April 14, 2021

Milestone 2 - Team Ornithomimus

State of the Project

We have a basic implementation of a RESTful API that will allow the flutter application to request hike information based on a unique hike-name. The API sends the user back a JSON-formatted response that can load information for the hike. This will be required to send the flutter application recommended hikes. We are working on allowing the API to receive information and communicate with the image classifier and recommender system for M3.

The frontend application currently contains a profile page, a hikes page, and a matches page. The matches page displays photos of matched hikes on a scrollable page in a grid format. The hikes page has a list of photos which the user can either swipe to the left or right (Tinder card style), although currently this does not add the photos to the matches page. The profile page contains a stock profile photo and a couple buttons which will be implemented later when integrated with the backend. The stock photo will be customizable by the user.

For our first implementation of the image classifier, we are using a pretrained ResNet18 model with a modified final layer to classify the images as likes or dislikes. Starting with a collection of unrated images, we have functions to label images, move them into the appropriate folder, and recompile the dataset. Using this dataset of liked and disliked photos, we can train the model. With the trained model, there is a final function which takes an image name as input, finds the image in the unrated folder, and predicts if the image will be a like or dislike. In addition, there are convenience functions for printing samples from each dataset, as well as printing an unrated image with its predicted rating.

Feature Changes

For the image classifier, our original proposal stated we would use a pretrained DenseNet or ResNeXt model. After some testing, we found that a simpler ResNet18 model provided similar accuracy in a much shorter time. Since the size of our dataset is proportional to the number of hikes the user has seen, and thus relatively small, it is very easy for our model to overfit; to compensate for this, we are using a simpler model and plenty of data augmentation.

As mentioned in the last milestone report, we were looking into the differences in Django and Flask for building our API. Based on our findings, we decided to use Flask. We chose Flask since it is more lightweight, and a lot of the features associated with Django were not required for our project. It is easier to customize our API around and does not require a lot of overhead to get running. Finally, Flask offers more flexibility in adding libraries and is more user-friendly for beginner developers and startup projects.

Current Challenges

We were unable to create a working implementation of the recommender system. We wanted to use a content-based recommender system that did not rely on other user reviews, as it was difficult to obtain a list of users from AllTrails. After careful consideration, we believe it would be best to shift back to a collaborative filtering recommender system, which will require us to scrape AllTrails for user reviews. We are going to shift the collection of user-reviews and creation of the recommender system to M3.

Team Contributions and Upcoming Tasks

Andrew Forde

Contributions: First implementation of image classifier: creating dataset based on user ratings, training model, and functions for classifying and displaying unrated images.
In-progress: Optimizing image classifier model.

Taylor Mcouat

Contributions: Basic API backend that allows hike querying, database creation, and merging of photo data and hike data.
In-progress: Recommender system model, integration of ML model with backend.

Leo Sun

Contributions: Research on Flask vs Django, initial Flask-Restful exploration.
In-progress: API backend Prototype.

Jayden Wong

Contributions: App navigation, grid view for hike match images, primitive profile screen, tinder-style image swiping.
In-progress: Adding features to profile page.