Crop Species Image Classification

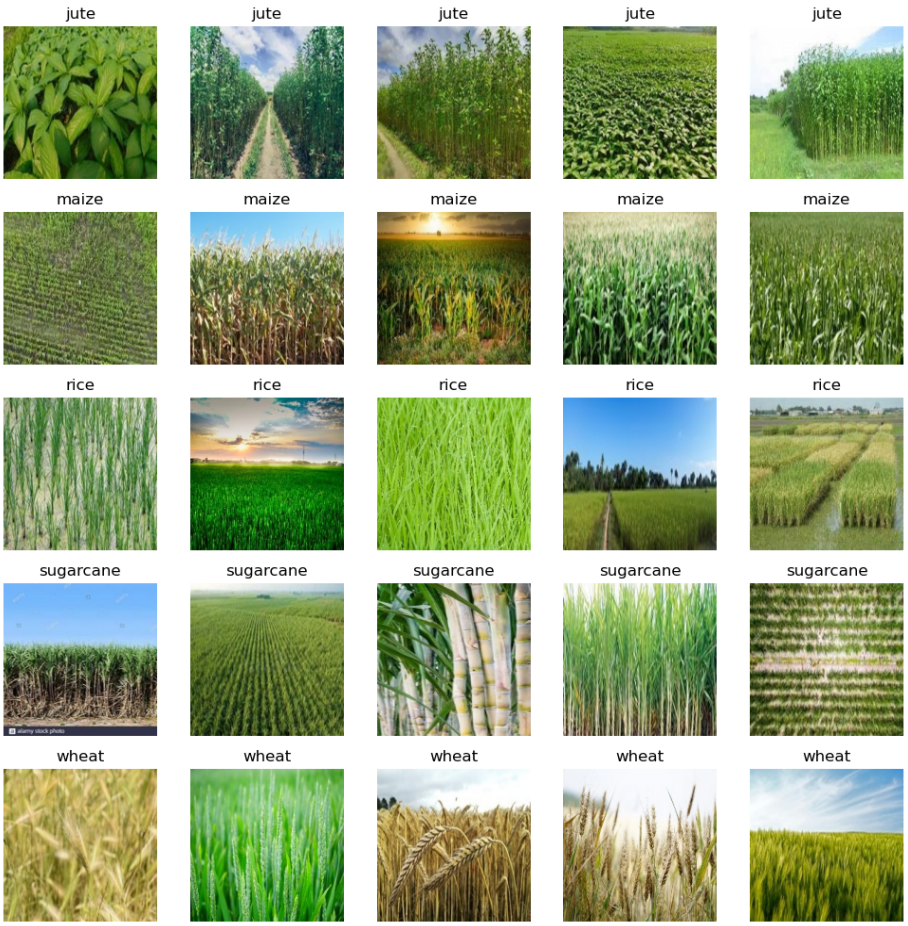
As Machine Learning advances, areas such as agriculture have greatly increased in productivity. New techniques have allowed them to detect soil quality, monitor crop health, and improve irrigation systems. With image classification for different species of crops, cities/farmers can better track the data of crops on their land which can be used for census data, detecting misplaced crops inside of fields, and potential crop damage.

1. Data

The images come from google images that were taken and combined together for kaggle use. The files can be found here:

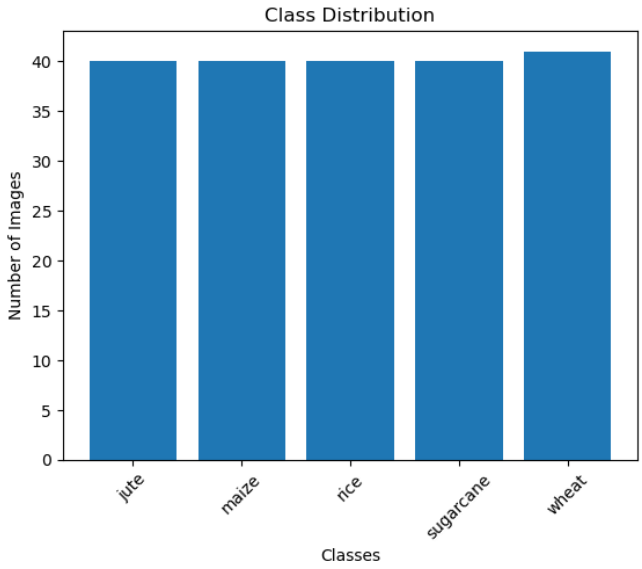
https://www.kaggle.com/datasets/aman2000jaiswal/agriculture-crop-images

It contains 201 test images and 45 test images each containing an equal number of images from the 5 different classes which are rice, maize, sugarcane, wheat, and jute.

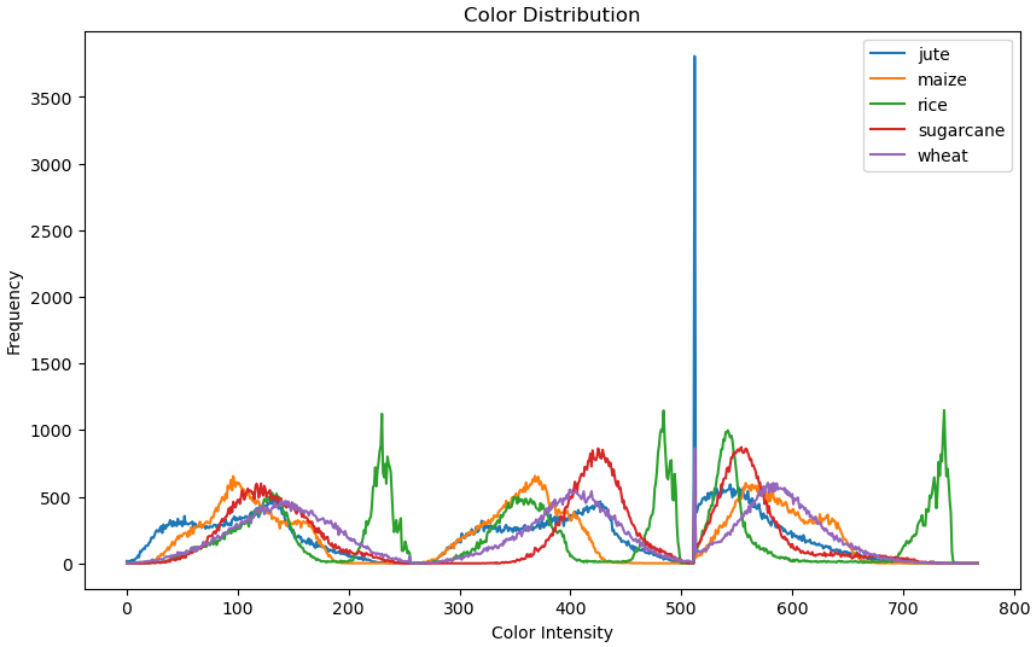


2. Cleaning and EDA

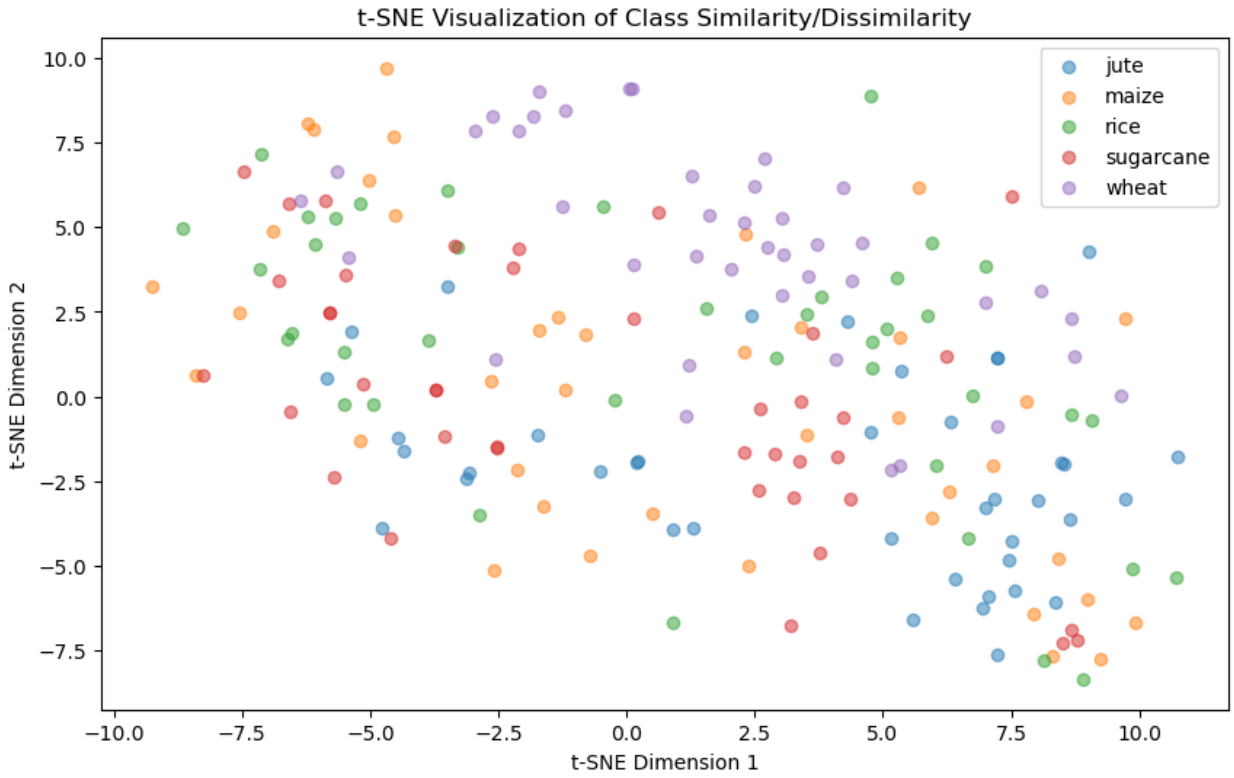
The images were loaded into a jupyter notebook for preprocessing. I made sure that the class distributions were closely equal, which they were with there only being one more image of wheat than the other classes.



Below is a color distribution of three images from each class stacked on the x-axis. 0-255 would be the first images, 256-511 the second group, and 512-766 the third group. All of the classes tend to peak around the middle of the intensity which makes sense they all contain mostly green pixels. Some of the images show a lot more of the sky in the background which is why there are spikes along the 230, 500, and 730 marks. The jute images tend to have more shadows in them because of the shape of the plant which is why there is a spike around 510.



The t-SNE reduction doesn't show too much of a separation of the classes. This can be difficult with images due to their complex nature.



3. Modeling

I chose to use neural networks for the modeling part of this project. I started by making my own layer by layer, but later learned that it was overfitting due to the scores on the training data being really high but the test data being really low. I then learned about transfer learning which tends to be a lot more effective since you are downloading weights from a model that is very heavily trained on lots of images. I decided to use imagenet weights from the Xception model.

```
base_model = keras.applications.Xception(
    weights='imagenet',  # Load weights pre-trained on ImageNet.
    input_shape=(256, 256, 3),
    include_top=False)  # Do not include the ImageNet classifier at the top.
```

I froze all the layers and added a pooling and dense layer so that the model could use my classes instead of the ones that came from the model. The results are as follows.

```
Epoch 20/20
7/7 [==============================] - 10s 1s/step - loss: 0.7525 - categorical_accuracy: 0.8010 - val_loss:
0.8126 - val_categorical_accuracy: 0.8000
```

: <keras.src.callbacks.History at 0x29a049690>

```
evaluation_results = model.evaluate(test_generator)
```

```
2/2 [==============================] - 2s 511ms/step - loss: 1.2966 - categorical_accuracy: 0.5333
```

The training score is 80% and the testing score is 53.3%.

I then unfroze all the layers of the model we downloaded and added a very small training rate so that the model could improve without changing the original weights too much.

```
Epoch 10/10
7/7 [==============================] - 46s 6s/step - loss: 0.2473 - categorical_accuracy: 0.9254 - val_loss:
0.2648 - val_categorical_accuracy: 0.9000
```

: <keras.src.callbacks.History at 0x292d75e70>

```
evaluation_results = model.evaluate(test_generator)
```

```
2/2 [==============================] - 2s 577ms/step - loss: 1.2952 - categorical_accuracy: 0.6000
```

The Training score increased to 92% and the test score increased to 60%.

4. Future Improvements

To improve this model, more images could be used such as images of a single crop, many more angles of the crop, crops with different diseases. This could improve the training set for the model used in this project, or could be used with other image classification models that use more parameters and tend to provide slightly better accuracy. Different training rates could be used to see if it would improve the testing accuracy without overfitting.