Introduction

We are interested in generating phenotypically accurate images of species of various animal types. Our main model will be a conditional GAN (CGAN) designed to take as input a species name and output a realistic image of that species. We plan to train our CGAN with a custom loss function based on a separately trained species classification model. If the classifier model can correctly identify the CGAN generated images, loss will be low, and vice versa. We plan to train on a number of different animal types in the same model to test performance on that more difficult task. Our goal is for the final model to be able to produce images that a human can recognize as matching the desired species.

Challenges

Pre-processing: The datasets we found at first on the surface seemed to be a great collection of images, but because of how they were organized, we needed to pivot to new datasets.

Architecture: We haven't faced major challenges in architecture beyond the difficulty of finding an architecture setup that trains effectively.

Training: *Classifier:* Ideally the classifier would perform better than 75%, especially since we use it in the custom loss function, but 75% is still good performance on hundreds of bird and butterfly species.

CGAN: Training the CGAN has been difficult so far. We started with an architecture that used Conv2DTranspose, but our outputs were nonsensical and had strange grid patterns. To remove the grid patterns, we switched to UpSampling2D. We also applied Wasserstein loss to improve stability. Even after those changes, outputs have only minimal relation to birds and butterflies, and no discernable connection to the actual species the model is supposed to generate. To improve performance, we also expanded the size of the generator model, but that introduced a performance bottleneck. Training now takes about 4 minutes per epoch on OSCAR, which isn't terrible but does make it difficult to assess the effectiveness of architecture tweaks when we train for 20+ epochs.

Insights/Progress

Pre-processing: Here is the dataset for bird images, and here is the dataset for butterfly images. For preprocessing, we wanted all images to be the exact same dimensions. All the butterfly images were square, and using annotations included in the bird dataset, we were able to crop all bird images to squares, ensuring that the full bird would be included in the resulting image. We then downscaled all images to 128x128, as we only have around 13k images, and a higher resolution would make training difficult. Finally, we combined all images into one folder, and created a csv that has rows for filenames, species label, species id #, and category (bird or butterfly).

Architecture:

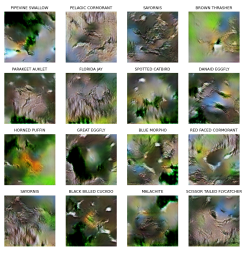
Classifier: The classifier is a fine-tuning of Google's EfficientNetB0 model. We essentially put an average pool and a large dense layer on top of the model and trained those weights to recognize bird/butterfly features from the general image recognition abilities that EfficientNetB0 is trained for.

CGAN: The CGAN is a variation of a traditional GAN architecture that begins by converting labels from the input text into a low-dimensional embedding representing the specific animal species the GAN should generate. This low-dimensional embedding is concatenated with a 128-channel random noise vector as the input. The generator then uses upsampling and convolution layers to produce a 128 by 128 image with 3 channels per pixel, while the discriminator uses convolution layers to decrease the dimensions back to the embedding dimension.

Training:

Classifier: Training the classifier was fairly straightforward. We fed the images and labels into it, ran it for 20 epochs and got a little over 75% accuracy. Improvement plateaued after about epoch 5. We saved the weights of our best performing model for use in the loss function.

CGAN: We've had difficulty getting good CGAN performance as described above, but we have produced images with bird/butterfly-like colors and shapes. We think that means our model is beginning to learn something, even if it's not as far as we'd like.



Full CGAN after 30 epochs

bird/butterfly CGAN after 10 epochs

Plan

We think it should be possible to get at least recognizable output from the CGAN. For the last week of the project, our primary focus will be on improving the performance. We wrote a simpler version that just asks the CGAN to generate a bird or a butterfly instead of a specific species, and we plan to implement various other architecture changes to get the best possible output.