Plant Seedlings Classification

DL-International (AI with Deep Learning- Oct'24)

Date: 12/22/24

Business Problem Overview and Solution Approach

In recent times, the field of agriculture has been in urgent need of modernizing, since the amount of manual work people need to put in to check if plants are growing correctly is still highly extensive. Despite several advances in agricultural technology, people working in the agricultural industry still need to have the ability to sort and recognize different plants and weeds, which takes a lot of time and effort in the long term. The potential is ripe for this trillion-dollar industry to be greatly impacted by technological innovations that cut down on the requirement for manual labor, and this is where Artificial Intelligence can benefit the workers in this field, *as the time and energy required to identify plant seedlings will be greatly shortened by the use of AI and Deep Learning*.

Our objective is to build a Convolutional Neural Network model which would classify an example dataset of plant seedlings into their respective 12 categories.

Load Dataset and Summary

After printing the shape of the images and labels dataset, we are given the following results for images and labels respectively,

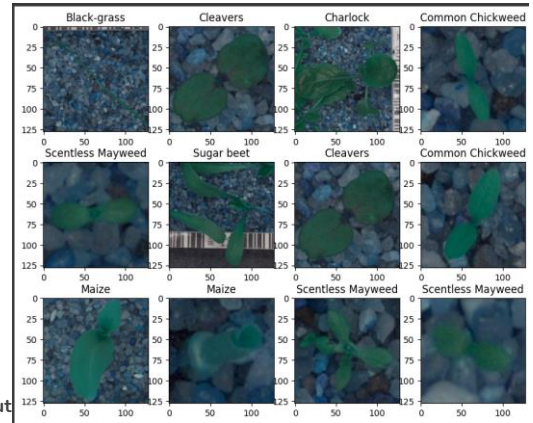
(4750, 128, 128, 3)

(4750, 1)

The result of the shape for the images dataset tells us 3 important things. There are **4,750 images**, their dimensions being **128 x 128**, and they also have **3 color channels**, the default format being a BGR format.

The results of the shape for the labels dataset tell us that there are **4,750 label entries** and that the entries are a **single value**.

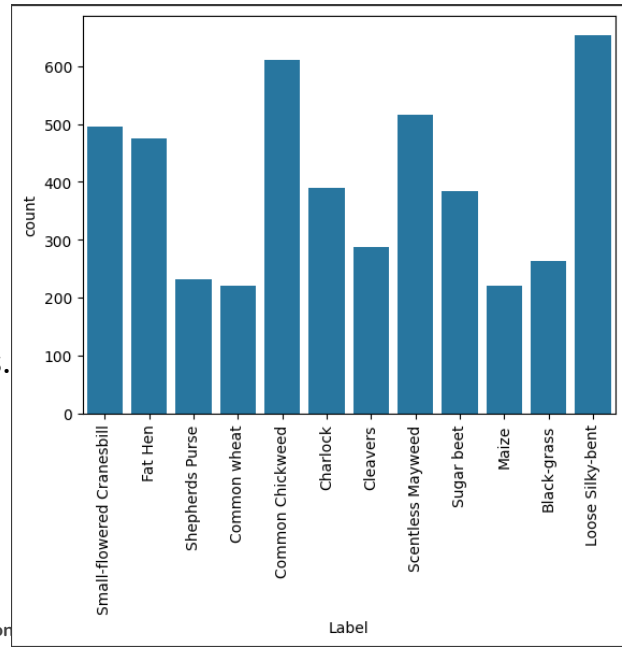
The image represents the result of plotting the images with their labels:



Additional EDA

The following image represents a count plot of the plotted images with the data.

- As we can see, there is a noticeable imbalance in the dataset. One such example of this imbalance is that Loose Silky-bent has over 600 samples, while Shepherds Purse and Black-grass have fewer than 250 samples.
- Loose Silky-Bent and Common Chickweed both consist of more than 600 samples, making them majority classes within the data.
- Shepherds Purse, Common wheat, Maize, and Black Grass consist of 250 or less samples, making them minority classes within the Data.
- As a result of class imbalance, there are possible issues of bias towards majority classes and underperformance on minority classes.



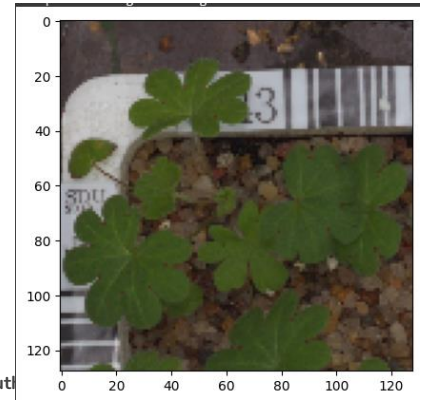
Data Preprocessing

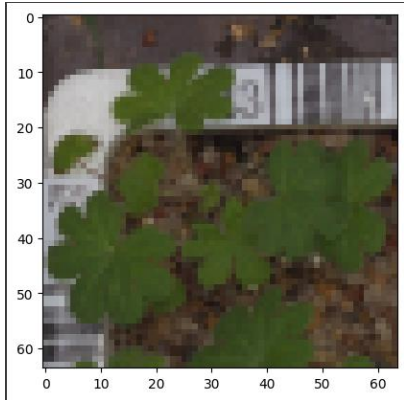
In preprocessing the data so that it is a more compatible input in our model, we:

- Converted the color channel format of the images from BGR to RGB. Images from OpenCV are loaded in as BGR format and it is helpful to convert it because most image processing libraries require images to have an RGB format.
- Images were compressed from its original 128 x 128 dimensions to 64 x 64 dimensions in order to make the training process less computationally expensive. The before and after is seen below.
- The train-test-split for the dataset is 80%-10%-10% respectively. 3800 data entries are being utilized for training and 475 entries are being used for testing and validation each.
- When we do .shape on the train, validation, and test sets, we are given the following results:

```
(3800, 64, 64, 3) (3800,)
(475, 64, 64, 3) (475,)
(475, 64, 64, 3) (475,)
```

- The first value in the set represents the number of data entries in the set, the next two values represent the dimensions of the new compressed images, and the fourth value represents the number of color channels.





Making the Data Compatible

- We used LabelBinarizer in order to encode each unique label with its own binary vector. Helpful for multi-class classification.

```
((3800, 12), (475, 12), (475, 12))
```

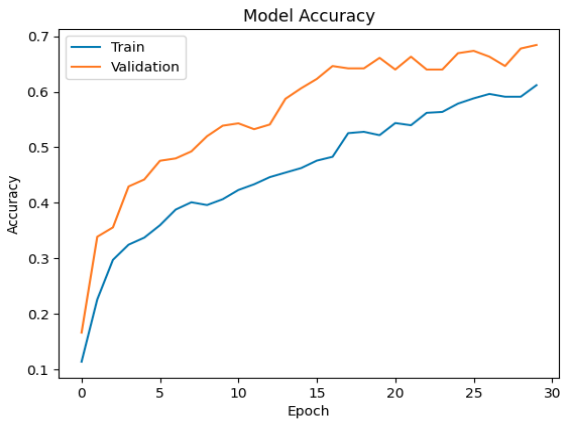
- We are given the above result after finding the shape of the train-test-validation splits after encoding the labels. It shows the amount of data entries in each set as well as that 12 unique binary vectors were created for the 12 unique labels that there are for the plant seedlings.

- The pixel values of the images range between 0-255, so we normalize the data by dividing all the pixel values by 255 in order to standardize the values between 0-1.

Model1 Summary and Performance

1. **Model Type**:
   - The model is a **Sequential CNN**, meaning layers are stacked sequentially, with each layer feeding into the next.
2. **Input Layer**:
   - **Input Shape**: `(64, 64, 3)`
   - This layer takes images of size 64x64 with 3 color channels (RGB).
3. **Convolutional and Pooling Layers**:
   - **1st Convolutional Block**:
     - 128 filters, kernel size `(3, 3)`, activation = `ReLU`, padding = `same`.
     - Followed by a **MaxPooling** layer with pool size `(2, 2)`.
   - **2nd Convolutional Block**:
     - 64 filters, kernel size `(3, 3)`, activation = `ReLU`, padding = `same`.
     - Followed by a **MaxPooling** layer with pool size `(2, 2)`.
   - **3rd Convolutional Block**:
     - 32 filters, kernel size `(3, 3)`, activation = `ReLU`, padding = `same`.
     - Followed by a **MaxPooling** layer with pool size `(2, 2)`.
4. **Flatten Layer**:
   - Converts the 2D feature maps from the final pooling layer into a 1D vector to feed into the fully connected layers.
5. **Fully Connected Layers (Dense Layers)**:
   - **First Dense Layer**:
     - 16 neurons, activation = `ReLU`.
   - **Dropout Layer**:
     - Dropout rate = `0.3` to prevent overfitting by randomly deactivating 30% of neurons during training.
   - **Output Layer**:
     - 12 neurons, activation = `softmax` (suitable for multi-class classification, where each neuron corresponds to a class).
6. **Optimizer and Loss**:
   - **Optimizer**: Adam (adaptive learning rate optimization).
   - **Loss Function**: Categorical Crossentropy (used for multi-class classification problems).
   - **Evaluation Metric**: Accuracy.

- Final Training Accuracy: .6118
- Final Validation Accuracy: .6842
- Loss decreases consistently throughout the training of the model.
- The higher validation accuracy leads us to believe that the model performs well on unseen validation data.
- We should try fine-tuning the model and play around with learning rates, batch sizes, normalization, etc.. We should also try augmenting the dataset.



Confusion Matrix Analysis

The confusion matrix indicates that there is significant class imbalance that is present within the data set and this is affecting the results of the model prediction.

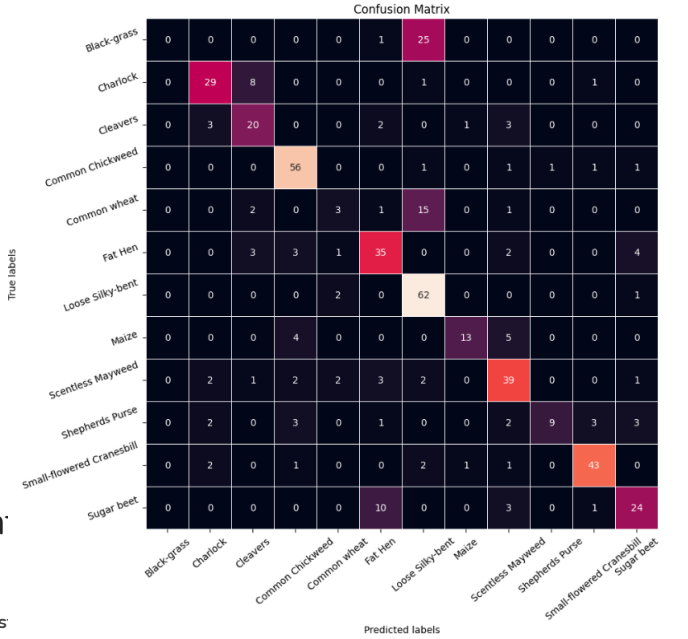
Strong performances were seen for the following:

- Common Chickweed: 56 samples correctly classified.
- Loose Silky-bent: 62 samples correctly classified.
- Small-flowered Cranesbill: 43 samples correctly classified.

Poor performances were for the following:

- Black-grass: 25 samples were incorrectly classified as another class.
- Charlock: 8 samples were misclassified into other classes.
- Common Wheat: A significant number of samples (15) were misclassified into another class.

Solution: We will augment the data to increase the variability so tha
Better accuracy for underperforming class



Model2 Summary and Performance

The model is a **Sequential CNN**, meaning layers are stacked sequentially, with each layer feeding into the next.

1. Input Layer

- **Input Shape**: `(64, 64, 3)` (images resized to 64x64 pixels with 3 color channels for RGB).

2. Convolutional and Pooling Layers

- **First Convolutional Block**:
  - **Convolution Layer**: 64 filters, kernel size `(3, 3)`, activation = ReLU, padding = `same`.
  - **MaxPooling Layer**: Pool size `(2, 2)`, reduces spatial dimensions by half.
- **Second Convolutional Block**:
  - **Convolution Layer**: 32 filters, kernel size `(3, 3)`, activation = ReLU, padding = `same`.
  - **MaxPooling Layer**: Pool size `(2, 2)`, further reduces spatial dimensions.
  - **Batch Normalization**: Normalizes activations, stabilizing and accelerating training.

3. Flatten Layer

- Converts the 2D feature maps into a 1D vector to connect to the dense layers.

4. Fully Connected Layers

- **Dense Layer**: 16 neurons, activation = ReLU.
- **Dropout Layer**: Dropout rate = `0.3` to prevent overfitting.
- **Output Layer**: 12 neurons, activation = `softmax` (for 12 classes, predicting class probabilities).

- **Optimizer**: Adam optimizer
- **Loss Function**: Categorical crossentropy
- **Evaluation Metric**: Accuracy.

- Final Training Accuracy: .7344
- Final Validation Accuracy: .7600
- Loss decreases consistently throughout the training of the model.
- The higher validation accuracy leads us to believe that the model performs well on unseen validation data.
- As seen by the results, both the validation and training accuracy's are significantly better.

Confusion Matrix Analysis

The confusion matrix for Model 2 indicates a significant improvement in classification after augmenting the dataset by rotating images by 20 degrees.

- Model 2 shows improvements in diagonal entries for several classes, indicating better classification accuracy for these classes.

Example:

- **Charlock**: Increased from **29** to **36** true positives.
- **Scentless Mayweed**: Increased from **39** to **45** true positives.
- **Loose Silky-bent**: Performance declined slightly, with true positives dropping from **62** to **40**.

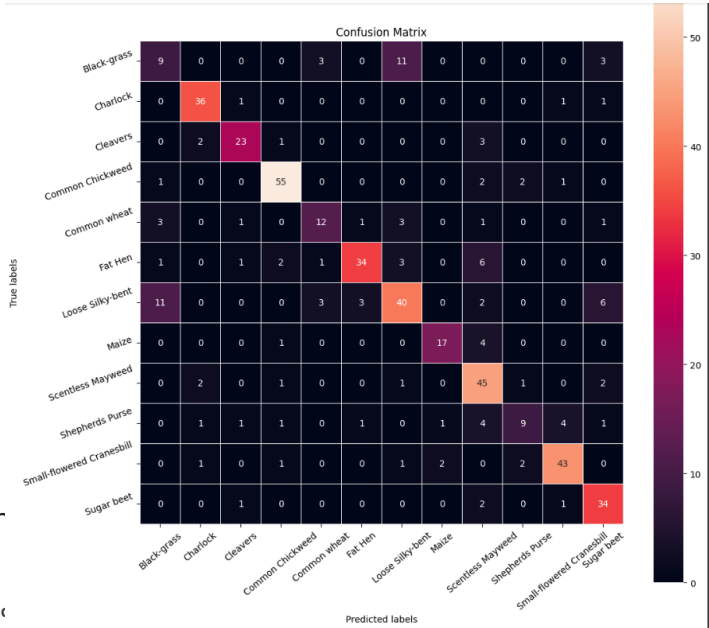
Consistency:

- Classes like **Common Chickweed** maintain high accuracy with **56** (Model 1) and **55** (Model 2) true positives.

Reduction in Misclassifications:

- **Black-grass**: Misclassifications reduced from **25** to **17**.
- **Charlock**: Misclassifications significantly reduced for other classes.
- **Fat Hen**: Misclassifications reduced slightly.

Improvement: Still has scope for improvement by providing more variation ir Dataset in order to improve class imbalance issues even more.



Conclusion and Recommendations

Key Takeaways

•Model 2 is superior to Model 1 due to its enhanced architecture, improved generalization, and better handling of misclassifications.
•While significant progress has been made, further efforts are needed to address class-specific weaknesses, particularly for visually similar and underrepresented classes.
•These models, with further refinements, show great promise for practical applications like automated plant seedling classification in agriculture.

Recommendations

Address Class Imbalance:
•Use oversampling techniques like SMOTE or class weighting to give more importance to minority classes.
Enhance Data Augmentation:
•Expand augmentation strategies (e.g., zoom, shear, brightness adjustments) tailored to underperforming classes.
Experiment with Pre-Trained Models:
•Use pre-trained models (e.g., ResNet, EfficientNet) to leverage deeper feature extraction capabilities.