Joseph Baruch

CS 212: Practical Python

Project 2: Image Classification

December 15, 2023

Biome Classification Model

Introduction

Using Ultralytics’ base model *yolov8n-cls,* I trained a model to classify different biomes including coniferous forest, desert, grassland, rain forests and tundra. Motivated to train a classification model beyond objects, an environment provides more variation which could lead to a greater challenge at creating an affective model. For an additional challenge and greater insight, I restricted my training to only changing my hyperparameter epoch count and data size in-between models.

Methods

Before discussing the different challenges and training variations/values, I believe it is important to discuss the general flow of my program. This overview will walk through the stages from data to testing the final models. Note: This overview will not discuss specific variables, libraries included or other minor details. First, the program starts by resizing all the photos in the data set. This is done simpily by traversing the file structure with for loops. Next, a function called augment\_and\_add() is call which traverses the file structure the same way by augmenting the photos and saving a copy of the augmentation. This process doubles the size of the data. The augmentation pipeline is predetermined and applied for each image. Lastly, using the prepared data, the model is trained using the .train() method from Ultralytics. Once, the model finishes training, a file named *test.py* iterates (using a for loop) over the test images and output the results to the terminal.

The challenges I faced during this process can be broken down into two different categories: program operation and training refinement.

First, the program operation challenges are main problems I needed to overcome to allow the program to run properly. Although, this program isn’t extremely advanced, learning new software and programs for python is a relatively new concept for me. The main challenge was getting my dependencies to properly work with my preferred IDE, Visual Studio Code. Using different pip commands and searching on the internet for more time then I would have liked, lead to me importing Ultralytics using a Github model rather than the Ultralytics’ personal specific module. After this point, the only other challenge with VSCode was converting to two standard ‘.py’ files rather than one python notebook. The other main program operation challenge I faced was using Albumentations. I was completely unfamiliar with this library, so I need to learn it from scratch. This took the form on figuring out how to pick a good augmentation pipeline for my data. After, the other challenge was needing to restructure my path values to my files. Since Albumentations deals with cv2, I need the full file path and not a relative file path to the images I was going to augment. This is something that I didn’t initially account for. Lucky, this was only changing string values. Generally, this was the extend of the main program operation challenges that I faced.

Secondly, there were challenges I faced while trying to refine my model through training. As I previously discussed, I wanted to limit myself to only changing the amount of data I had through augmenting the existing data set and changing the number of epochs I had. Although, I will reserve the discussing on how well my different models performed to the conclusion, I will discuss the general pathway I took to finding my best model. Generally, I only tried to change one variable at a time, either adding more epochs or images. I first started with a low number of epochs, then kept the epoch count constant while changing the number of images. Lastly, I wanted to test to see what changing the epoch count and images at the same time, which lead to some unexpected results. Overall, the challenges was dealing with the compromises of increasing the epochs or images. When increasing these, it adds a substantial amount of time to training and might benefit results.

Lastly, it is important to discuss the different values in my program although most of these can be found in the source code. I used an 80-10-10 (train-test-val) structure on my data, I believed this would be a good option because it allows for a great amount of training. While using Albumentations to augment my images, I used 3-4 different alterations in my pipeline. First, the images would be randomly cropped to 512 x 512, then horizontally flipped, then the brightness would be reduced and then the image would be rotated. Lastly, I used the yolov8n-cls model from Ultralytics to train an image classification model to tell the difference between five different classes: coniferous forests, desert, grassland, rain forest and tundra.

Results

I trained four different models while adjusting the epoch count or number of images in the training by using augmentations. I tested the same images every time to keep consistency. The images are below followed by the data from every model:

Coniferous Forest

A mountain with trees and a clear sky

Description automatically generated

Desert

A desert landscape with blue sky and clouds

Description automatically generated

Grassland

A green hills with grass

Description automatically generated

Rain Forest

A close-up of a forest

Description automatically generated

Tundra

A snowy mountain range with blue sky

Description automatically generated

Model 1

Epoch:

Number of images (per class):

Augmentation:

Data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Coniferous Prediction Confidence(%) | Desert Prediction Confidence(%) | Grassland Prediction Confidence(%) | Rain Forest Prediction Confidence(%) | Tundra Prediction Confidence(%) |
| Coniferous Forest Image |  |  |  |  |  |
| Desert Image |  |  |  |  |  |
| Grassland Image |  |  |  |  |  |
| Rain Forest Image |  |  |  |  |  |
| Tundra Image |  |  |  |  |  |

Model 2

Epoch:

Number of images (per class):

Augmentation:

Data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Coniferous Prediction Confidence(%) | Desert Prediction Confidence(%) | Grassland Prediction Confidence(%) | Rain Forest Prediction Confidence(%) | Tundra Prediction Confidence(%) |
| Coniferous Forest Image |  |  |  |  |  |
| Desert Image |  |  |  |  |  |
| Grassland Image |  |  |  |  |  |
| Rain Forest Image |  |  |  |  |  |
| Tundra Image |  |  |  |  |  |

Model 3

Epoch:

Number of images (per class):

Augmentation:

Data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Coniferous Prediction Confidence(%) | Desert Prediction Confidence(%) | Grassland Prediction Confidence(%) | Rain Forest Prediction Confidence(%) | Tundra Prediction Confidence(%) |
| Coniferous Forest Image |  |  |  |  |  |
| Desert Image |  |  |  |  |  |
| Grassland Image |  |  |  |  |  |
| Rain Forest Image |  |  |  |  |  |
| Tundra Image |  |  |  |  |  |

Model 4

Epoch:

Number of images (per class):

Augmentation:

Data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Coniferous Prediction Confidence(%) | Desert Prediction Confidence(%) | Grassland Prediction Confidence(%) | Rain Forest Prediction Confidence(%) | Tundra Prediction Confidence(%) |
| Coniferous Forest Image |  |  |  |  |  |
| Desert Image |  |  |  |  |  |
| Grassland Image |  |  |  |  |  |
| Rain Forest Image |  |  |  |  |  |
| Tundra Image |  |  |  |  |  |

Conclusion

A discussion that explains what you learned from this project, general insight about what your model can be used for, ways that your model could be improved, and anything else that you would like to mention.

* Decided to see what model could predict a set of images I prodided (above) with greatest accuracy but this could be different for images of these biomes as a whole. This is something that I might want to change in a different machine learning project.
* Note: Model 1-3 resulted in error
* Changing allowing more hyperparameters
* More epochs can lead to a more accurate model but it can take more time.
* Augmentating images too much or augmenting images too much and over train a model. By the time I augmented images for a second time, I had more augmented images then original images.
* Best model was 100% confident in predicting grassland but thought desert was also grassland all times.
  + When increasing epochs, predicting desert still resulted in grassland but desert was the second more confident option.
  + May because images provided had open blue sky (not blocked by many mountains, or trees) just like grassland and deserts have ( but in different ways).
* Model 6 predicted coniferous correctly but rain\_forest was close behind. This is because they may seem very similar.
  + This isn’t the same for rain\_forest though. The model predicted rainforests with 100% accuracy.