Sea Turtle Conservation AI Project

Final Project

EEL3872

Jay Rosen

Abstract

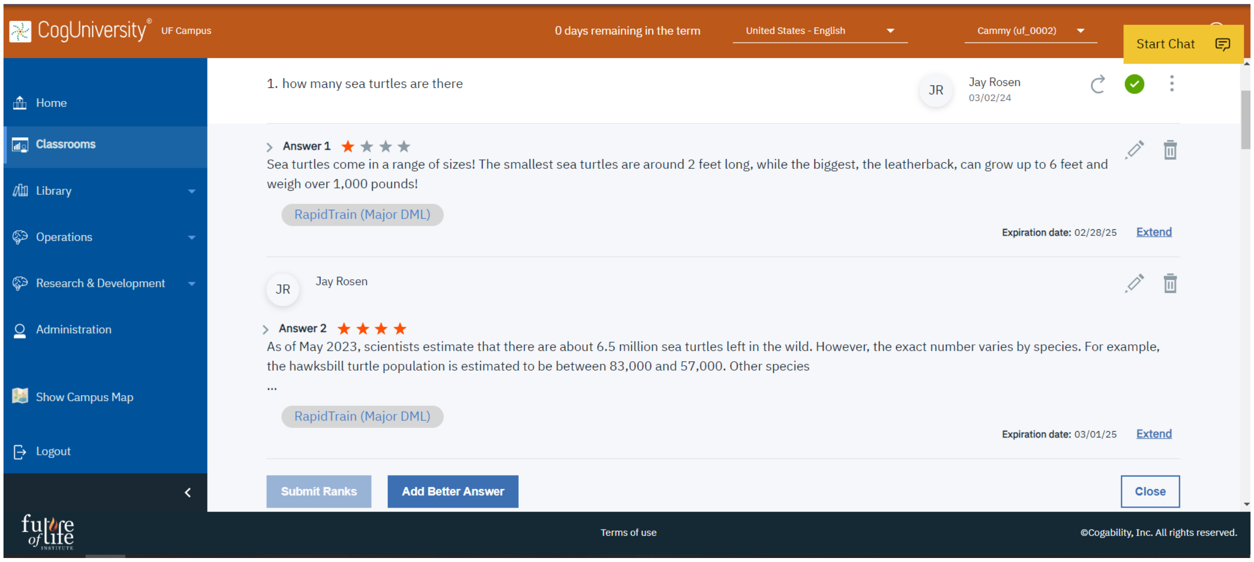
This project details the creation and application of "Cammy," an AI chatbot, across a series of experiments aimed at enhancing sea turtle conservation efforts. Initiated as part of an educational endeavor to bridge the knowledge gap regarding sea turtle conservation, Cammy was developed to interact with the public, providing essential information and fostering engagement. The first phase of Cammy's training involved equipping the chatbot with the ability to answer common questions about sea turtles, utilizing a dataset of 50 inquiries sourced from the community. Subsequent phases expanded Cammy's capabilities to include predicting the weight of sea turtles from visual measurements obtained through stereo camera imagery and identifying sea turtle species from user-submitted photos. These functionalities were integrated using advanced machine learning techniques, including AutoKeras Regression for weight prediction and image recognition algorithms for species identification. Throughout the project, challenges such as data biases, model overfitting, and the ethical use of copyrighted images were addressed. The culmination of these experiments demonstrated Cammy's potential as a powerful tool not only for public education and engagement but also as a supportive asset in scientific research and conservation strategies, significantly advancing the goals of the Sea Turtle Conservation Team. This initiative highlights how AI can be leveraged to make a meaningful impact on wildlife conservation, particularly by enhancing public awareness and participation in preservation efforts.

Introduction

Sea turtles, which have thrived for millions of years, are now facing unprecedented threats from habitat destruction, climate change, and human interactions. While awareness and protective measures are increasing, the gap between scientific knowledge and public engagement remains significant. This gap hinders effective conservation efforts, as public cooperation and understanding are crucial for the sustainability of these initiatives. To address these challenges, we undertook a project aimed at leveraging artificial intelligence to enhance sea turtle conservation efforts. This project involved the creation of "Cammy," an advanced AI-driven chatbot designed to serve as a bridge connecting the public with scientific knowledge and conservation practices. Cammy was developed at CogUniversity to fulfill three main functions: educating the public through interactive dialogue, predicting the weight of sea turtles from photographic data, and identifying sea turtle species from images submitted by users. Each phase was aimed at equipping Cammy with the necessary tools to assist in the conservation efforts spearheaded by Dr. Harris and his team in Florida. The overarching goal of this project was not only to foster greater public engagement and education regarding sea turtle conservation but also to explore the potential of AI in transforming wildlife conservation strategies into more interactive, informative, and engaging practices. Through Cammy, we seek to demonstrate how AI can effectively contribute to the conservation of endangered species and promote a more informed public, thereby enhancing the collective efforts to safeguard our marine biodiversity.

Problem Statement

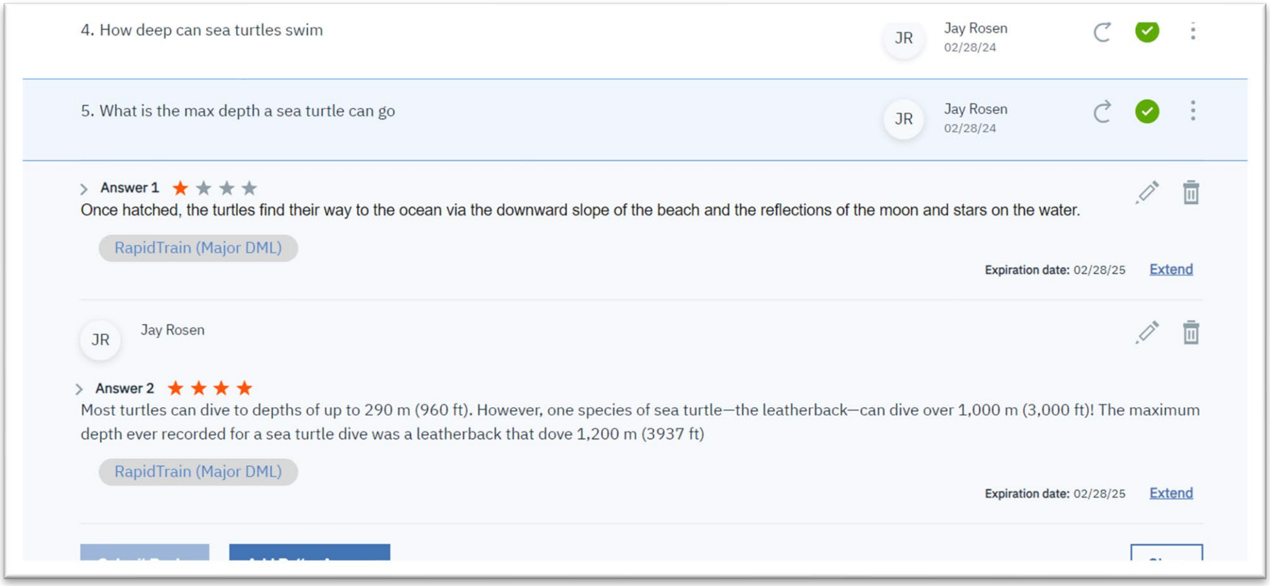
Despite ongoing conservation efforts, Sea Turtle populations continue to decline due to human activities affecting the marine ecosystem. One of the primary barriers to effective sea turtle conservation is the public's lack of understanding and awareness of how their actions impact these creatures. Traditional methods of environmental education and public engagement have not fully succeeded in bridging the gap between scientific knowledge and community practices. Moreover, data collection for research purposes, such as measuring and tracking sea turtles, often involves invasive techniques that can further stress the animals. This project addresses the need for innovative, non-invasive methods to engage the public and gather data without harming the animals. The development of "Cammy," an AI-driven chatbot, is proposed to fulfill this need by providing a platform for education and data analysis.



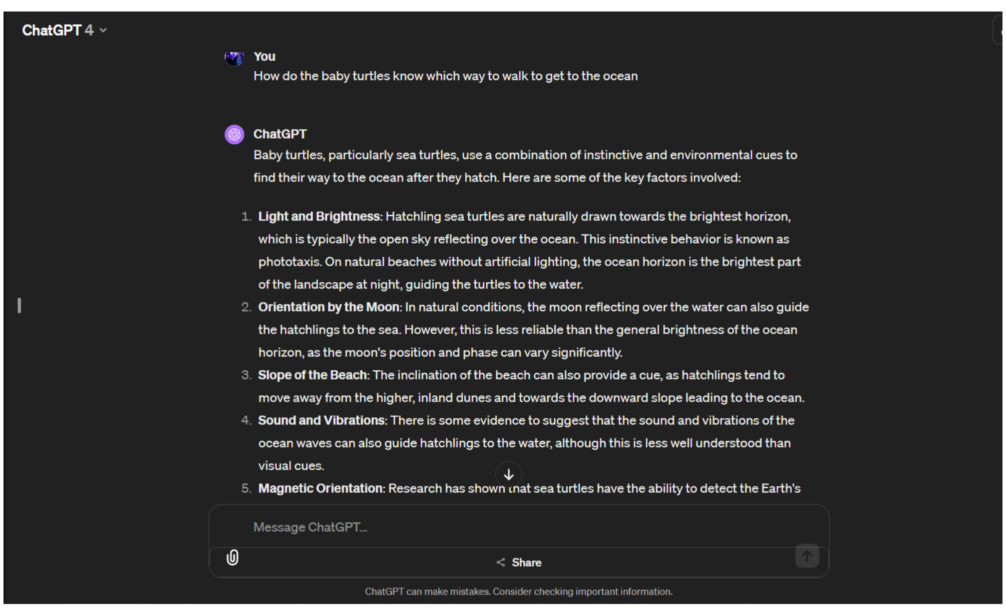
Creating the Knowledge database of questions and answers at CogUniveristy.

<u>Experiment 1</u>

To start, we gathered a collection of 50 questions from the community about Sea Turtles, such as how long they live, how large can they be, and preservation questions regarding how to observe them safely. As each question was answered, the CogBot AI would be trained using a ranking system. As more questions were answered over time, previous answers would be called upon from the AI's memory and Natural Language Processing deciphering the question. The answers that Cammy would provide would be ranked again given on scale of 1 -5, given the correctness or incorrectness of the given answer to the context of the individual questions. Many of the questions were similar, so Cammy was able to answer correctly from the previous correct responses. Cammy would answer incorrectly when a question was too specific or not at all related to the original questions that Cammy was trained on. Most likely it is individual words or phrases that the AI is looking up to quickly make its response. If a question is mistyped or words are misspelled, the chatbot has less accuracy at answering the question properly, which also suggest its unable to process bad data into cleaned data before making its predicted answer.

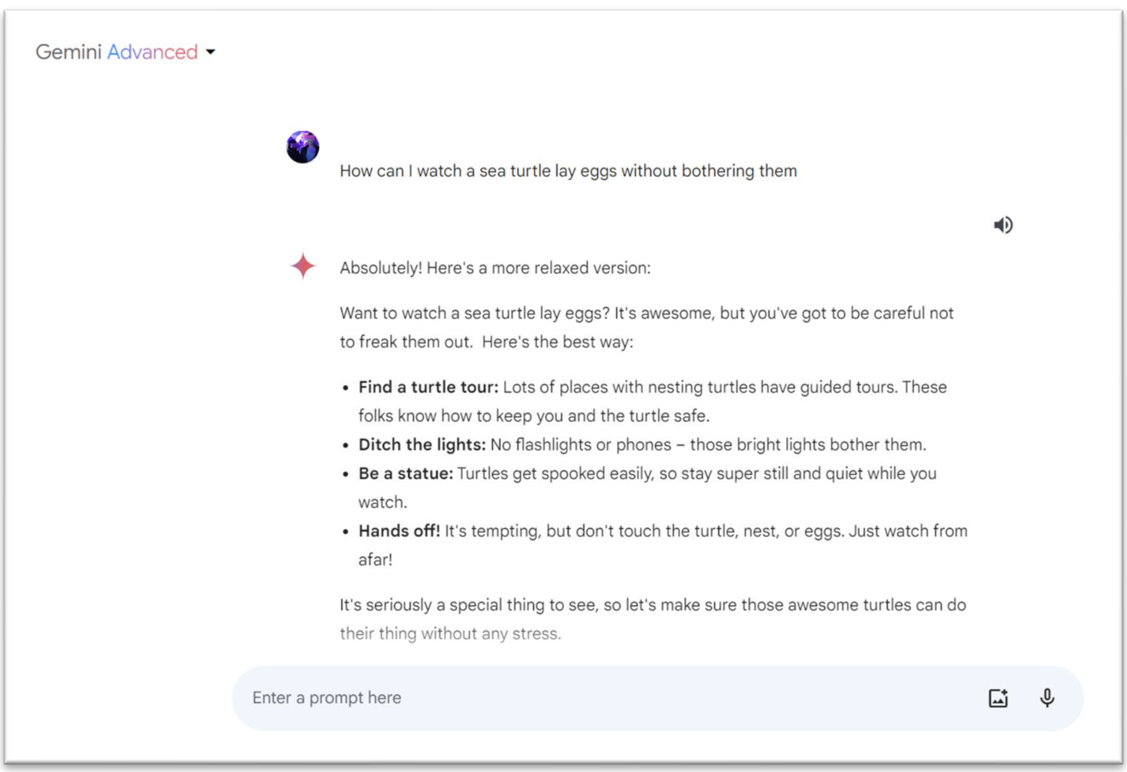


For research methods, Google search engine was used to look up these questions. The top search results would come from reputable sources like National Ocean and Atmospheric Administration (NOAA), National Geographic, or See Turtles Organization. The information on these websites was very similar since its factual, but the wording on these websites is slightly different, with NOAA having a more professional and scientific tone, while National Geographic and See Turtles Organization was simpler and easier to understand.



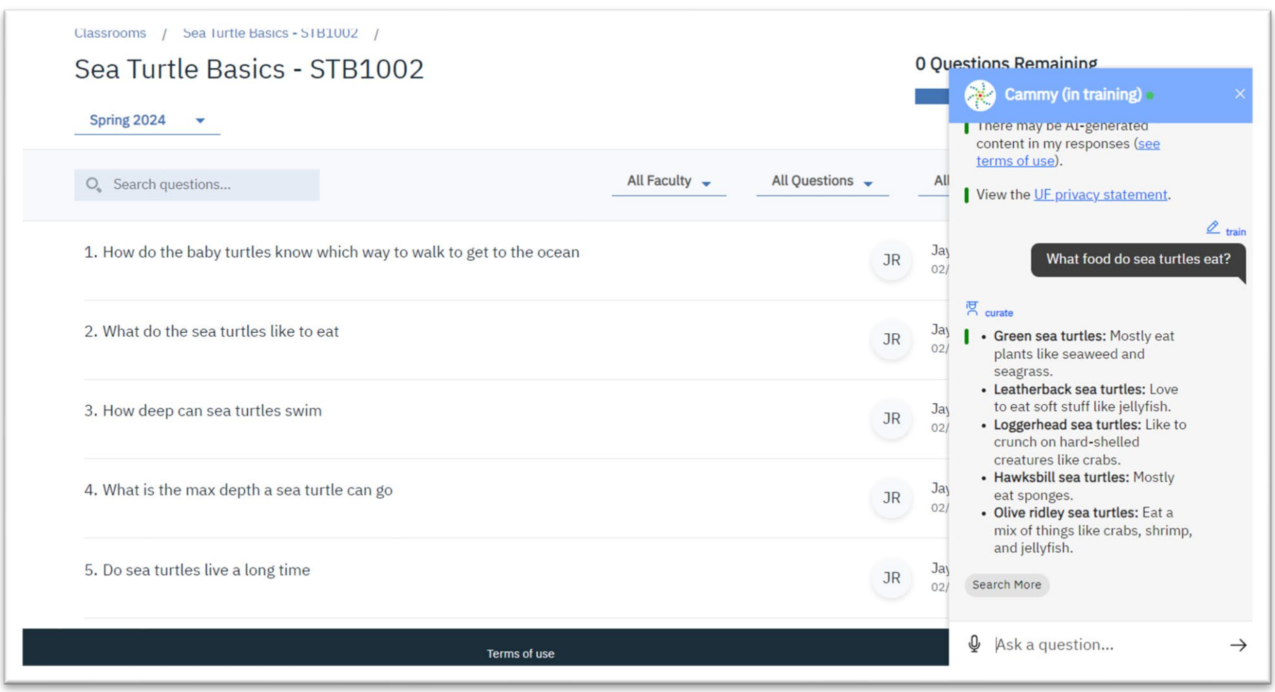
ChatGPT was used to answer some of the questions.

Additionally, newer tools such as ChatGPT and Google Gemini AI chatbots were utilized to answer these questions. It was interesting to compare how the 2 most popular AI chatbots would answer the questions, and I would even share the different responses with these 2 chatbots so they can understand how the other LLM is producing an answer. It was like the 2 chatbots were having a conversation with each other, and I served as a medium to exchange the information. The answers given by the 2 chatbots were very similar in wording, which suggests that ChatGPT and Gemini were trained on the same information, or similar data sources. It did seem kind of odd to use AI generated answers to teach another generative AI chatbot, but this was the quickest way to answer some of the questions given that Gemini is now integrated into Google Search. I do see that in the future, this method of generative AI training other generative AI models could lead to degrading quality of answers without human review, or diversity of knowledge. The only bias I was able to recognize was more mentions of Leatherback Sea turtles, but this is possibly due to their larger size and longer lifespans compared to other species of sea turtles.



Google Gemini was used to answer some of the questions.

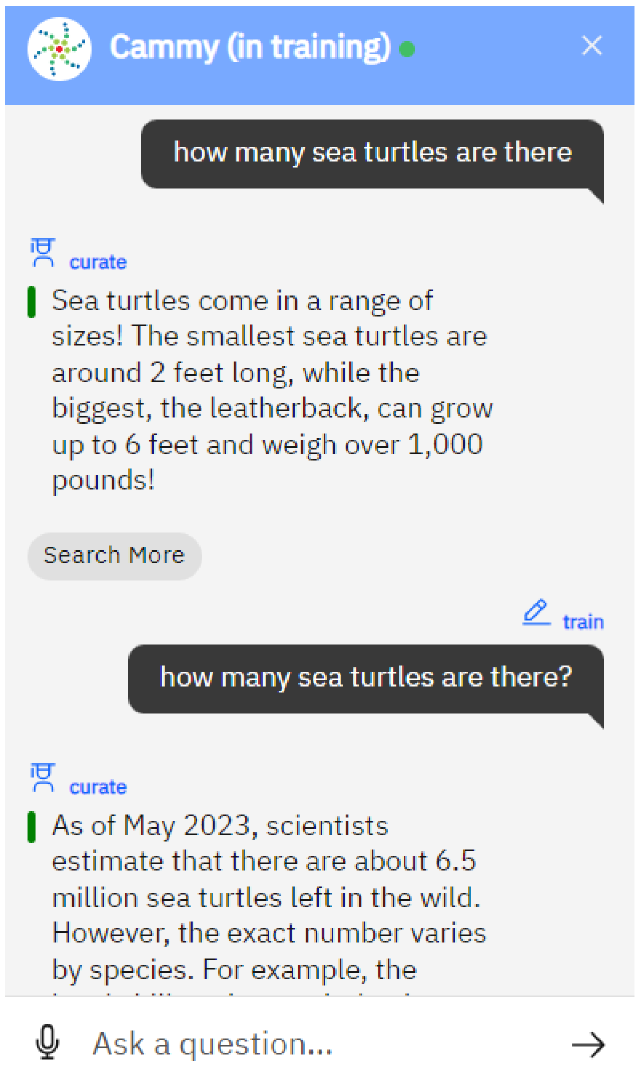
Cammy's tone of voice and vocabulary was exactly as it was trained based on the information from NOAA, National Geographic, ChatGPT, and Gemini. Cammy did not make up sentences it was not trained on, but just repeated answers verbatim it had already learned. CogBot has additional answer tooling to respond to questions in different manner such as telling jokes, giving directions, and technical tools suc has displaying information about the trained model or mention the user's name. For this training, I just used the Add Answer feature to supply an answer to the question. Add answer uses a TinyMCE text editor, which is standard for blogs and website content creation. There is capability to supply a media file like image or video with the Answer, but I only added text-based responses. Some questions such as "What type of food do turtles eat", would potentially have multiple responses since different turtles eat different foods. For these questions, I answered with bullet point responses, so that each of the sea turtles could have their foods as part of the answer.



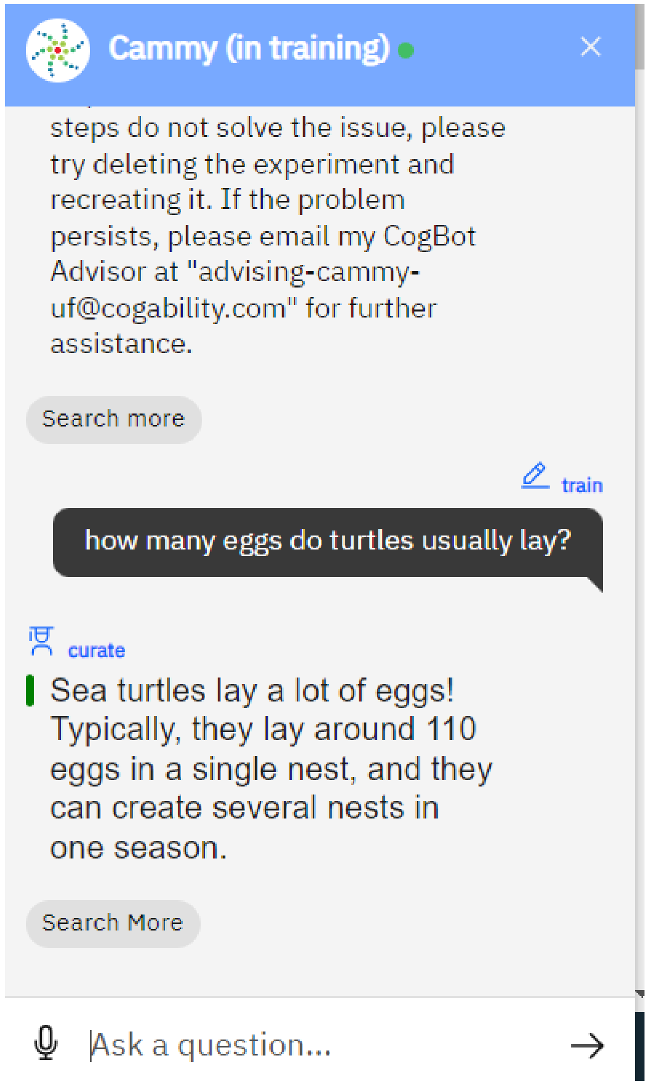
Cammy chatbot answering a question with bullet point answers.

When using the chatbot, there is option to Search More, which opens a relevant website in a new browser tab. Unfortunately, the Search More buttons kept going to CogBot website instead of a website about sea turtles. It seems this feature would be to see the citied data source but is currently not used like that in this training session. Another feature that did not seem to work was the microphone button, which I assumed was going to annotate my speech to text, but clicking the microphone button did not appear to do anything.

When a question is asked that has not been asked before, such as “How many types of sea turtles are there?” or “What countries do sea turtles habitat”, it would give an incorrect answer, but the UI does allow this new question to be added to the database with “Train” button, increasing the AI’s knowledge on the subject. After adding this question to the database, providing an answer, and ranking the suggested answers, the chatbot was able to answer the question shortly after.



Cammy answering questions incorrectly.



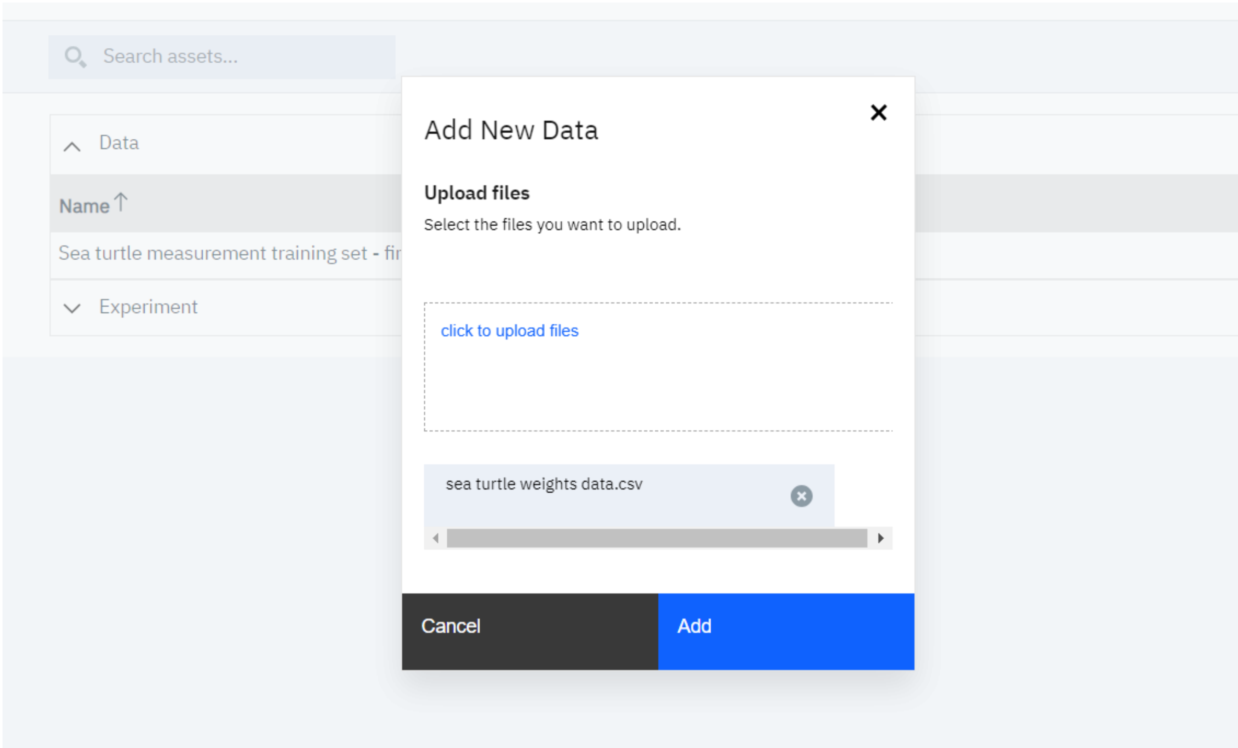
Cammy answering questions correctly.

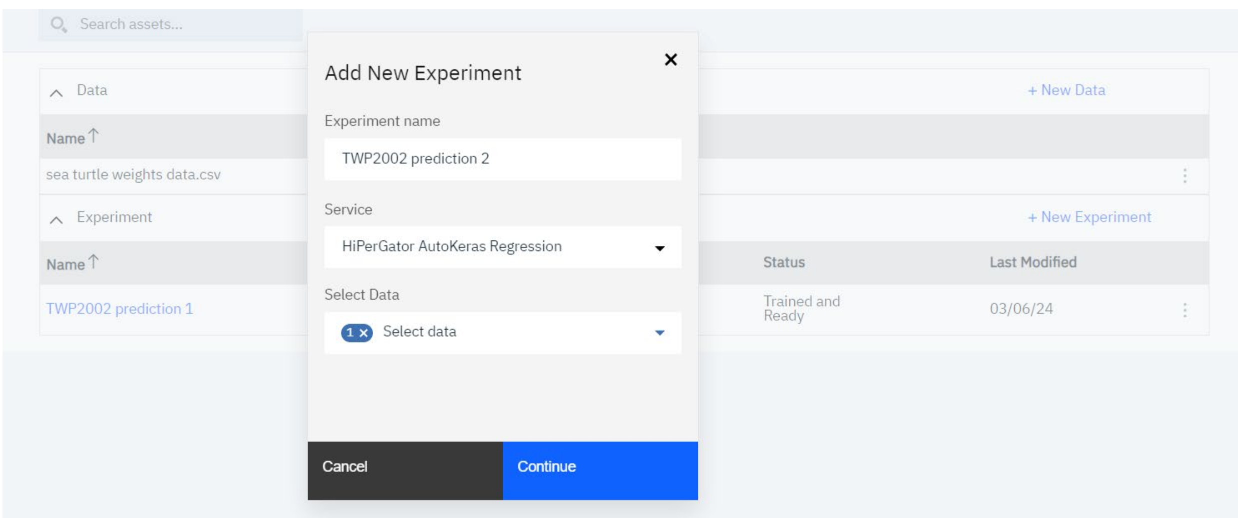
Experiment 2 – AutoKeras Regression

For this study, we analyzed a dataset of 5700 sea turtles' measurements, with datapoints for width(cm), length (cm), weight (kg), and species of sea turtles (green, loggerhead, hawksbill, and ridley). The csv file was uploaded to CogBot University for training Cammy, the AI Chatbot, using HiPerGator CPU. The AutoKeras Regression algorithm was used for training, which is automated machine learning built on top of TensorFlow library. AutoKeras Regression was setup with the prediction target set to weight (kg), and uses the characteristics of width, length, and species for calculating the predicted weight.

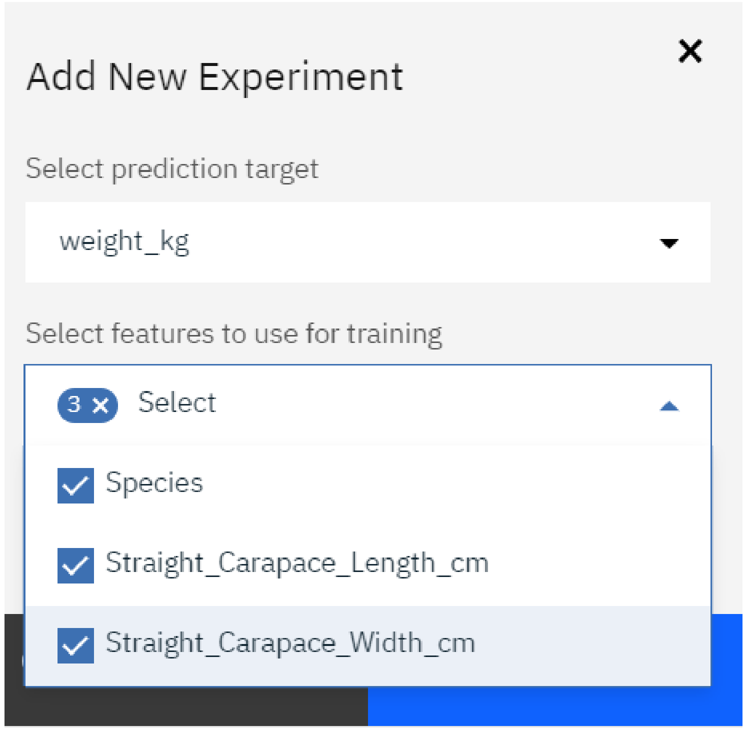
1	Species	Straight_Carapace_Length_cm	Straight_Carapace_Width_cm	weight_kg
2	loggerhead	59.4	48.9	32.4
3	loggerhead	4.41	3.33	17.3
4	loggerhead	61.1	52.4	36.4
5	green	33.4	26.4	4.5
6	green	36.2	30	5.1
7	loggerhead	54.4	44.9	24.4
8	ridley	30.4	28	4
9	loggerhead	72.5	56.7	28.9
10	loggerhead	66.6	54.7	45
11	green	40.5	31.7	7.2
12	loggerhead	64.2	51.3	37.2
13	loggerhead	63.2	50.2	32.8
14	loggerhead	4.88	3.93	24.3
15	ridley	29.9	27.9	3.7
16	loggerhead	63.8	52.9	38

The csv dataset of Turtle dimensions





Uploading the csv data and creating experiment on HiPerGator



Selecting which specific data features to use for training

The training time took about 1 minute, and the output files showed the AutoKeras Regression was trained for 20 epochs. The 1st epoch had a high loss value of 22.7916, and the 2nd epoch had loss value of 5.54, which demonstrated substantial reduction in loss in this 1st cycle. Epoch 3 had loss value of 3.27, which continued to show learning with loss cut in half. From Epoch 4 through 20, the loss and loss validation showed stabilization, staying in the 2.5 – 1.93 range. The lower loss demonstrated the model could recognize pattern of species and dimension correlating to weight. The output summary showed that the model was trained on 1,224 total parameters, and only had 7 non-trainable parameters.

Name ↑
Sea turtle measurement training set - final.csv

Experiment + New Experiment

Name ↑	Service	Status	Last Modified
TWP2002 prediction 1	HiPerGator_AutoKeras_Regression	Trained and Ready	03/06/24

Model Id
650cdf01-ded1-4922-8bd7-90b689854a63

Target field
weight_kg

Train fields
Straight_Carapace_Length_cm, Species, Straight_Carapace_Width_cm

Data
1. Sea turtle measurement training set - final.csv

Output
1. 650cdf01-ded1-4922-8bd7-90b689854a63.log
2. summary.txt

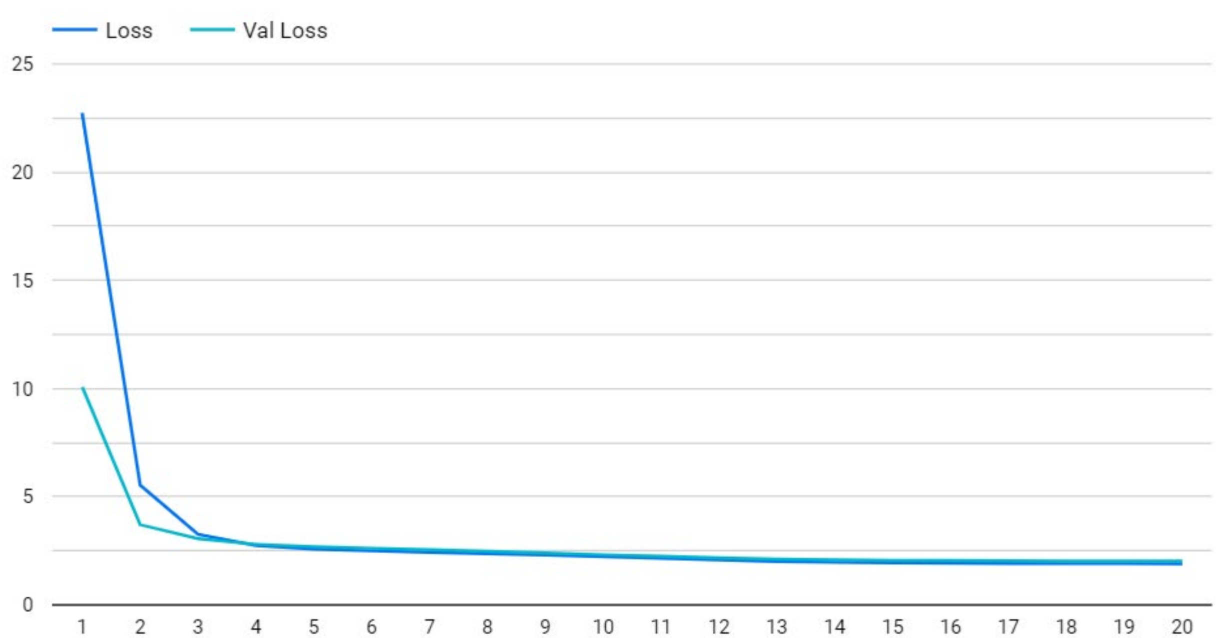
Experiment Output screen including Model ID

```
Model: "model"
_________________________________________________________________
Layer (type)                Output Shape              Param #
=================================================================
input_1 (InputLayer)        [(None, 3)]               0
_________________________________________________________________
multi_category_encoding (Mul (None, 3)                0
_________________________________________________________________
normalization (Normalization (None, 3)                7
_________________________________________________________________
dense (Dense)               (None, 32)                128
_________________________________________________________________
re_lu (ReLU)                (None, 32)                0
_________________________________________________________________
dense_1 (Dense)             (None, 32)                1056
_________________________________________________________________
re_lu_1 (ReLU)              (None, 32)                0
_________________________________________________________________
regression_head_1 (Dense)   (None, 1)                 33
=================================================================
Total params: 1,224
Trainable params: 1,217
Non-trainable params: 7
_________________________________________________________________
```

Summary of the AutoKeras training

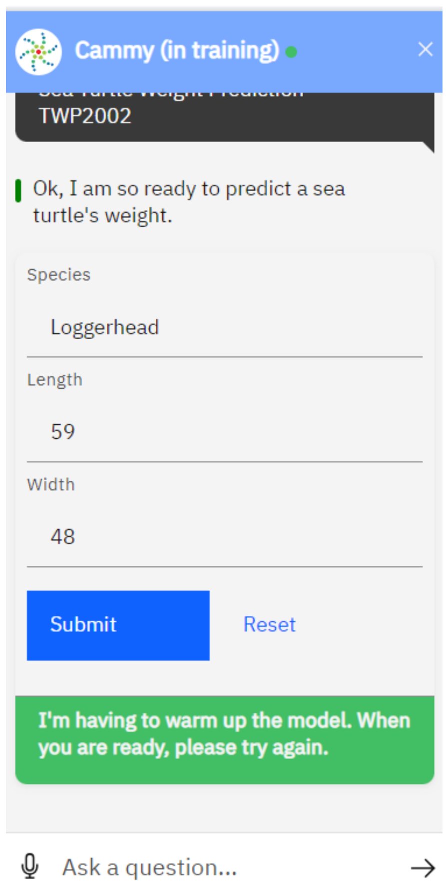
Epoch	Loss	Val Loss
1	22.73	10.07
2	5.54	3.71
3	3.27	3.06
4	2.75	2.8
5	2.58	2.69
6	2.5	2.62
7	2.43	2.55
8	2.37	2.48
9	2.31	2.4
10	2.24	2.32
11	2.16	2.25
12	2.08	2.17
13	2.01	2.12
14	1.97	2.09
15	1.94	2.05
16	1.93	2.05
17	1.92	2.04
18	1.91	2.02
19	1.91	2.03
20	1.9	2.02

Summary of the training by epoch



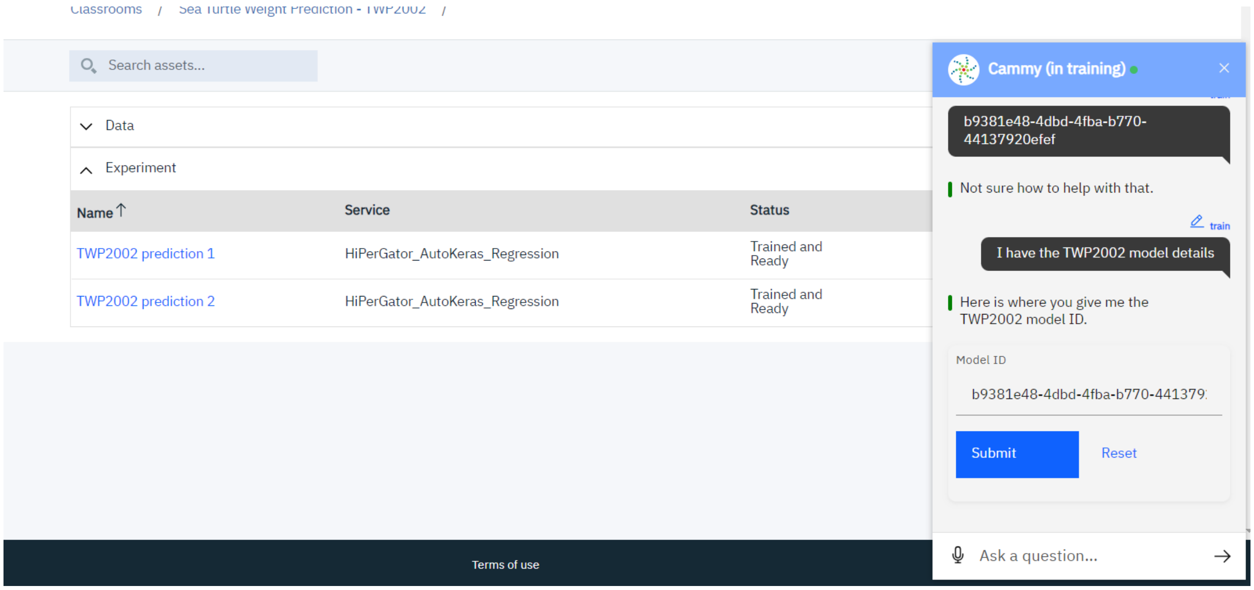
AutoKeras training over 20 epochs

The trained model had an ID 650cdf01-ded1-4922-8bd7-90b689854a63, and was given to Cammy chatbot for it to reference. Once Cammy had the model loaded to memory, it made a refence to The Matrix film, declaring Cammy knows Jiujitsu. Cammy then displayed an input form in the conversation window, to input species, width, and height. When trying this several times, I ran into issues, where Cammy would respond “I’m having to warm up the model, when you are ready, please try again.”. This happened over the span of a few days, so I tried to re-upload the data and performed the experiment again of training a new model with AutoKeras on HiPerGator, using same setup.



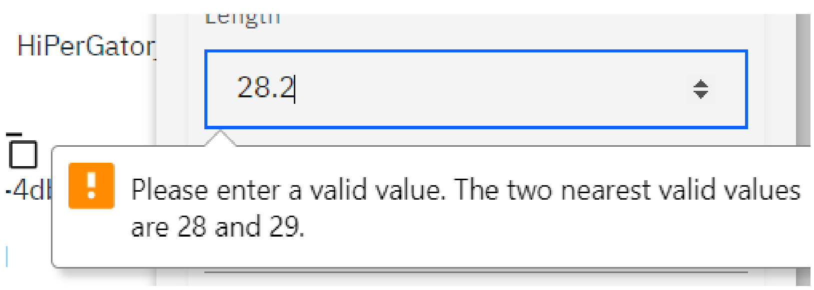
Updated UI with input fields for measurements

The 2nd trained model had ID b9381e48-4dbd-4fba-b770-44137920efef. To give Cammy the new model, I had to type “I have the TWP2002 model details”. This process of giving Cammy a new model was not intuitive, since the default behavior of starting the training session would auto-populate the “I have the TWP2002 model details” to the chatbot, and no other option to change model unless say that again, without instructions of the usable commands. Typing /help did not give set of phrases to say to reset the model.

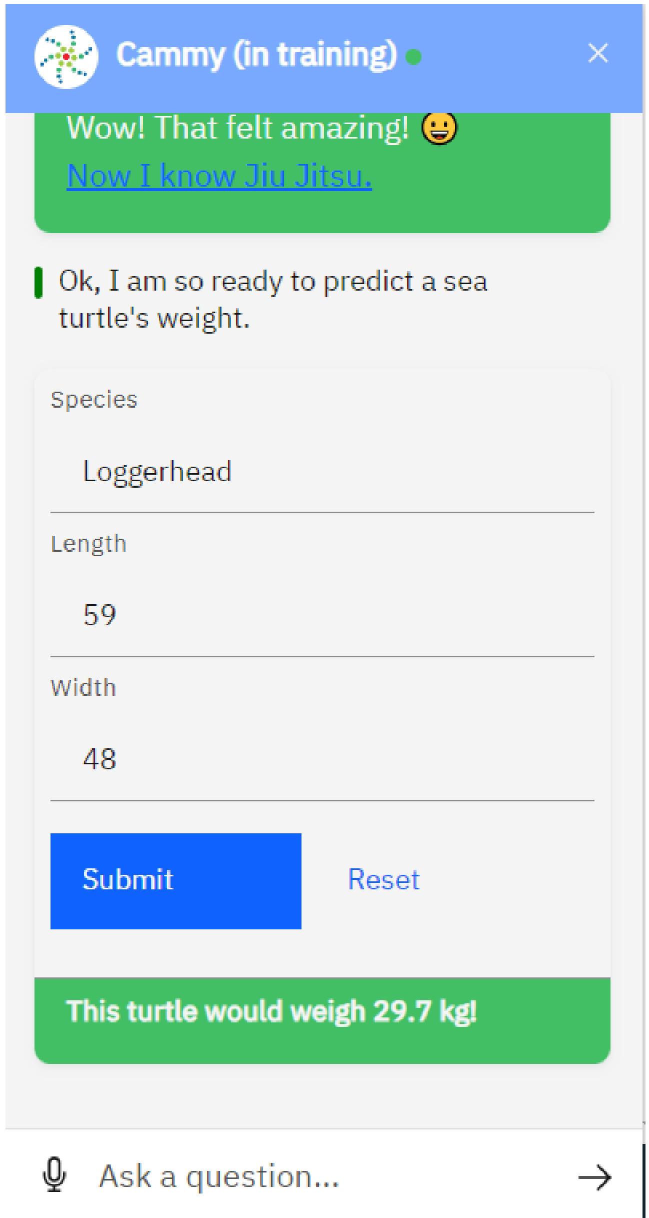


Two experiments created on HiPerGator due to errors, created different Model IDs

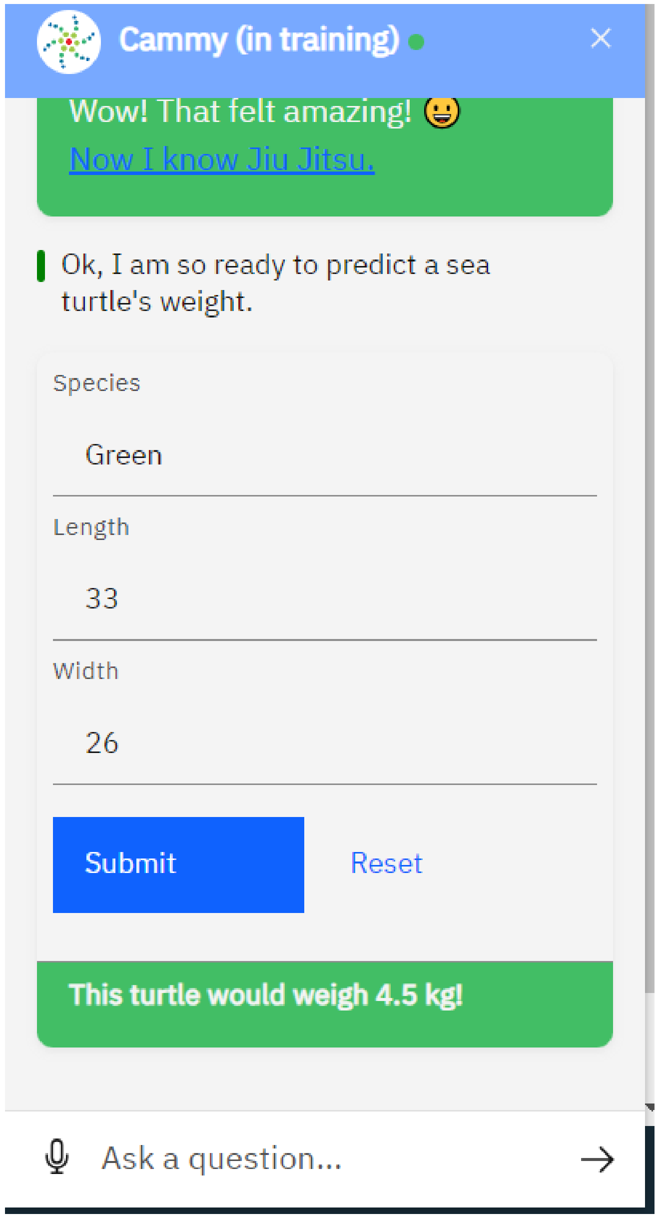
The 2nd model trained did work, and I was able to see predicted weights by entering dimensions and species. I referenced the original dataset, testing 10 different rows of data, or 2-3 different sizes of the 4 turtle species. The answers were close to the actual weight of the turtle being measured. For example, a loggerhead that is 40 x 40 cm would be predicted to have a weight of 13.2 kg. One of the issues I see with the Cammy input, is that it would only accept whole integer numbers, and would not allow floating decimal numbers for input, even though it would output floating decimal numbers for weight. The model was trained on floating decimals in the dataset, so it seemed odd to be unable to use decimal input. This could show loss of data and loss of precision for the experiment, however, does make it easier for the end-user interacting with chatbot by just entering simple numbers.



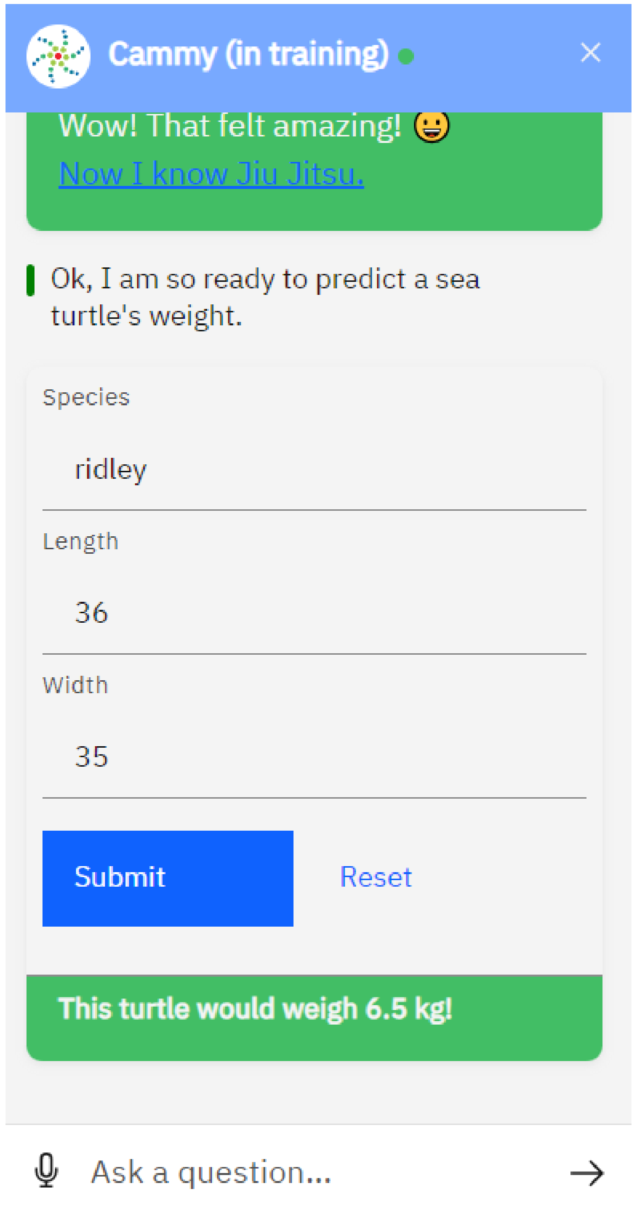
The input fields were only able to use whole numbers, not decimal numbers like the trained data.



Cammy was correct in the AutoKeras training



Cammy was correct in the AutoKeras training

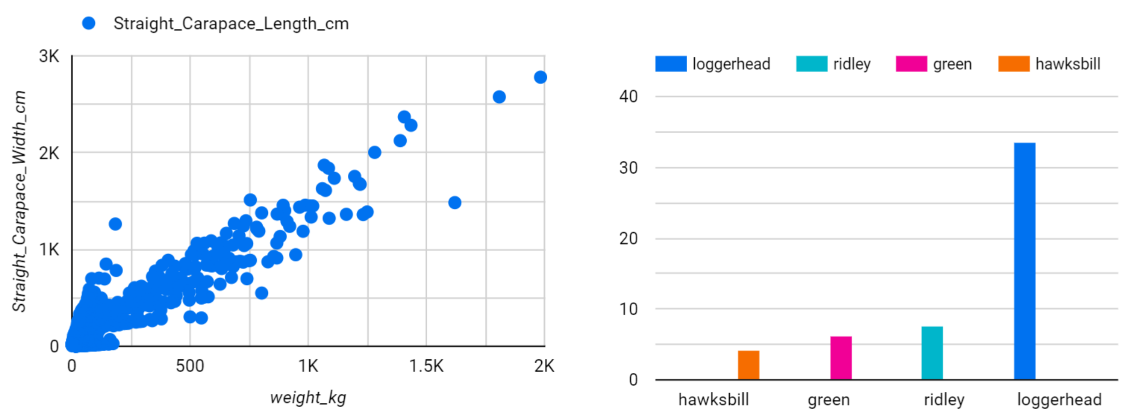


Cammy was correct in the AutoKeras training

1	Species	Straight_Carapace_Length_cm	Straight_Carapace_Width_cm	Predicted weight_kg (Cammy)
2	loggerhead	59	48	29.7
3	green	33	26	4.5
4	ridley	36	35	6.5
5	green	26	20	2.2
6	green	28	22	2.8
7	loggerhead	58	47	28.3
8	hawksbill	34	26	5
9	hawksbill	40	29	7.3
10	loggerhead	54	46	23.7
11	ridley	40	40	13.2

Chart of the predicted sea turtle weights with given input dimensions

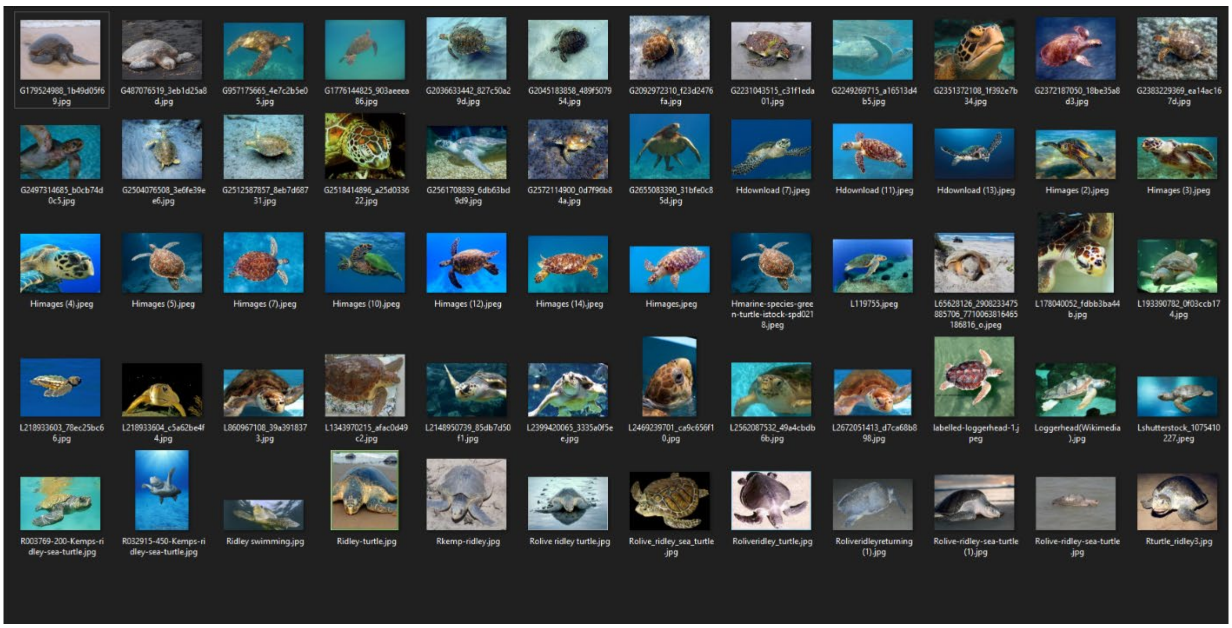
Google Looker Studio was used to chart the dataset of the 5700 turtles from the experiment. Scatter plot charts illustrated the correlation of dimensions and weight, and bar chart demonstrated the different sizes of species of turtle with Loggerheads appearing much larger than other species. Additional tables were added to chart the AutoKeras learning curve, and the experiment was logged as a csv of input integer dimensions and the predicted decimal weights for the 10 tests with Cammy.



Charts of the distribution of weights by sea turtle species

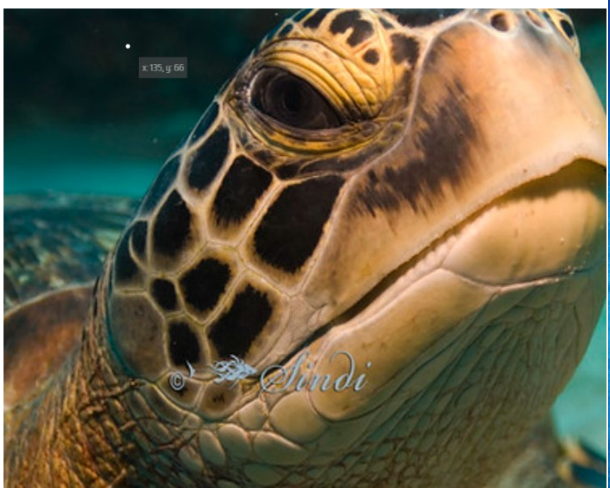
Experiment 3 – Image Recognition

For this experiment, we gathered a collection of 60 images of sea turtles that have been pulled from online, and trained Cammy AI chatbot to recognize the species of sea turtles in the images. The images were provided by Cog University as a zip archive, and on inspection of the images, seem to be the same images that appear in Google image search results for Sea turtles. The image file names were rather random, but the first letter of the image file name began with the species name to help us identify the turtle for labeling. The identifier letters were ‘G’ for Green Sea Turtle, ‘H’ for Hawkbill Sea Turtle, ‘L’ for Loggerhead Sea Turtle, and ‘R’ for Ridley Sea Turtle.



The 60-image dataset

It is to be noted that the images provided for this dataset show signs of copyright infringement, with watermarks like “Copyright SeaPics.com” and photographer’s names clearly visible in these photos, sometimes directly over the sea turtle itself. Copyright infringement is a legal and ethical concern, but this specific AI experiment was done for scientific educational purposes only, and not for commercial applications. The watermark text will interfere with the AI Training, as the model will learn to identify the watermark as a visual feature of certain sea turtles, which is inaccurate and adds undesirable noise to the refinement process. For LLMs like ChatGPT and AI Image Generators like MidJourney or DALL-E, this is a major problem for AI industry as a whole, as these AI models are being trained on copyright images and copyright works, without the original artist or author's expressed consent. It can be difficult to identify a copyright image if it doesn't contain a watermark or metadata embedded within the image, but it is also even more difficult to gather these professional underwater photographs of Sea Turtles to begin with. This is a dilemma that AI companies, regulators, and the community will have to figure out a way to purge out copyright works from these established AI models, or way to credit or compensate original creators.

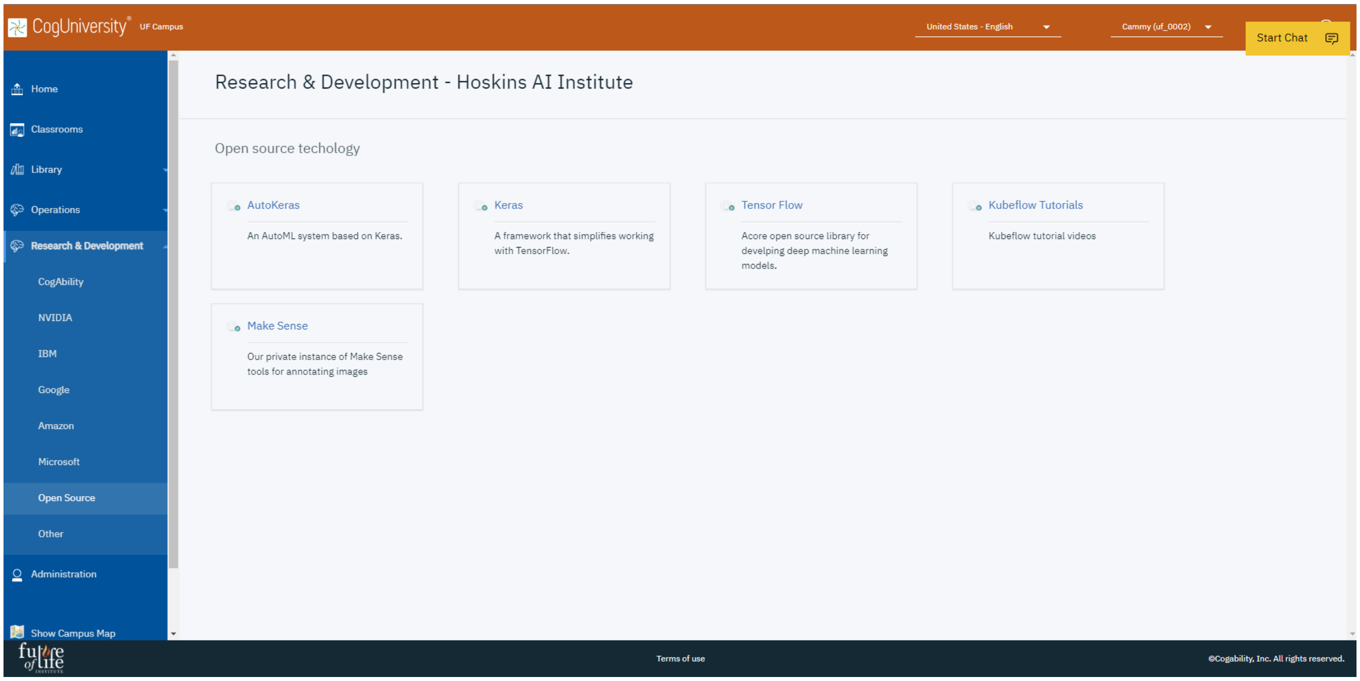


Images with copyright watermarks

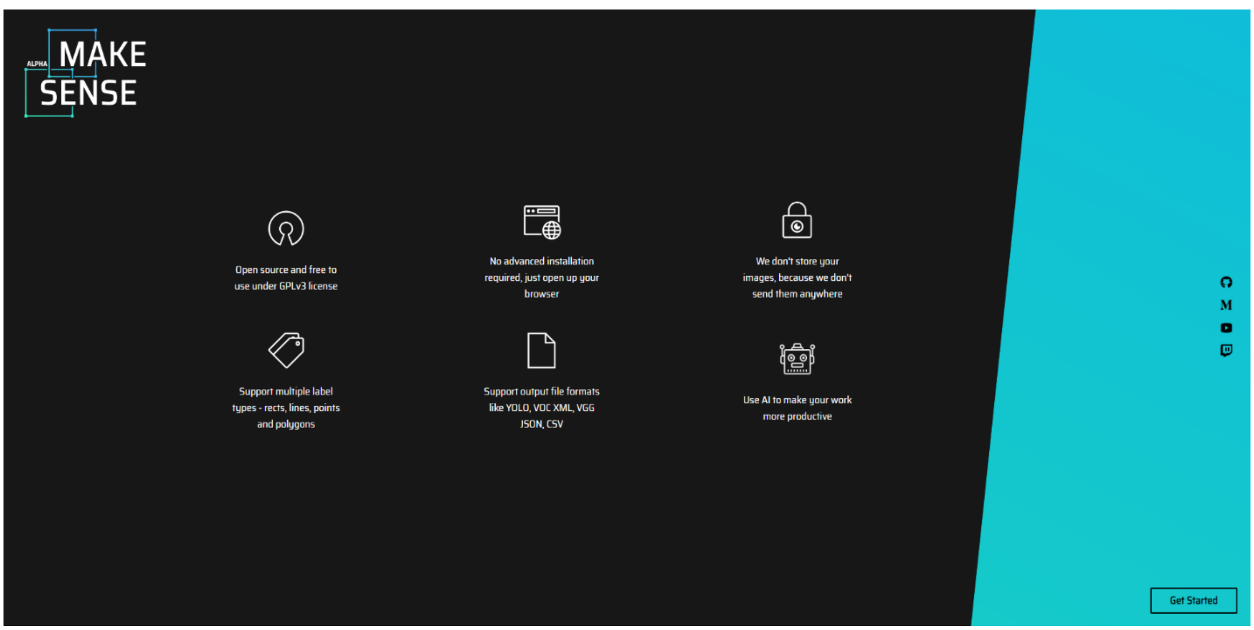
Another note about these specific images is they were all taken with professional underwater cameras and gear. These photos are beautiful and taken by professional photographers who have mastered underwater wildlife photography, which means our AI trained model will only know what professionally photographed sea turtles underwater look like, and not trained on wider range of photos such as smartphone photos taken by common person at the beach or at an aquarium. Only the Ridley Sea Turtle photos appear to be shot with a less professional camera, at lower resolution, and with flash enabled, which over-exposes the sea turtle subject. Besides 2 photos of Green Sea Turtles and 1 photo of Loggerhead Sea Turtles, all the Ridley Sea Turtle photos are captured on land, while all other photos in dataset were taken underwater. This bias in the dataset does train the AI model to recognize Riddley Sea Turtles with sandy colored backgrounds instead of aqua colored ocean backgrounds. The photos in general are from variety of angles and distances which is good, but also include close ups of the turtle's face which can skew the results as the close ups are cropped in and big. These closeups may train the AI model to think that these turtles are larger than they should be, and don't contain other features like shell or arms and legs.

Ridley Sea Turtle photos are lower resolution and taken on land.

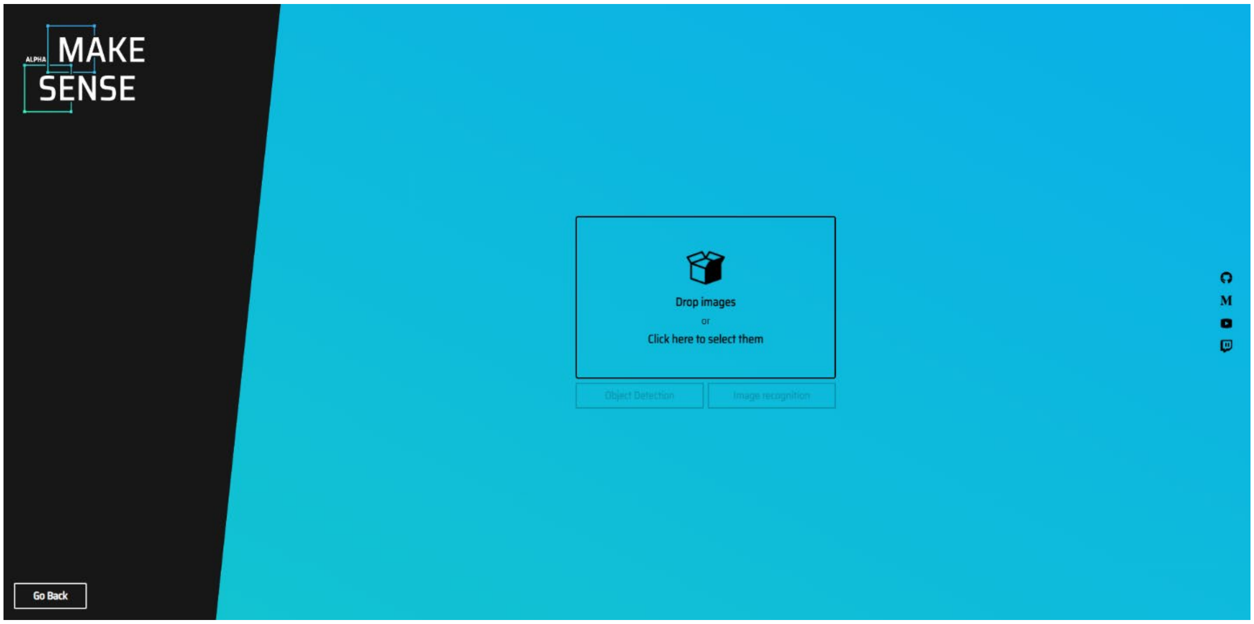
For labeling the images, we used Make Sense AI program which helped us associate words for the images. On the Make Sense AI website image upload screen, there were 2 options to choose from, Object Detection or Image Recognition. For this experiment, the image Recognition option was selected and the 60 images were uploaded. To start the labeling process, the Create Labels dialog screen prompted labels for their mages, and in this case the 4 species of Sea Turtles were each entered as a separate label. Once completed, each image was displayed in the lightbox gallery, and the right sidebar showed the 4 species as labels to associate for the image. Since the images had their filenames starting with the first letter of the species, it was relatively easy to identify the proper label of the Sea Turtle. If the filenames had not been previously named in this scheme, it would be more challenging to use our human brain to identify the Sea Turtle with knowledge of the Sea Turtle species visual features and logical judgment. Once all 60 images had been labeled, the Actions menu generated a 2 columns CSV file containing the images and labels. This CSV was uploaded to Google Looker Studio for simple table and chart analysis.



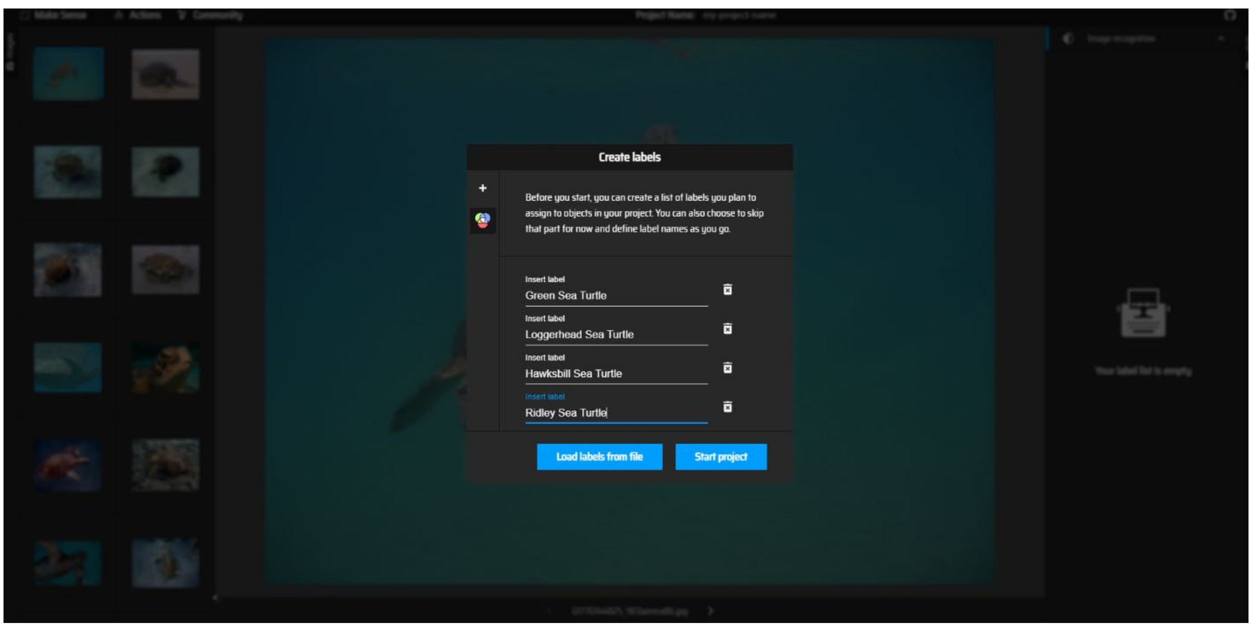
Make Sense AI was located in CogUniveristy > Research & Development > Open Source



Make Sense AI landing page.

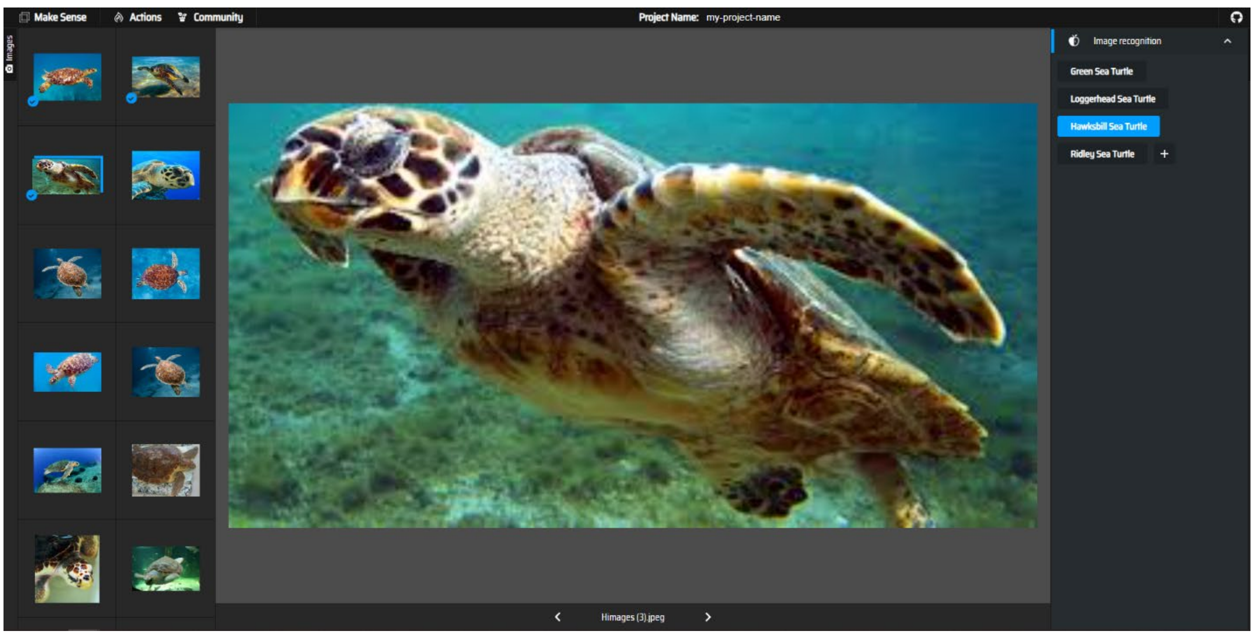


Make Sense Upload screen.



Creating the labels

Labeling a Green Sea Turtle



Labeling Hawksbill Sea Turtle

Labeling Loggerhead Sea Turtle

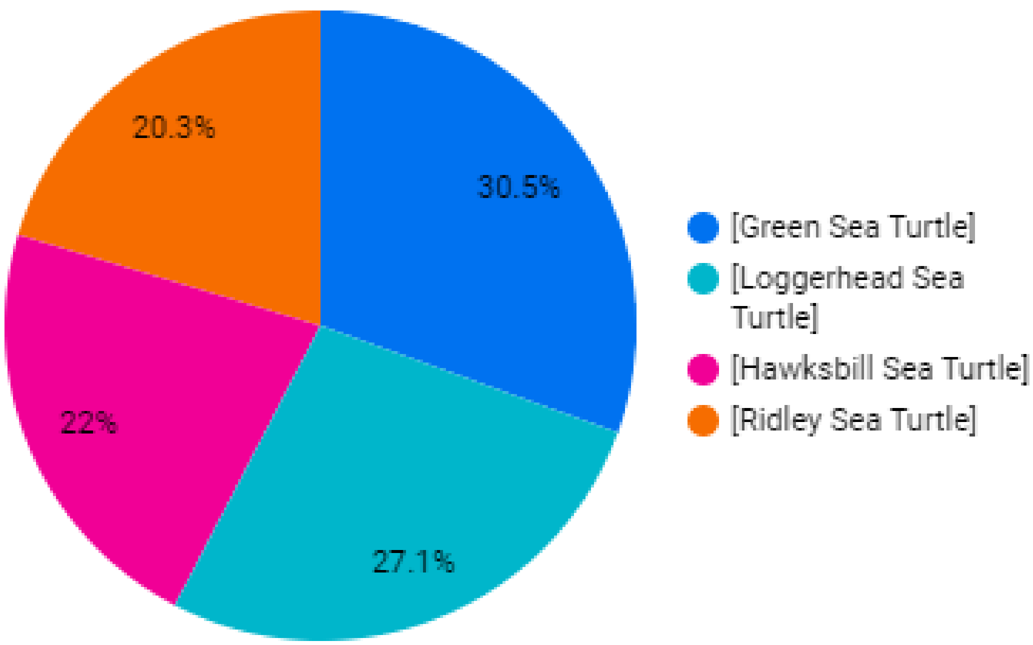
	A	B
1	G1776144825_903aeeea86.jpg	[Green Sea Turtle]
2	G179524988_1b49d05f69.jpg	[Green Sea Turtle]
3	G2036633442_827c50a29d.jpg	[Green Sea Turtle]
4	G2045183858_489f507954.jpg	[Green Sea Turtle]
5	G2092972310_f23d2476fa.jpg	[Green Sea Turtle]
6	G2231043515_c31f1eda01.jpg	[Green Sea Turtle]
7	G2249269715_a16513d4b5.jpg	[Green Sea Turtle]
8	G2351372108_1f392e7b34.jpg	[Green Sea Turtle]
9	G2372187050_18be35a8d3.jpg	[Green Sea Turtle]
10	G2383229369_ea14ac167d.jpg	[Green Sea Turtle]
11	G2497314685_b0cb74d0c5.jpg	[Green Sea Turtle]
12	G2504076508_3e6fe39ee6.jpg	[Green Sea Turtle]
13	G2512587857_8eb7d68731.jpg	[Green Sea Turtle]
14	G2518414896_a25d033622.jpg	[Green Sea Turtle]
15	G2561708839_6db63bd9d9.jpg	[Green Sea Turtle]
16	G2572114900_0d7f96b84a.jpg	[Green Sea Turtle]
17	G2655083390_31bfe0c85d.jpg	[Green Sea Turtle]
18	G487076519_3eb1d25a8d.jpg	[Green Sea Turtle]
19	G957175665_4e7c2b5e05.jpg	[Green Sea Turtle]
20	Hdownload (11).jpeg	[Hawksbill Sea Turtle]
21	Hdownload (13).jpeg	[Hawksbill Sea Turtle]
22	Hdownload (7).jpeg	[Hawksbill Sea Turtle]
23	Himages (10).jpeg	[Hawksbill Sea Turtle]
24	Himages (12).jpeg	[Hawksbill Sea Turtle]
25	Himages (14).jpeg	[Hawksbill Sea Turtle]
26	Himages (2).jpeg	[Hawksbill Sea Turtle]
27	Himages (3).jpeg	[Hawksbill Sea Turtle]
28	Himages (4).jpeg	[Hawksbill Sea Turtle]
29	Himages (5).jpeg	[Hawksbill Sea Turtle]
30	Himages (7).jpeg	[Hawksbill Sea Turtle]
31	Himages.jpeg	[Hawksbill Sea Turtle]
32	Hmarine-species-green-turtle-istock-spd0218.jpeg	[Hawksbill Sea Turtle]
33	L119755.jpeg	[Loggerhead Sea Turtle]
34	L1343970215_afac0d49c2.jpg	[Loggerhead Sea Turtle]
35	L178040052_fdbb3ba44b.jpg	[Loggerhead Sea Turtle]
36	L193390782_0f03ccb174.jpg	[Loggerhead Sea Turtle]
37	L2148950739_85db7d50f1.jpg	[Loggerhead Sea Turtle]
38	L218933603_78ec25bc66.jpg	[Loggerhead Sea Turtle]

labels_my-project-name_2024-03-

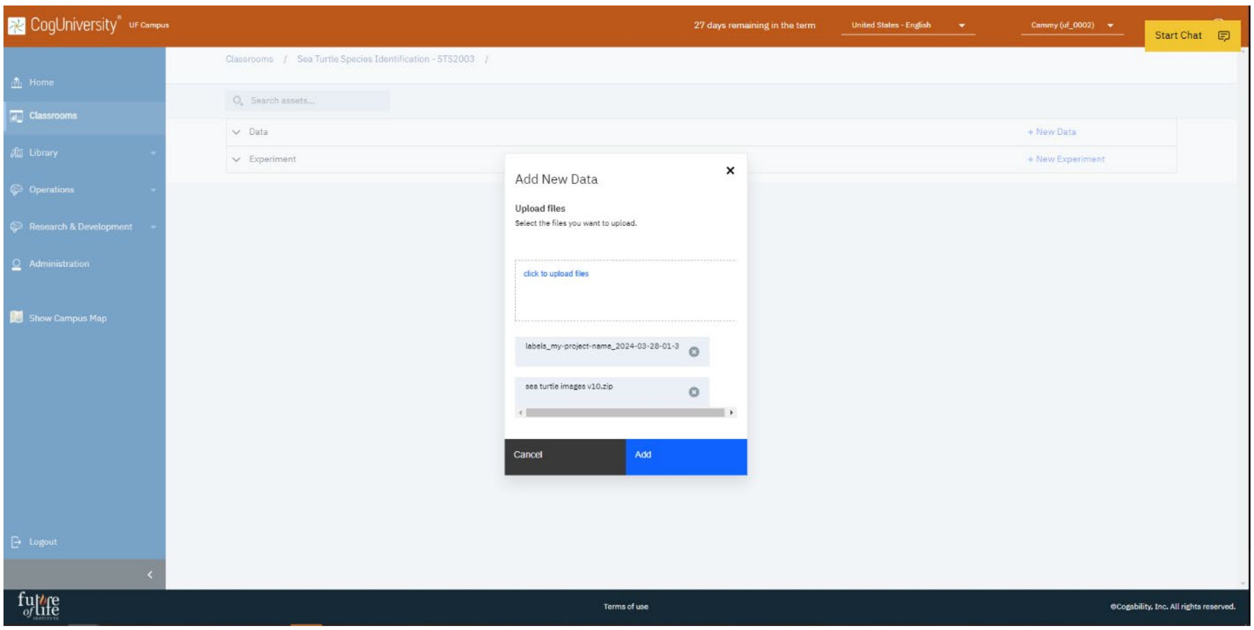
Ready Accessibility: Unavailable

The generated CSV of image and labels

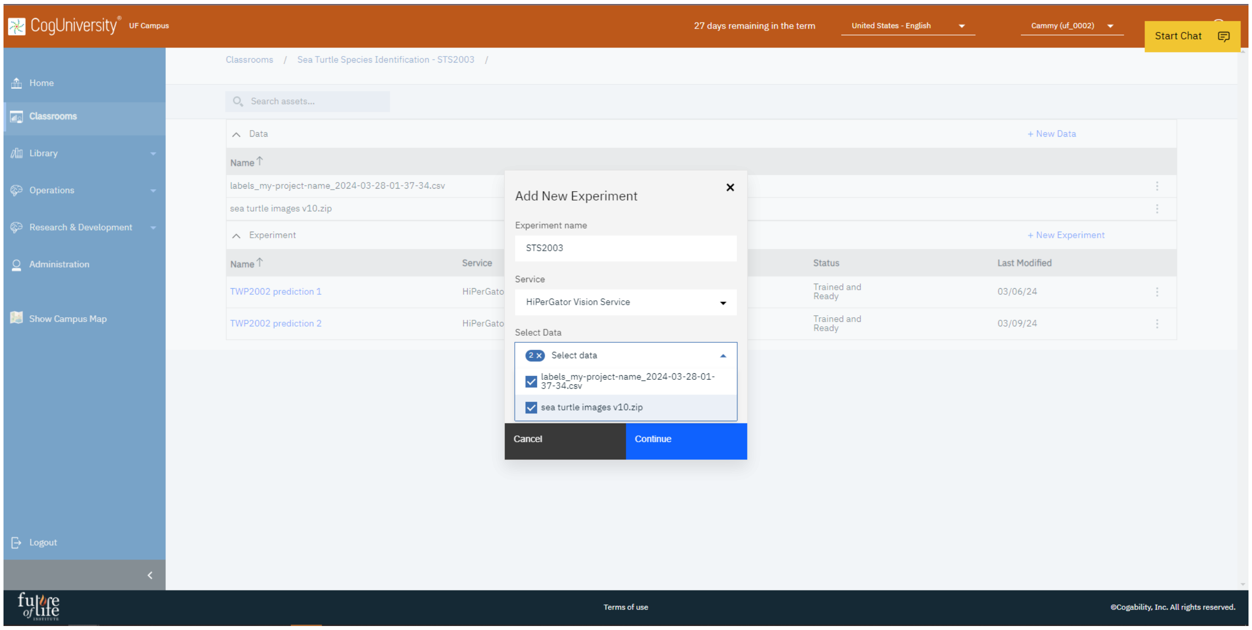
	[Green Sea Turtle]	Record Count ▾
1.	[Green Sea Turtle]	18
2.	[Loggerhead Sea Turtle]	16
3.	[Hawksbill Sea Turtle]	13
4.	[Ridley Sea Turtle]	12



Back at CogUniveirsty, the zip archive of images and CSV of labeled images were uploaded as Data for the Experiment. For creating the experiment, the service selected was HiperGator Vision Service, and both uploaded files were used as the data source. The time to train the AI model took a few minutes. There were reports from other colleagues that they had run into issue with this step, as the HiperGator Vision service was not running properly. Fortunately the AI training completed for my experiment after the bug had been fixed by CogUniveristy. After Training completed, the output were 2 text files of the HiperGatorVision service log and a summary.



Uploading the data for training



Creating the AI image recognition experiment

Submitting Experiment for training to HiperGator

Training

Trained and Ready screen with output log and summary

```
/
c0704a-s1.ufhpc
Thu Mar 28 13:50:55 EDT 2024
No GPU detected, loading a CPU version.
Running job request /blue/coguniversity/cml-prod/experiments/82f8adf4-0a4b-409a-bb61-7e0fd37141cc/data null null /blue/coguniversity/cml-prod/experiments/82f8adf4-0a4b-409a-bb61-7e0fd37141cc/out
7e0fd37141cc/model 82f8adf4-0a4b-409a-bb61-7e0fd37141cc uf-hipergator c54548dc8c34457bb417224d87756748/uf_0002/experiments/82f8adf4-0a4b-409a-bb61-7e0fd37141cc HiPerGator-Vision
2024-03-28 13:50:56.851360: I tensorflow/stream_executor/platform/default/dso_loader.cc:49] Successfully opened dynamic library libcudart.so.11.0
2024-03-28 13:51:03.895015: I tensorflow/compiler/jit/xla_cpu_device.cc:41] Not creating XLA devices, tf_xla_enable_xla_devices not set
2024-03-28 13:51:03.896169: W tensorflow/stream_executor/platform/default/dso_loader.cc:60] Could not load dynamic library 'libcuda.so.1'; dlerror: libcuda.so.1: cannot open shared object file:
2024-03-28 13:51:03.896195: W tensorflow/stream_executor/cuda/cuda_driver.cc:326] failed call to cuInit: UNKNOWN ERROR (303)
2024-03-28 13:51:03.896220: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:156] kernel driver does not appear to be running on this host (c0704a-s1.ufhpc): /proc/driver/nvidia/version doe
2024-03-28 13:51:03.896522: I tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (oneDNN) to use the following CPU ins
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
2024-03-28 13:51:03.896610: I tensorflow/compiler/jit/xla_gpu_device.cc:99] Not creating XLA devices, tf_xla_enable_xla_devices not set
2024-03-28 13:51:04.045667: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:116] None of the MLIR optimization passes are enabled (registered 2)
2024-03-28 13:51:04.057357: I tensorflow/core/platform/profile_utils/cpu_utils.cc:112] CPU Frequency: 1996350000 Hz
AKModel2 starting
Creating Image Classification Model
loading training dataset
Found 60 files belonging to 4 classes.
Using 48 files for training.
loading validation dataset
Found 60 files belonging to 4 classes.
Using 12 files for validation.
fitting training data

Search: Running Trial #1

Hyperparameter    |Value             |Best Value So Far
image_block_1/b...|vanilla           |?
image_block_1/n...|True              |?
image_block_1/a...|False             |?
image_block_1/c...|3                 |?
image_block_1/c...|1                 |?
image_block_1/c...|2                 |?
image_block_1/c...|True              |?
image_block_1/c...|False             |?
image_block_1/c...|0.25              |?
image_block_1/c...|32                |?
image_block_1/c...|64                |?
classification_...|flatten           |?
classification_...|0.5               |?
optimizer         |adam              |?
learning_rate     |0.001             |?
```

The training process used CPU instead of GPU processing, indicated by TensorFlow warnings of failing to load CUDA libraries, which was caused by no GPU support for this training experiment. The training was conducted with 20 epochs, with parameters being tested such as image block type, normalization, augmentation, convolution block depth, and optimizer settings.

```
Epoch 1/20

1/1 [==============================] - ETA: 0s - loss: 1.4780 - accuracy: 0.2812
1/1 [==============================] - 6s 6s/step - loss: 1.4780 - accuracy: 0.2812 - val_loss: 24.8129 - val_accuracy: 0.3750
Epoch 2/20

1/1 [==============================] - ETA: 0s - loss: 21.0812 - accuracy: 0.3438
1/1 [==============================] - 5s 5s/step - loss: 21.0812 - accuracy: 0.3438 - val_loss: 9.7577 - val_accuracy: 0.4375
Epoch 3/20

1/1 [==============================] - ETA: 0s - loss: 4.2606 - accuracy: 0.6250
1/1 [==============================] - 6s 6s/step - loss: 4.2606 - accuracy: 0.6250 - val_loss: 13.7769 - val_accuracy: 0.3125
Epoch 4/20

1/1 [==============================] - ETA: 0s - loss: 6.7240 - accuracy: 0.5625
1/1 [==============================] - 5s 5s/step - loss: 6.7240 - accuracy: 0.5625 - val_loss: 10.5579 - val_accuracy: 0.3750
Epoch 5/20

1/1 [==============================] - ETA: 0s - loss: 3.5321 - accuracy: 0.6250
1/1 [==============================] - 5s 5s/step - loss: 3.5321 - accuracy: 0.6250 - val_loss: 6.3897 - val_accuracy: 0.3750
Epoch 6/20

1/1 [==============================] - ETA: 0s - loss: 0.5420 - accuracy: 0.8438
1/1 [==============================] - 7s 7s/step - loss: 0.5420 - accuracy: 0.8438 - val_loss: 5.5841 - val_accuracy: 0.3750
Epoch 7/20

1/1 [==============================] - ETA: 0s - loss: 0.0382 - accuracy: 1.0000
1/1 [==============================] - 6s 6s/step - loss: 0.0382 - accuracy: 1.0000 - val_loss: 7.1130 - val_accuracy: 0.3750
Epoch 8/20

1/1 [==============================] - ETA: 0s - loss: 0.4108 - accuracy: 0.8438
1/1 [==============================] - 5s 5s/step - loss: 0.4108 - accuracy: 0.8438 - val_loss: 7.4062 - val_accuracy: 0.3750
Epoch 9/20

1/1 [==============================] - ETA: 0s - loss: 0.3791 - accuracy: 0.8438
1/1 [==============================] - 9s 9s/step - loss: 0.3791 - accuracy: 0.8438 - val_loss: 6.9181 - val_accuracy: 0.3750
Epoch 10/20

1/1 [==============================] - ETA: 0s - loss: 0.1157 - accuracy: 0.9375
1/1 [==============================] - 8s 8s/step - loss: 0.1157 - accuracy: 0.9375 - val_loss: 6.2464 - val_accuracy: 0.3750
Epoch 11/20

1/1 [==============================] - ETA: 0s - loss: 0.0251 - accuracy: 1.0000
1/1 [==============================] - 5s 5s/step - loss: 0.0251 - accuracy: 1.0000 - val_loss: 5.5852 - val_accuracy: 0.3750
Epoch 12/20

1/1 [==============================] - ETA: 0s - loss: 0.0102 - accuracy: 1.0000
1/1 [==============================] - 5s 5s/step - loss: 0.0102 - accuracy: 1.0000 - val_loss: 4.9490 - val_accuracy: 0.3750
Epoch 13/20

1/1 [==============================] - ETA: 0s - loss: 0.0059 - accuracy: 1.0000
1/1 [==============================] - 6s 6s/step - loss: 0.0059 - accuracy: 1.0000 - val_loss: 4.3583 - val_accuracy: 0.3750
Epoch 14/20

1/1 [==============================] - ETA: 0s - loss: 0.0047 - accuracy: 1.0000
1/1 [==============================] - 5s 5s/step - loss: 0.0047 - accuracy: 1.0000 - val_loss: 3.8472 - val_accuracy: 0.3750
Epoch 15/20

1/1 [==============================] - ETA: 0s - loss: 0.0074 - accuracy: 1.0000
1/1 [==============================] - 6s 6s/step - loss: 0.0074 - accuracy: 1.0000 - val_loss: 3.4286 - val_accuracy: 0.3750
Epoch 16/20

1/1 [==============================] - ETA: 0s - loss: 0.0077 - accuracy: 1.0000
1/1 [==============================] - 5s 5s/step - loss: 0.0077 - accuracy: 1.0000 - val_loss: 3.0993 - val_accuracy: 0.5000
Epoch 17/20

1/1 [==============================] - ETA: 0s - loss: 0.0108 - accuracy: 1.0000
1/1 [==============================] - 5s 5s/step - loss: 0.0108 - accuracy: 1.0000 - val_loss: 2.8638 - val_accuracy: 0.5000
Epoch 18/20

1/1 [==============================] - ETA: 0s - loss: 0.0133 - accuracy: 1.0000
1/1 [==============================] - 5s 5s/step - loss: 0.0133 - accuracy: 1.0000 - val_loss: 2.7114 - val_accuracy: 0.5000
Epoch 19/20

1/1 [==============================] - ETA: 0s - loss: 0.0138 - accuracy: 1.0000
1/1 [==============================] - 5s 5s/step - loss: 0.0138 - accuracy: 1.0000 - val_loss: 2.6300 - val_accuracy: 0.5000
Epoch 20/20

1/1 [==============================] - ETA: 0s - loss: 0.0125 - accuracy: 1.0000
1/1 [==============================] - 5s 5s/step - loss: 0.0125 - accuracy: 1.0000 - val_loss: 2.6136 - val_accuracy: 0.4375
[2K
[2K
Trial 1 Complete [00h 01m 59s]
val_loss: 2.613588809967041

Best val_loss So Far: 2.613588809967041
```

Training the Image Recognition model with 20 epochs

```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 180, 180, 3)]     0
_________________________________________________________________
cast_to_float32 (CastToFloat (None, 180, 180, 3)       0
_________________________________________________________________
normalization (Normalization (None, 180, 180, 3)       7
_________________________________________________________________
conv2d (Conv2D)              (None, 178, 178, 32)      896
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 176, 176, 64)      18496
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 88, 88, 64)        0
_________________________________________________________________
dropout (Dropout)            (None, 88, 88, 64)        0
_________________________________________________________________
flatten (Flatten)            (None, 495616)            0
_________________________________________________________________
dropout_1 (Dropout)          (None, 495616)            0
_________________________________________________________________
dense (Dense)                (None, 4)                 1982468
_________________________________________________________________
classification_head_1 (Softm (None, 4)                 0
=================================================================
Total params: 2,001,867
Trainable params: 2,001,860
Non-trainable params: 7
_________________________________________________________________
```

Output Summary

Epoch	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy
1	1.478	28.12%	24.8129	37.50%
2	21.0812	34.38%	9.7577	43.75%
3	4.2606	62.50%	13.7769	31.25%
4	6.724	56.25%	10.5579	37.50%
5	3.5321	62.50%	6.3897	37.50%
6	0.542	84.38%	5.5841	37.50%
7	0.0382	100.00%	7.113	37.50%
8	0.4108	84.38%	7.4062	37.50%
9	0.3791	84.38%	6.9181	37.50%
10	0.1157	93.75%	6.2464	37.50%
11	0.0251	100.00%	5.5852	37.50%
12	0.0102	100.00%	4.949	37.50%
13	0.0059	100.00%	4.3583	37.50%
14	0.0047	100.00%	3.8472	37.50%
15	0.0074	100.00%	3.4286	37.50%
16	0.0077	100.00%	3.0993	50.00%
17	0.0108	100.00%	2.8638	50.00%
18	0.0133	100.00%	2.7114	50.00%
19	0.0138	100.00%	2.63	50.00%
20	0.0125	100.00%	2.6136	43.75%

Table summary of training accuracy

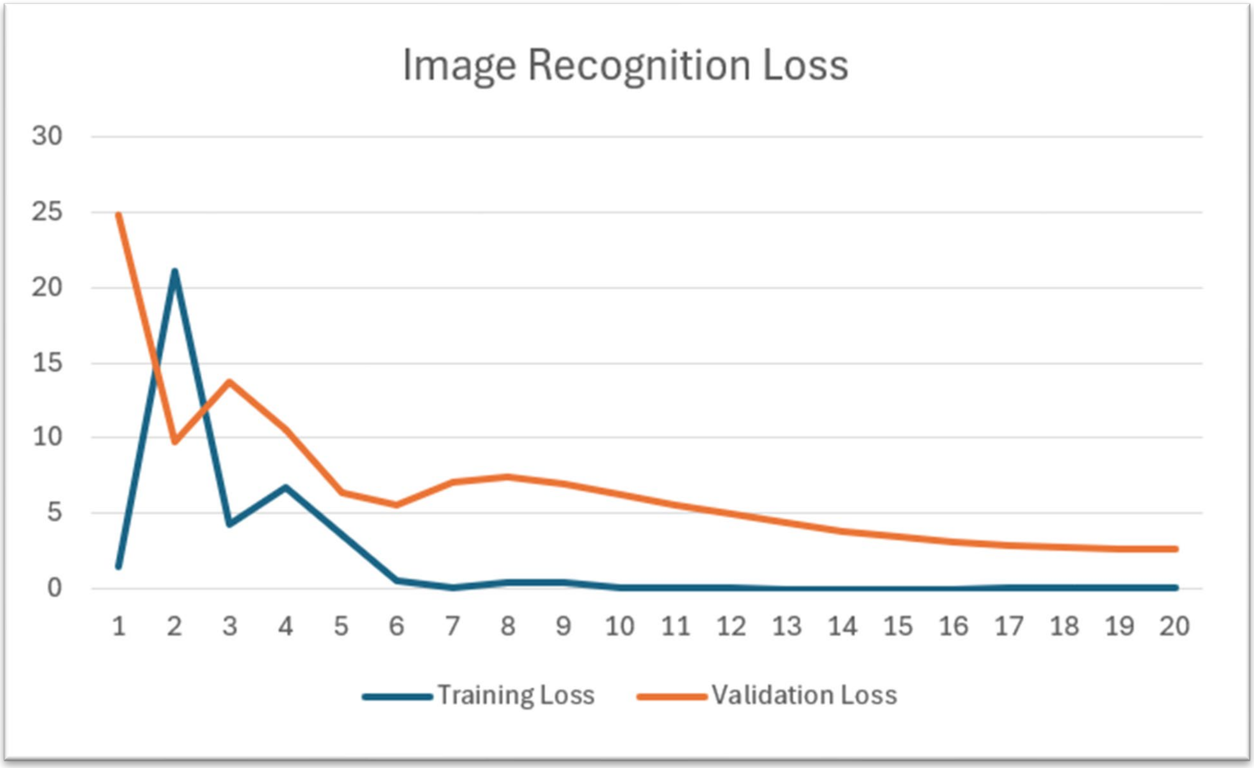


Image Recognition Loss
30
25
20
15
10
5
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
Training Loss
Validation Loss

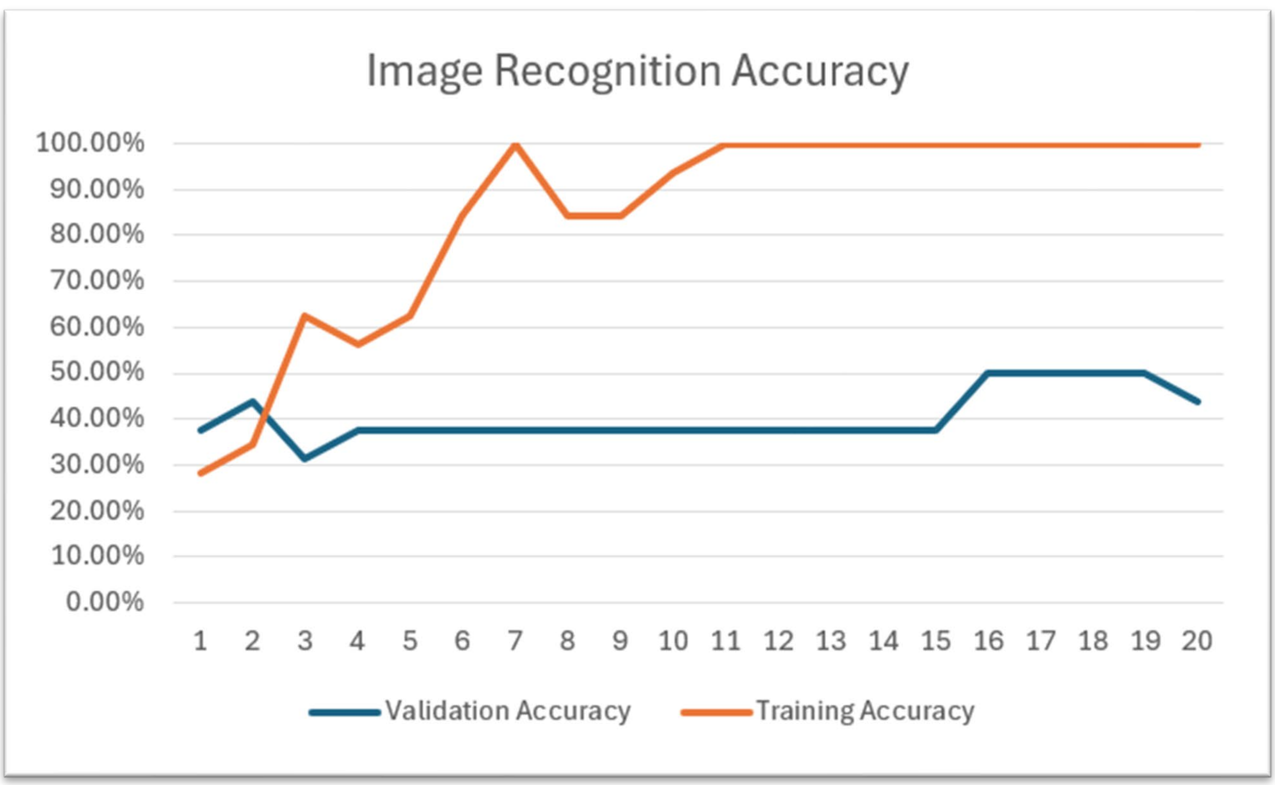
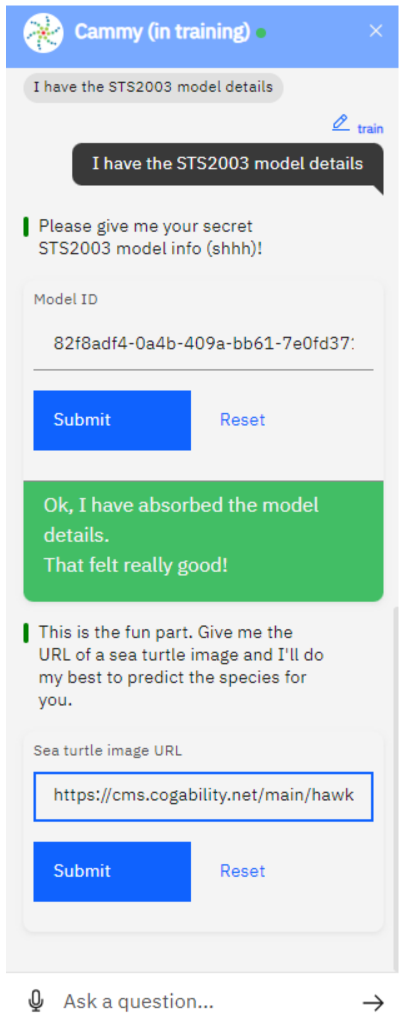


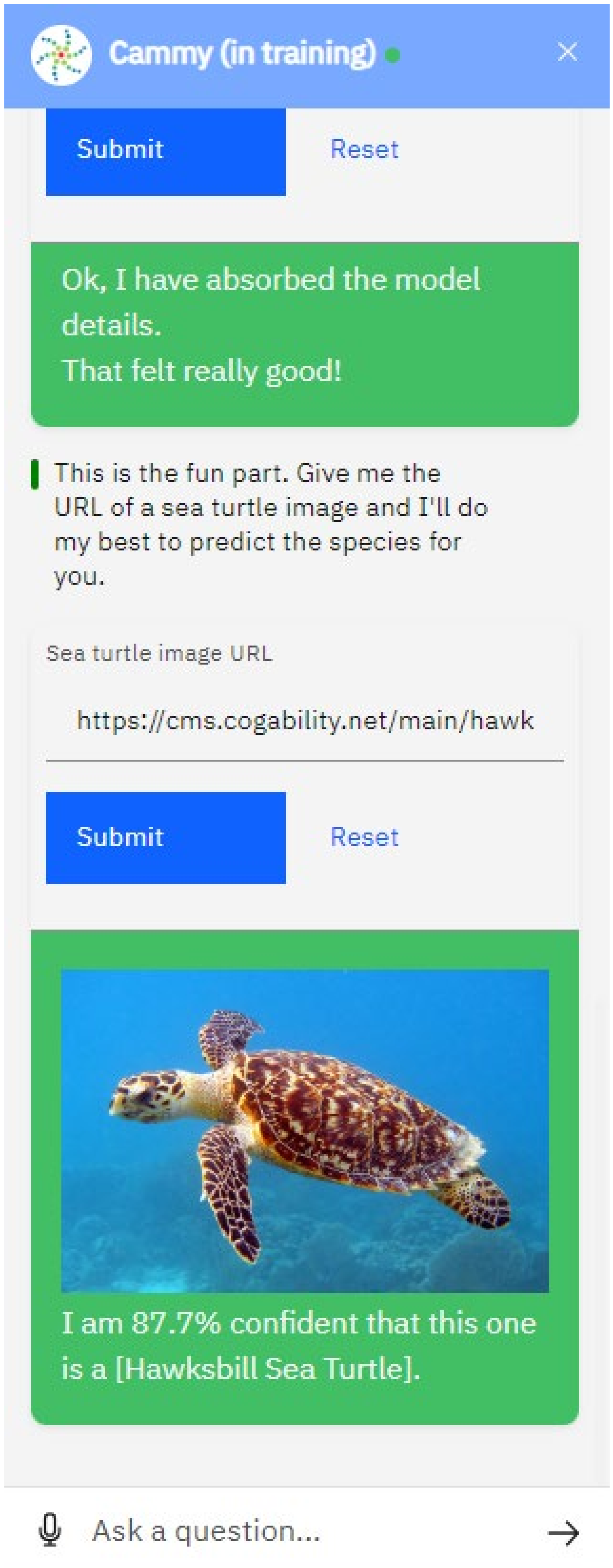
Image Recognition Accuracy
100.00%
90.00%
80.00%
70.00%
60.00%
50.00%
40.00%
30.00%
20.00%
10.00%
0.00%
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
Validation Accuracy
Training Accuracy

Based on the training results, the training accuracy reached 100% by Epoch 7, indicating the model is perfectly fitting the training data. However, the validation accuracy does not improve at similar rate, only reaching 43.7% by end of training. This indicates that the training data is overfitted and has a lack of different data to analyze and compare.

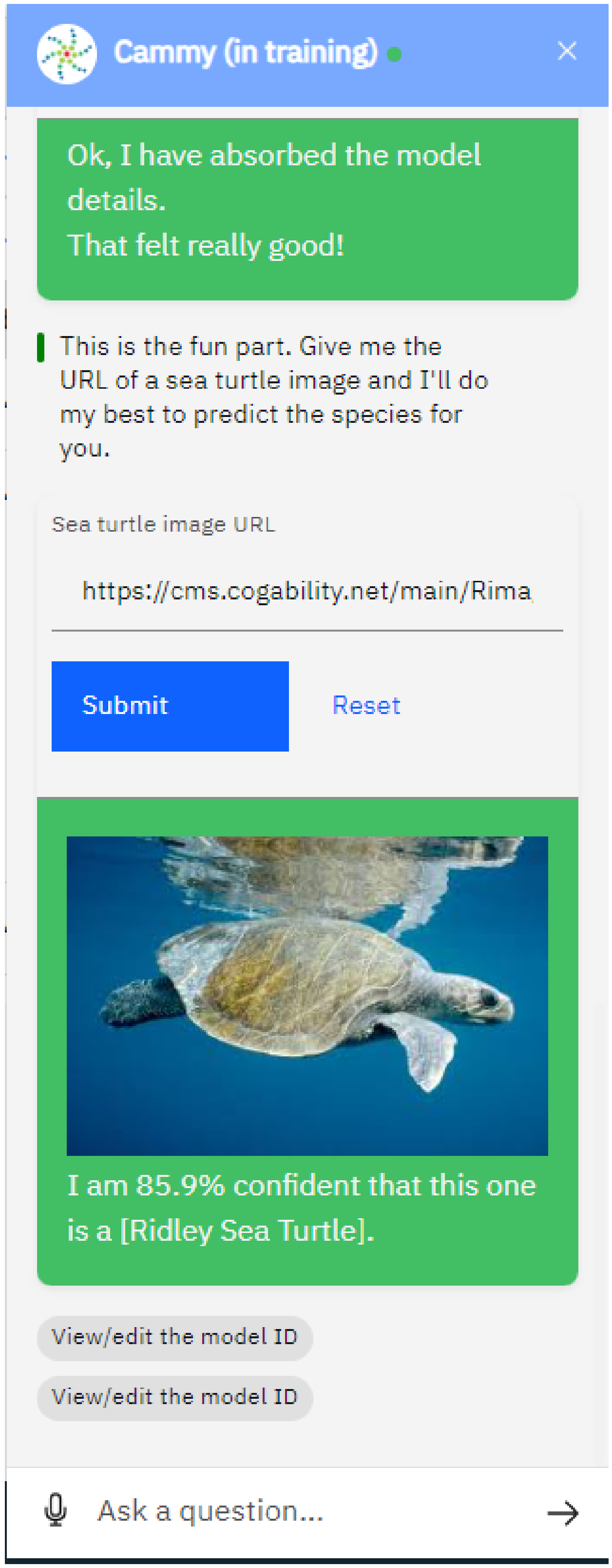
With the model trained on the images and labels, the model was applied to Cammy chatbot to allow it to predict images uploaded. For testing the image recognition, we used 9 test images provided by CogUniversity in a csv file. There were 2 test images per species that Cammy had not seen before, and each of the image urls were pasted into the input and Cammy made the species prediction.



Attaching the trained AI model to Cammy



Correct prediction for Hawksbill Sea Turtle

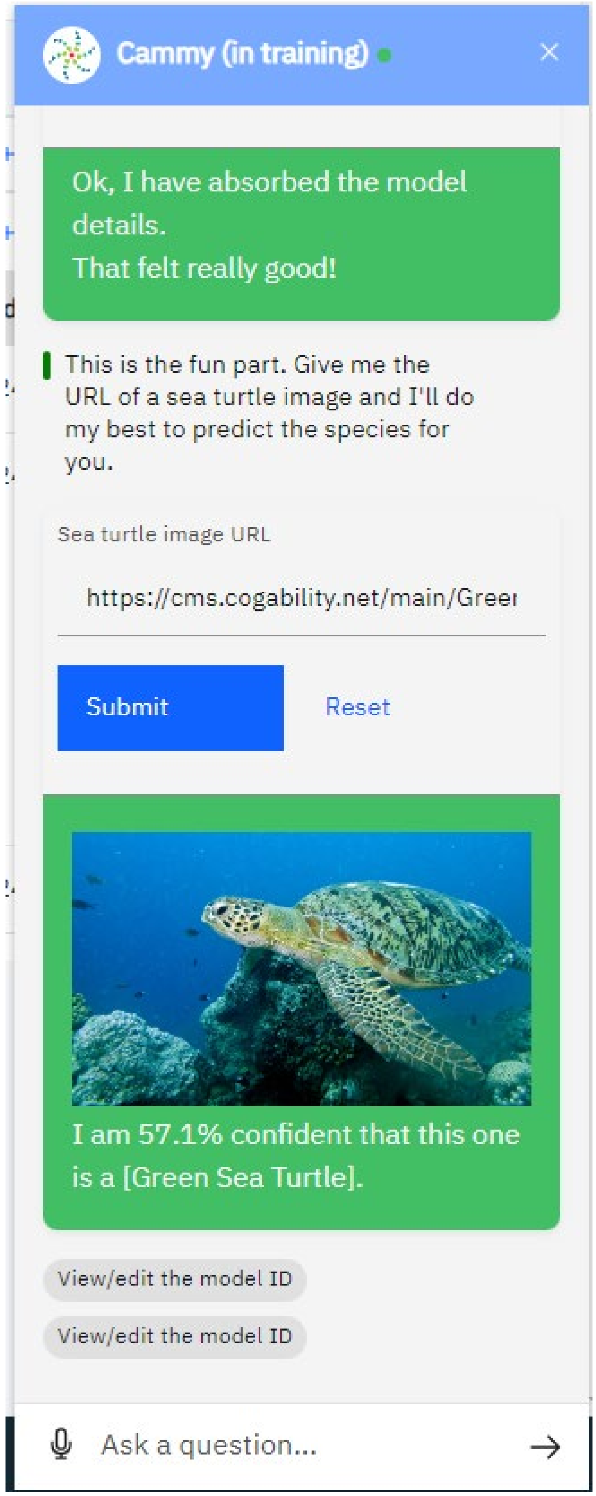


Correct prediction for Ridley Sea Turtle

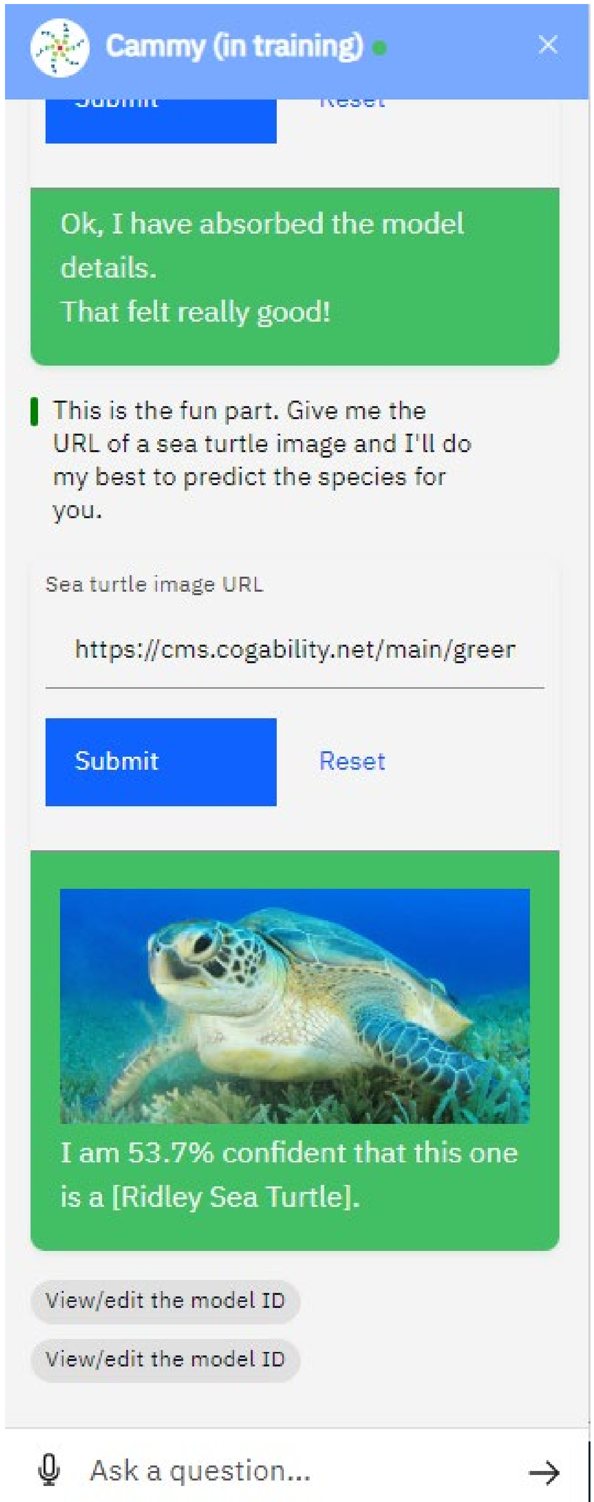
Cammy had greatest success at recognizing the Hawksbill Sea Turtles, with accuracy over 85%, and was also able to identify the Ridley Sea Turtle, with accuracy 60 – 85%.

Sea Turtle Test Images	Prediction	Accuracy
Hawksbill sea turtles		
https://cms.cogability.net/main/hawksbill_sea_turtle.jpeg	TRUE	87.70%
https://cms.cogability.net/main/himages.jpeg	TRUE	95%
Loggerhead sea turtles		
https://cms.cogability.net/main/La29582bef3bdf958ffcb31801a9639cd.jpeg	False - Hawksbil	84.60%
https://cms.cogability.net/main/L218933603_78ec25bc66R.jpg	False - Green	47.90%
Ridley sea turtles		
https://cms.cogability.net/main/Rimages.jpeg	TRUE	85.90%
https://cms.cogability.net/main/ridley_sand.jpeg	TRUE	61%
Green sea turtles		
https://cms.cogability.net/main/Green_Turtle_Chelonia_mydas_6133097542.jpeg	TRUE	57.10%
https://cms.cogability.net/main/green_weeds.jpeg	False - Ridley	53.70%
https://cms.cogability.net/main/green-sea-turtle_australia_shutterstock.jpeg	False - Hawksbb	49.50%

Cammy failed to recognize the Loggerhead Sea Turtle images, predicting the images were of HawksBill at 85% accuracy, and also a 47% accuracy predicton of HawksBill as a Green Sea Turtle. There were mixed results with predicting the Green Sea Turtle images, with only 1 of the 3 images being identified correctly with 57% accuracy, and the other 2 images being identified at 50% accuracy as Ridley and Hawksbill Sea Turtles.

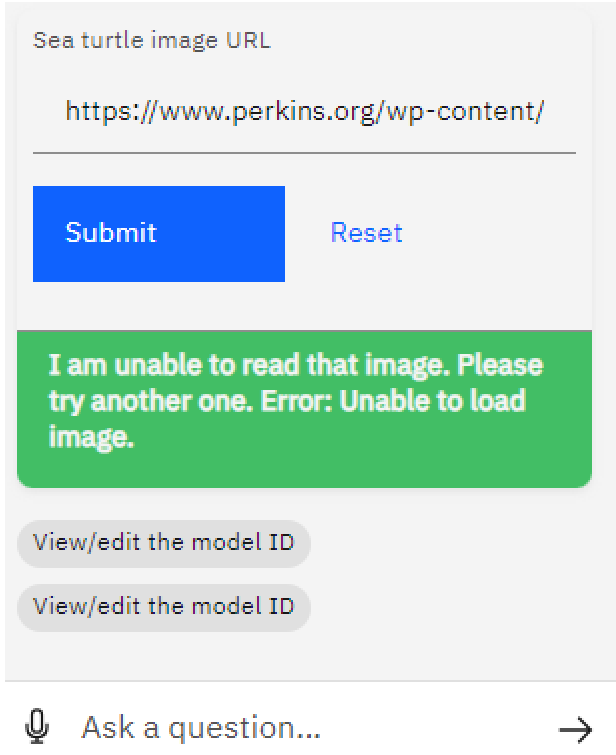


Correct prediction for Green Sea Turtle



Incorrect prediction for Green Sea Turtle

Incorrect prediction for Loggerhead Sea Turtle



There was an error encountered when trying to use an external image URL that was not part of the test images provided by CogUniveristy.

Summary

The Sea Turtle Conservation AI Project, implemented through three progressive experiments using the AI chatbot Cammy, showcased the multifaceted applications of AI in wildlife conservation.

- The first experiment involved training Cammy to answer community-sourced questions about sea turtles, enhancing public knowledge and engagement. The chatbot utilized a database of frequently asked questions to provide accurate, contextual answers, employing Natural Language Processing (NLP) for continuous improvement of response quality.

- The second experiment extended Cammy's capabilities to include the prediction of sea turtle weights based on visual data obtained via stereo camera images. Utilizing the AutoKeras Regression algorithm, the model leveraged existing data from over 5,700 sea turtles to accurately estimate weights from measurements, demonstrating the practical application of machine learning in non-invasive wildlife research.

- The third experiment focused on species identification, where Cammy was trained to recognize different species of sea turtles from a curated dataset of images. This phase highlighted the challenges and limitations of AI, particularly in terms of data quality and the training environment's congruence with real-world conditions. Despite mixed results in image recognition accuracy, the experiment provided valuable insights into the factors affecting AI performance in biodiversity conservation tasks.

Recommendations

To enhance the reliability of AI tools like Cammy, it is essential to use a diverse and extensive dataset that mirrors the variety of real-world conditions under which the chatbot will operate. Future experiments should include images from various sources and settings, ensuring that the AI can accurately function with data like what it will encounter in practical use. User feedback can also help refine the chatbot's answers and functionalities. Ethical and Legal Considerations in AI training, particularly concerning copyright infringement. Going forward, it is recommended to utilize copyright-free images or obtain proper licenses for all training materials to avoid legal issues and ensure ethical AI training practices. Future iterations of the project could explore additional functionalities such as integrating geographic mapping turtle sightings and migrations. Future projects could leverage similar AI technologies for other conservation and research areas, potentially transforming how data-driven insights are generated and applied in environmental science at large.

References

1. OpenAI. "ChatGPT: Advanced AI Language Model for Conversational Agents." https://chat.openai.com/ Accessed 3/28/24

2. Google. "Google Gemini AI." https://gemini.google.com/ Accessed 3/1/24

3. "National Oceanic and Atmospheric Administration (NOAA)", https://www.fisheries.noaa.gov/sea-turtles Accessed 3/1/24

4. "National Geographic" https://www.nationalgeographic.com/animals/reptiles/facts/sea-turtles Accessed 3/1/24

5. See Turtles Organization. https://www.seeturtles.org/baby-turtles Accessed 3/1/24

6. "MakeSense." MakeSense.ai, n.d., www.makesense.ai/. Accessed 3/28/24

7. "CogAbility." CogAbility, n.d., www.cogability.com/. Accessed 3/28/24