Todd Crooks

ID # 010138754

C964 – CS Capstone Task 2

**Part A: Letter of Transmittal**

February 23, 2024

Jack Donaghy PhD, Chief Research Officer

Ocean Research and Conservation Administration (ORCA)

123 Ocean Ave

Clearwater, FL

Dear Doctor Donaghy,

To better serve our oceans, we must better understand our oceans. Anthropogenic Climate Change (ACC) is an issue that will undoubtedly have serious implications for sea level rise, ocean carbonization, and changes in species around the world. Collecting accurate species counts is an important metric of monitoring change, but it is a tedious one that often requires multiple manual rechecks of photos and video collected by divers and submersible-attached cameras.

I am therefore proposing a Machine Learning model that will potentially reduce the amount of time spent on manually counting species and rechecking data by accurately classifying sea animals by species from still images. As your doctorate thesis was concerning bioluminescence in the Mauve Stinger jellyfish (*Pelagia noctiluca*) I decided to build out a prototype that will classify jellyfish into one of six species – the other five being the Barrel jellyfish (Rhizostoma *pulmo*), the Blue jellyfish (*Cyanea lamarckii*), the Compass jellyfish (*Chrysaora hysoscella*), the Lion’s Mane jellyfish (*Cyanea capillata*), and the Moon jellyfish (*Aurelia aurita*). This initial training and test data was pulled from a publicly published data set of jellyfish images from these six species.

This model aims to significantly reduce the amount of time and resources spent to perform species identification as a part of species counts. By automating the process of correctly identifying sea life species, we could greatly speed up calculations of species changes over time.

Currently, this prototype is only designed to work with the previously mentioned species of jellyfish, but with your approval I will construct a full-fledged development timeline and complete cost analysis for this project to move forward. Ideally, the model will be able to accurately classify all manner of invertebrate sea creatures within six months, with many more species being added over time.

The model is hosted in a Jupyter Notebook currently, which means anyone wanting to try it will need to have IT set them up with the correct tools, but it also allows for individual researchers to use the product simultaneously without cross-contamination of results or competing bandwidth using locally hosted image files.

The model will be developed via an agile methodology to allow for early access among researchers and easy updates to the Convolutional Neural Network at the heart of the model. Funding for this project is expected to remain low, as all the data for training can come from freely available published image sources in addition to our internal database of images from previous counts. Additionally, the software and related libraries are all open source which will aid in keeping the project’s costs to a minimum. Since we already release all of our image data for public use, we could even release the model as an open source project to get additional assistance and good publicity for ORCA. Since we will not need to use any private or sensitive data, there are no legal liabilities for building and releasing the model.

All told, this project should cost less than $1000 for additional computational resources over the next 6 months, not including from the cost of my continued salary. The board is certain to appreciate the cost savings over the next two years as we can reduce the research team’s wasted time on tedious work and instead shift them more towards new research.

I strongly believe that I am the best man to spearhead this project as I not only have my BSc in Computer Science and two years’ experience in Data Analysis, but I also have my BSc in Environmental Science and a passion for environmental conservation.

Kind Regards,

Todd Crooks, Associate Data Analyst I

Ocean Research and Conservation Administration

**Part B: Executive Summary**

February 27, 2024

Gary Scharf, Director of IT

Ocean Research and Conservation Administration

* **Problem Statement and Development Opportunity**

I have previously discussed with Dr. Donaghy the need for an automated system for identifying and classifying sea life for species counts. As it stands, currently our research teams are wasting dozens of hours each quarter on performing manual species counts from still photographs taken by our field teams. This is a poor use of their time and a cost sink for the organization. I am looking to reduce the man hours spent on these counts by creating a Machine Learning model that will quickly and accurately classify these images.

* **Customer Description**

Initially, this model will be maintained entirely in-house and used only by our research teams to automate some of the tedious tasks that take away from their research and publication. However, Dr. Donaghy has confirmed that many other conservation organizations are certain to find our tool useful for their work as well, so there is the possibility that we could open it up to other NGOs. The board may initially wish to sell the tool, but I feel that releasing it as an open source project would attract the attention of many other researchers willing to contribute to its development and maintenance as well as attracting positive publicity for ORCA.

* **Existing Gaps**

Currently, all identification and classification work is done manually by research teams and interns. This is an inefficient use of their time, even for the interns. Additionally, manual counts are prone to human error, and have led to documented cases of ‘burn out’ among the research staff. By developing and deploying an automated classification model we can not only save the organization on wasted funding, but also increase staff retention.

* **Data**

The training and validation data will be obtained from publicly available sea life image repositories. We regularly publish our own compiled image data obtained by our field teams, as do many other conservation-focused NGOs. All image data will be collected and cleaned to remove duplicates, missing pixels, and generally poor images. The model prototype, a minimum viable product version, runs all image data through a transformation process to increase sample size and streamline image filtering. Images are resized and cropped before being converted to tensors to ensure the images best contain the subject, and all images go through a normalization process to streamline similarity matching. Normalization values are currently based on the ImageNet recommendations, but values can be adjusted based on additional image preprocessing as necessary.

* **Methodology**

As mentioned above, all image data will be compiled, cleaned, and normalized to remove duplicate information and to best allow for pattern matching by the machine learning algorithm. The model will run images through multiple convolutional layers and image filters to match visual patterns, allowing for species identification.

* **Outcomes and Deliverables**

The primary deliverables for this project are of course the classification model itself as well as the accompanying documentation to allow for ease of use by non-technical staff. The expected outcome is a reduction in time spent on manual identification and classification of sea species. After an initial time investment among research staff, estimated around 5-10 hours per Team, we are projecting a reduction of 35-40% in time spent on classification tasks across all research teams over the next 6 months.

* **Implementation**

The project will be developed in an ongoing, agile process. This will allow for continuous updates and improvements to be delivered to the research teams. The first official release will include a GUI that can be used to easily check for updates and pull them from the repository, allowing non-technical staff to always have the most up-to-date model for their work. An agile development process also allows for easy patches to fix discovered bugs.

* **Validation**

The model, given its intended agile development cycle, will need to be continuously validated to ensure accuracy in classification. Validation can be done using known good, labeled data to compare to predicted values. The included prototype has been run through validation testing to ensure accurate predictions and allow for tweaking the image filters.

* **Environments, Tools, and Resources**

The model is built in Jupyter Notebooks using Python and the Numpy, Pandas, and PyTorch libraries. The model was built on a single computer by me but should be moved to a server for further development and production use. Source code should be housed in an external repository, such as GitHub or GitLab, to ensure changes can be made easily and rolled back as necessary. I can continue solo development of the model, unless you deem it worthwhile to increase the development team. At least one or two Quality Assessment team members will be needed on a regular basis to validate the model runs properly and accurately classifies images. Ideally, I would like to request to lead a small team of three people – myself for primary development, a software engineer to build the researcher-facing GUI and to assist in building a data pipeline, and a QA engineer to continually check for issues and validate the model.

* **Timeline**

I have included a proposed timeline below based on my recommendation of a three-person team:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Step #** | **Prerequisite Step(s)** | **Step** | **Expected Completion** | **Resources** |
| 0 |  | Build prototype model | Complete! | Discussed above |
| 1 |  | Design front-end GUI | 3/4 – 3/8 | Dev team |
| 2 | 1 | Build Front-End | 3/11 – 3/22 | Dev team, Dash Framework |
| 3 |  | Design data pipeline | 3/25 – 3/29 | Dev team |
| 4 | 3 | Build first data pipeline | 4/1 – 4/4/12 | Myself, Python |
| 5 | 2, 4 | QA initial build | 4/15 – 4/19 | QA, Jenkins |
| 6 | 2, 4, 5 | Deploy model to select research team(s) for initial feedback | 4/22 – 5/17 | Dev team, GitLab |
| 7 | 6 | Design updates based on feedback | 5/20 – 5/24 | Dev team |
| 8 | 6, 7 | Build updates | 5/27 – 6/14 | Dev team, JuPyter Notebook, Dash Framework, PyTorch |
| 9 | 8 | QA new model version | 6/17 – 6/28 | QA, Jenkins |
| 10 | 8, 9 | Deploy new model | 7/1 – 7/5 | Dev team, GitLab |
| ∞ | | Design updates from feedback and issue tickets | Continuous! | Dev team |
| ∞ | | Build updates | Continuous! | Dev team |
| ∞ | | QA updates | Continuous! | QA, Jenkins |
| ∞ | | Deploy updates | Continuous! | Dev team, GitLab |

**Part C: Functional Data Product**

Included in the zip file is a Data folder that includes all of the images for training and testing the model, a Jupyter Notebook file (“TCrooks\_C694 .ipynb”) that includes the source code for the model, a copy of the saved state dictionary of the model (“TCrooks\_C964.pt”), a copy of this write up, and an additional copy of Task 1.

Part D below walks through the process for a user to run the model using the included data as well as an appendix to ensure all required tools are installed.

**Part D: Post-Implementation Report**

**Solution Summary**

This Convolutional Neural Network model quickly and accurately classifies images of jellyfish by species based on a supervised learning process. This model reduces time spent on manual tasks for environmental research by automating a repetitive task.

**Data Summary & Source Code**

The imagery data comes from a publicly available Kaggle dataset of jellyfish images, available at this link:

<https://www.kaggle.com/datasets/anshtanwar/jellyfish-types>

These images were randomly rotated and/or mirrored to increase the dataset for training. Images were normalized using ImageNet recommended mean and standard deviation values. Images were transformed into a tensor and run through multiple convolutional layers and image filters to recognize image patterns.

All code for the model is included in the zip folder in a Jupyter Notebook file, as well as backup source code python file. Additionally, the entire project can be accessed from GitHub here:

https://github.com/Ujili/C964

**Hypothesis Verification**

The model was able to quickly classify images of jellyfish by species with moderate (>40%) accuracy. The model can train from hundreds of images and then classify 40 images by species in under a minute, far faster than any human could. Thus, with additional training and improvements the model will be able to accurately automate the tedious work of manually identifying species from photos and images taken from recorded video.

**Visualizations**

Below are snapshots of a random sample of the image data used for training, without (first) and with (second) normalization applied:

A collage of jellyfish

Description automatically generated

Below is the text output from training and running the model over 5 epochs:

A screenshot of a computer

Description automatically generated

Additionally, matplotlib was used to visualize the loss of both the training and testing sets.

A graph with lines and text

Description automatically generated

Matplotlib was also used to visualize the accuracy of both the training and testing runs.

A graph on a screen

Description automatically generated

**Accuracy and Results**

The image below shows that test cases were above 40% accuracy every time, which is notably higher than random chance (around 16.66%, or one in six).

A screenshot of a computer code

Description automatically generated

Therefore, the model is successful given that it was able to classify 40 images in under a minute with more than 40% accuracy.

**User Guide**

The below guide walks through how to view and use the included Jupyter Notebook file to run the model. If you need to install Jupyter Notebook or are missing any of the required dependencies, plus see Appendix I.

After running Jupyter Notebook, open the included ‘TCrooks\_C694.ipynb’ file. First, we will run the first section to ensure all of the required libraries are imported and ready. If you have any issues, see Appendix I for installing dependencies. You can run each section either by clicking in the cell and clicking ‘Run’ at the top, or by pressing Ctrl+Enter.

A screen shot of a computer

Description automatically generated

When you see the below next to the cell, the imports have finished successfully.



Next, you will need to run the next cell the same way. This section creates the two transforms used to clean, randomize, and normalize the images used for both training and testing. If you see “It worked!” printed out, then the cell was run successfully.

A screenshot of a computer code

Description automatically generated

Next we will load the data using the following cell. If “Success!” prints, then the data was loaded successfully. If you do not see this printed, double check to ensure that you have the data folder in the same root directory as the Jupyter Notebook.

A screen shot of a computer code

Description automatically generated

The next two cells are optional, but can be run to view the class names and the amount of data for training and testing.

A screenshot of a computer

Description automatically generated

The next cell is also optional, but running the cell shows a random sample of training data with and without normalization values. If this runs successfully, you should see two sets of the same images, such as shown below.

A screenshot of a computer program

Description automatically generated

A collage of jellyfish

Description automatically generated

The next cell is not optional, as it contains the class for the model itself. The model is made up of three fully connected layers and two convolutional layers with over 3.5 million parameters. This cell then instantiates the class as the model object CNNmodel. We set a manual seed using PyTorch because there is some inherent randomness in Machine Learning, and setting a manual seed allows for my results to be replicated.

Finally, this cell sets the criteria for testing and validation using CrossEntropy loss and the Adam optimizer and a learning rate of 0.00005. Increasing this learning rate can increase the speed of the model at the possible cost of accuracy.

A computer screen shot of a program

Description automatically generated

The next cell is optional and can be run to examine the number of parameters within the model.

A screenshot of a computer code

Description automatically generated

The next cell runs the model itself and may take a few moments to run. On my computer it completes in under a minute, but this may be quicker or slower depending on CPU speed.

A screenshot of a computer program

Description automatically generated

A screen shot of a computer program

Description automatically generated



Finally, the last three cells are run to show the accuracy of the model. The first charts the loss of the training and testing batches, the second charts the accuracy of the model across both training and testing, and the third shows the accuracy of the testing runs in plain text.

A graph with lines and text

Description automatically generated

A graph on a screen

Description automatically generated

A computer screen shot of a computer code

Description automatically generated

**Part E: Sources**

Tanwar, A. (2023, October 23). *jellyfish classification starter notebook*. Kaggle. https://www.kaggle.com/code/anshtanwar/jellyfish-classification-starter-notebook

**APPENDIX I:**

**Installing Python:**

Python can be downloaded for free from <https://www.python.org/downloads/>

Download the version of Python for your Operating System. When the download finishes, activate the file and follow the instructions to install Python.

**Installing Anaconda and Jupyter Notebook:**

This project was created in Jupyter Notebook, included in Anaconda. The included model should run in any version of Jupyter Notebook that is installed, but Anaconda can be downloaded for free from <https://www.anaconda.com/download>

When the download finishes, run the file and follow the instructions to install Anaconda as well as Jupyter Notebook.

**Installing Required Dependencies:**

The following commands can be run in your terminal or command shell to download and install the required dependencies:

* pip install numpy
* pip install pandas
* pip install torch
* pip install torchvision