parks Logo. https://africanparks.org
3. World Wildlife Fund Logo. https://socialmovements.trinity.duke.edu/groups/world-wildlife-fund
4. El Tuparro National Park. https://www.google.com/maps/place/El+Tuparro+National+Park/@5.2687328,-68.6084527,15z/data=!4m2!3m1!1s0x0:0xed3699227828119d?sa=X&hl=en&ved=2ahUKEwis9d6W4KH5AhV_LEQIHSkaDpgQ_BJ6BAhdEAU
5. Corcovado National Park Image. https://costa-rica-guide.com/nature/national-parks/corcovado/
6. Contrastive Learning Image. https://www.v7labs.com/blog/contrastive-learning-guide

Additional Sources

1. Botswana Predator Conservation Trust (2022). Panthera pardus CSV custom export. Retrieved from African Carnivore Wildbook 2022-04-28.
2. Botswana Predator Conservation Trust (2022). Crocuta crocuta CSV custom export. Retrieved from African Carnivore Wildbook 2022-04-28.
3. Parham J, Crall J, Stewart C, Berger-Wolf T, Rubenstein DI. Animal population censusing at scale with citizen science and photographic identification. In AAAI Spring Symposium-Technical Report 2017 Jan

Archive

Challenges

- ...

Next Steps

- ...

End-To-End Deliverable Checklist

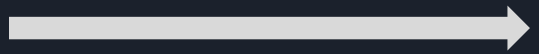
- ✅ Acquire Datasets and EDA.
- ✅ Setup our training environments in Databricks.
- ✅ Prototype of Web UI.
- ✅ Transformation Layer between Object Detection to Individual Detection. - Script that crops image to bounding box and saved into s3 location
- ✅ Object Detection Model
- ✅ Individual Detection Model
- 🚧 Retraining Integrated Pipeline
- ⏯️ Finalize Models
- ⏯️ Finalize Web UI
- ⏯️ Profit $$

Annotations

Inputs:

- Minimum X (Float type)
- Minimum Y (Float Type)
- Maximum X (Float Type)
- Maximum Y (Float Type)
- Original Height (Float Type)
- Original Width (Float Type)

Passed to BoundingBox Class that converts Yolov5 predictions to Coco format



Outputs:

- X = ((Minimum X / 640) * Original Width) + (Original Width / 2)
- Y = ((Minimum Y / 640) * Original Height) - (Original Height / 2)
- Width = ((Minimum X / 640) * Original Width)
- - ((Maximum X / 640) * Original Width)
- Height = ((Minimum Y / 640) * Original Height)
- - ((Maximum Y / 640) * Original Height)

YOLO Predictions: predict_bounding_boxes.py

Classes	Functions
InputImage: Reads tuple with file name, image location/height/width, and resized image location	read_images: Reads image location/height/width and resized image location
BoundingBox: Applies bounding box around animal in image	predict_bounding_boxes: Places bounding box on animal and makes Yolov5 predictions
	crop_and_upload: Crops images and uploads to s3 bucket
YolovPrediction: Applies Yolov5 predictions	annotate_and_upload: Makes bounding box annotations of image and uploads to s3 bucket
	yolov2coco: Converts Yolov5 predictions to Coco format, scaled to input image size

Individual Predictions: predict_individual.py

Classes	Functions
IndividualPrediction: Reads tuple with cropped file name, individual label, and individual name	group_yolov_predictions_by_species: Appends predictions per species to results set
	predict_individuals_from_yolov_predictions: Loads pre-trained embeddings from previously trained model backbone
NoneClassifier: Reads tuple with images that didn't pass through YOLOv5 prediction process	predict_individuals_from_species: Converts each species classifier and labels into 'RGB' and applies embeddings
	images_to_embeddings: Transforms and normalizes input images using ImageNet values

Retrain Embeddings: retrain_embeddings.py

Classes	Functions
EmbeddingImageDataSet: Retrieves and transforms randomly-selected S3 training images	embedding_num_sample: Takes local file path of uploaded images and returns number of prior training images to sample from S3
LocalImageDataset: Retrieves and maps custom images with labels, then resizes to required dimensions for Resnet18	main: Takes uploaded images, s3 resource object, and s3 bucket to be referenced, then returns an updated SimCLR model backbone for feature extraction and projection head
SimCLRModel: Version of SimCLR model for embedding re-training	

Retrain Classifiers: retrain_classifier.py

Classes	Functions
SpeciesOldTrainDataset: Retrieves and transforms all species' S3 training images	generate_embeddings: Takes pretrained Resnet 18 backbone and dataloader object for which to return embeddings and labels, then returns numpy array of embeddings and labels
LocalImageClassifierDataset: Custom dataset for locally saved images to be used for training	full_classifier_retrain: Takes s3 resource, s3 bucket, and species name for classifier retraining, then returns fitted model saved to a local folder

YOLOv5 Object Detection

1. Data Pre-Processing (Yolov5 Compatible)

2. Model - Training

3. Inference

- yolov5/data/

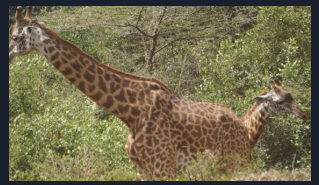
- train/
- val/
- test/

- yolov5/config/

- yolov5l.yaml


```
{
  “xmin”: 26,
  “ymin”: 59.8,
  “xmax”: 513,
  “Ymax”: 620
}
```


weights.py



End-to-End Process

End Users

Front End UI

YOLO Predictions

BBox Predictions -> Cropped Images -> Annotated Images -> **List of Tuples with Original File Name, Cropped Image Name, Annotated Image Name, BBox Prediction for each Image BBox**

Individual Predictions

YOLO Prediction Results -> Grouped by Species -> Grouped Species Processed by Respective Species Model -> Non-BBox Predictions Passed to Non Classifier -> **List of Tuples with Cropped Image Name and Predicted Cropped Image Label**

List of Annotations

Joins both YOLO Predictions and Individual Predictions -> Sorts and Combines each List of Cropped Image Name -> **Produces and Returns List of Annotations back to User**

Triggers Retraining Process If User Wants To Make Corrections

Retrained Embeddings

Retrieves and Transforms Random Images -> Retrains on Saved Images -> SimCLR Model for Embedded Retraining -> Updates SimCLR Model Backbone

Retrained Classifiers

Retrieves and Transforms All Species' Images -> Retrains on Locally Saved Images -> Generates Embeddings -> Returns Fitted Model

Retrained YOLOv5

Trains YOLOv5 Model on Training Parameters -> Makes Predictions on Inference Parameters -> Saves Inferenced Models with Confidence Scores into Folder

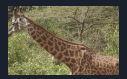
Solution Architecture

Databricks Workspace

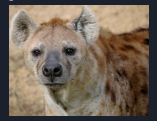
3.) SimCLR Individual Recognition - Confidence Prediction

Leopard: 85% Confidence

Giraffe A



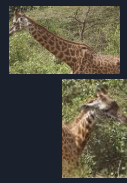
Hyena: 90% Confidence



Giraffe B

Shared Notebooks

2.) COCO Bounding Boxes - Image Cropping



1.) YOLO Object Detection - Bounding Boxes

S3 for storing uploaded images

S3 with yolov5 dependencies

User interacts with Web Interface

For results - API GET Request

For updates - API POST Request

For uploads - API POST Request

Hyena Image Results

Leopard Image Results

Giraffe Image Results

Unidentified Image Results

End-to-End Process

End Users

Front End UI

YOLO Predictions

BBox Predictions -> Cropped Images -> Annotated Images -> **List of Tuples with Original File Name, Cropped Image Name, Annotated Image Name, BBox Prediction for each Image BBox**

Individual Predictions

YOLO Prediction Results -> Grouped by Species -> Grouped Species Processed by Respective Species Model -> Non-BBox Predictions Passed to Non Classifier -> **List of Tuples with Cropped Image Name and Predicted Cropped Image Label**

List of Annotations

Joins both YOLO Predictions and Individual Predictions -> Sorts and Combines each List of Cropped Image Name -> **Produces and Returns List of Annotations back to User**

Triggers Retraining Process If User Wants To Make Corrections

Retrained Embeddings

Retrieves and Transforms Random Images -> Retrains on Saved Images -> SimCLR Model for Embedded Retraining -> Updates SimCLR Model Backbone

Retrained Classifiers

Retrieves and Transforms All Species' Images -> Retrains on Locally Saved Images -> Generates Embeddings -> Returns Fitted Model

Retrained Yolov5

Takes object classifications to extract multiple individuals for individual classification -> Returns retrained fine-tuned custom class predictions