Course Name: Introduction to Computer Vision

Computer Vision Project - Case Study

Date: Wednesday, September 11, 2024

Student Name: Vijaya Laxmi Kumbaji

Contents / Agenda

Executive Summary

- The objective is to reduce the amount of time and efforts put in in order to identify the plant seedlings in agricultural industry there by eliminating or minimizing human efforts and automating the processs of identification.

- Deep learning artificial intelligence was leveraged to develop highly efficient intelligent models to perform the task. Multiple Convolutional neural network (model) were developed. The knowledge we gained is the more number of trainable parameters we hve, the more accurate the model performed. Two models were developed and one of them was finally selected to adopt in the agricultural industry. In order to create CNN models, raw / original images were converted from BGR to RGB and re-sized to 64 pixes from 128 pixels.

- Selected model accurately predicted the labels based on seedling images after fine tuning and fitting.

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 actually 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.**

- The ability to do so far more efficiently and even more effectively than experienced manual labor, could lead to better crop yields, the freeing up of human involvement for higher-order agricultural decision making, and in the long term will result in more sustainable environmental practices in agriculture as well.

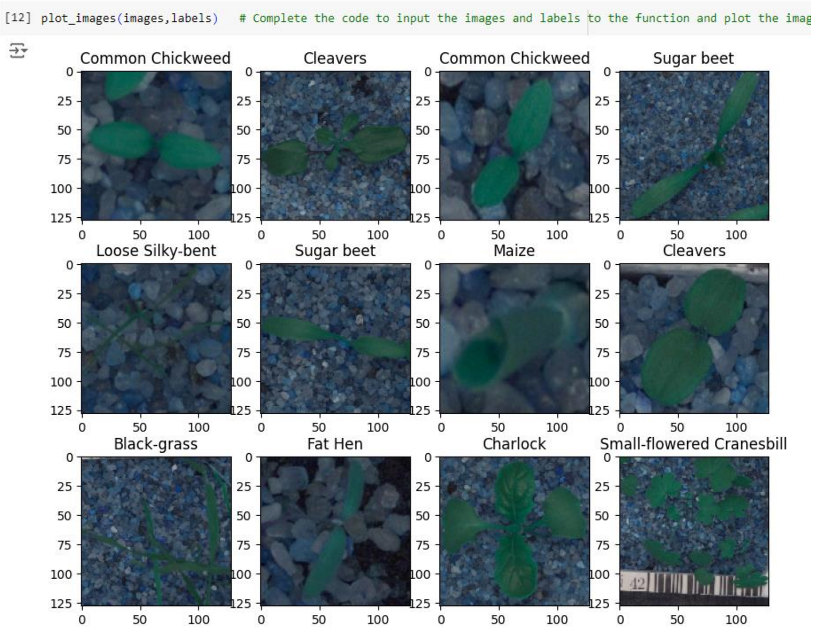
- The aim of this project is to Build a Convolutional Neural Network to classify plant seedlings into their respective categories.

Data Dictionary

- The data dictionary for this case study contains two data entities, one of them is for images of plants and is called images.npy. The second data entity is called Labels.csv.
- Images file has images / photos of the following plant species:
- black-grass
- Charlock
- Cleavers
- Common chickweed
- Common wheat
- Fat hen
- Loose silky-bent
- Maize
- Scentless mayweed
- Shepherd’s purse
- Small-flowered cranesbill
- Sugar beet
- Dataset size:
- **Labels: dataset size: 4750 rows of labels.**
- **Images: dataset size : (4750, 128,128,3)**

EDA Results

- From each species of the plants, one photo / image was selected randomly and the images were shown as below in a N X M matrix with labels and photos:



EDA Results continued…

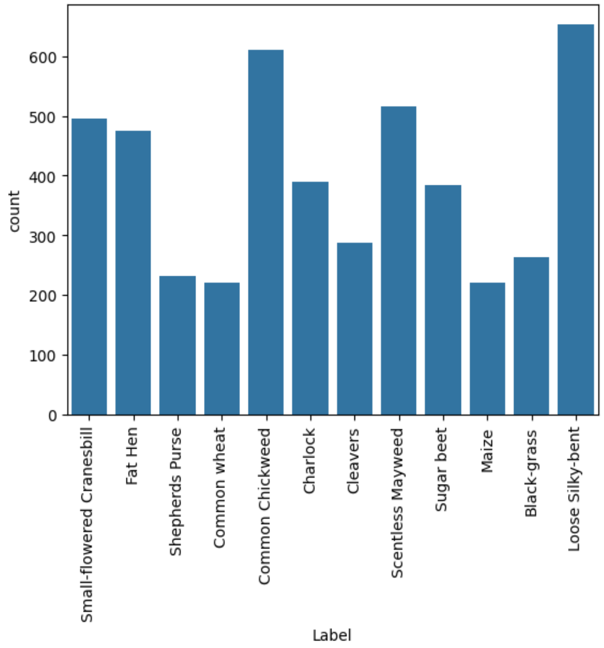
Distribution of the target variable (label) was plotted .top 3 species with highest number of occurrence were:

species ”Loose silky-bent” > 600, Common chick-weed >= 600,

scentless mayweed > 500 & < 600.

```python
sns.countplot(x=labels['Label'])
plt.xticks(rotation='vertical')
```

```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11],
 [Text(0, 0, 'Small-flowered Cranesbill'),
  Text(1, 0, 'Fat Hen'),
  Text(2, 0, 'Shepherds Purse'),
  Text(3, 0, 'Common wheat'),
  Text(4, 0, 'Common Chickweed'),
  Text(5, 0, 'Charlock'),
  Text(6, 0, 'Cleavers'),
  Text(7, 0, 'Scentless Mayweed'),
  Text(8, 0, 'Sugar beet'),
  Text(9, 0, 'Maize'),
  Text(10, 0, 'Black-grass'),
  Text(11, 0, 'Loose Silky-bent')])
```



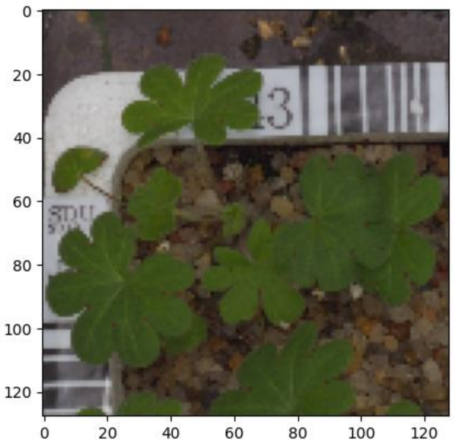
Data Preparation for model building

- Images were converted from BGR images to RGB images
- Resizing Images: images were re-sized and decreased from 128 to 64 pixels.

- Original image (before re-sizing), resized image(after resizing)

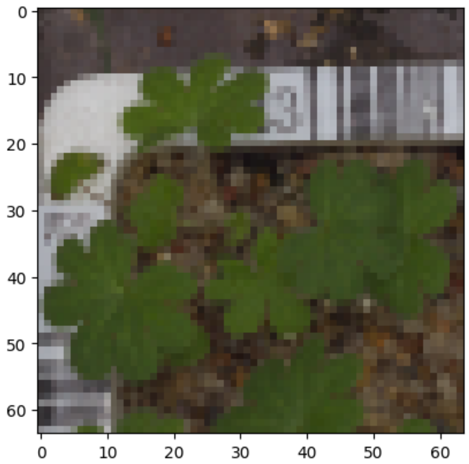
```
plt.imshow(images[3])
```

```
&lt;matplotlib.image.AxesImage at 0x7c7862fd4400&gt;
```




```
plt.imshow(images_decreased[3])
```

```
<matplotlib.image.AxesImage at 0x7c7862ebe680>
```



Data Preparation for model building continued...

- Before encoding the target variable the resized images dataset waas split into train, test and validation datasets.
- With 80%, 20% and 75% , 25% ratios. The resulting shape of the datasets was as follows:

```
[28] # Complete the code to check the shape of train, validation and test data

     print(X_train.shape, y_train.shape)  # Training data shapes
     print(X_val.shape, y_val.shape)      # Validation data shapes
     print(X_test.shape, y_test.shape)     # Test data shapes
```

```
(3847, 64, 64, 3) (3847,)
(428, 64, 64, 3) (428,)
(475, 64, 64, 3) (475,)
```

- Encoding was done using LabelBinarizer().
- The resulting shape of the train, test and validation datasets are as follow after encoding:

```
y_train_encoded.shape,y_val_encoded.shape,y_test_encoded.shape
 # Complete the code to check the shape of train, validation and test data
```

```
((3847, 12), (428, 12), (475, 12))
```

Data Preparation for model building continued...

- Data normalization:
- Image data was scaled as the image pixel values range from 0-255.
-

```
# Normalize the image pixels of train, test and validation data
X_train_normalized = X_train.astype('float32') / 255.0
X_val_normalized = X_val.astype('float32') / 255.0
X_test_normalized = X_test.astype('float32') / 255.0
```

- The shape of the normalized image data as follows:
-

```
print(X_train_normalized.shape)
print(X_val_normalized.shape)
print(X_test_normalized.shape)
```

```
(3847, 64, 64, 3)
(428, 64, 64, 3)
(475, 64, 64, 3)
```

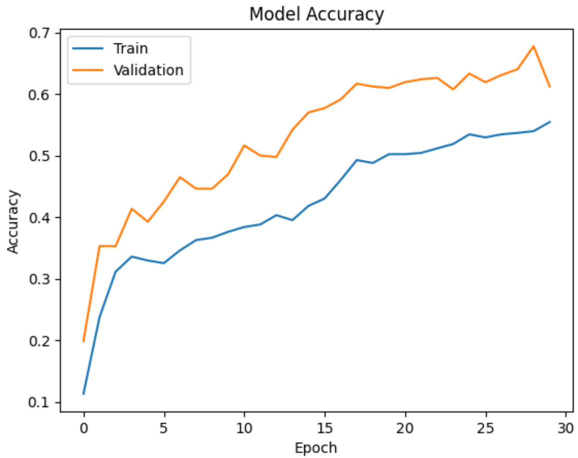
Model Building (model1)

- Sequential method was used to build model1 using Adam optimizer.
- The model1 was fit on the train normalized and encoded data for 30 epochs. . Highest validation accuracy was 58% with trainable parameters: 128,828.

Model: "sequential"

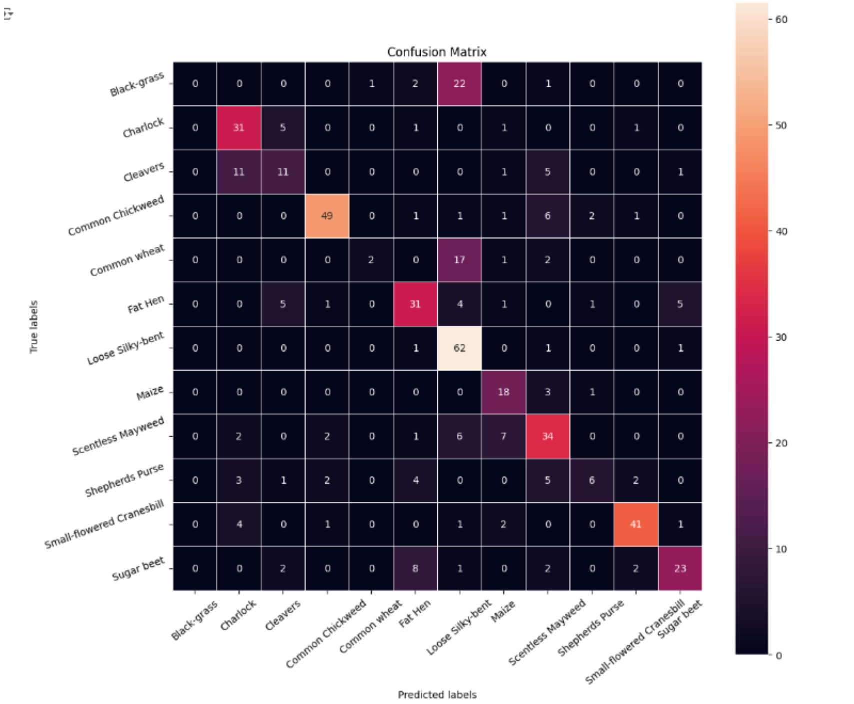
Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 64, 64, 128)	3,584
max_pooling2d (MaxPooling2D)	(None, 32, 32, 128)	0
conv2d_1 (Conv2D)	(None, 32, 32, 64)	73,792
max_pooling2d_1 (MaxPooling2D)	(None, 16, 16, 64)	0
conv2d_2 (Conv2D)	(None, 16, 16, 32)	18,464
max_pooling2d_2 (MaxPooling2D)	(None, 8, 8, 32)	0
flatten (Flatten)	(None, 2048)	0
dense (Dense)	(None, 16)	32,784
dropout (Dropout)	(None, 16)	0
dense_1 (Dense)	(None, 12)	204

Total params: 128,828 (503.23 KB)
Trainable params: 128,828 (503.23 KB)
Non-trainable params: 0 (0.00 B)



Model Building (model1) continued...

- Sequential method was used to build model1 using Adam optimizer.
- “Confusion matrix” with true labels and predicted labels and “Classification report” are below:



	precision	recall	f1-score	support
0	0.00	0.00	0.00	26
1	0.61	0.79	0.69	39
2	0.46	0.38	0.42	29
3	0.89	0.80	0.84	61
4	0.67	0.09	0.16	22
5	0.63	0.65	0.64	48
6	0.54	0.95	0.69	65
7	0.56	0.82	0.67	22
8	0.58	0.65	0.61	52
9	0.60	0.26	0.36	23
10	0.87	0.82	0.85	50
11	0.74	0.61	0.67	38
accuracy			0.65	475
macro avg	0.60	0.57	0.55	475
weighted avg	0.63	0.65	0.62	475

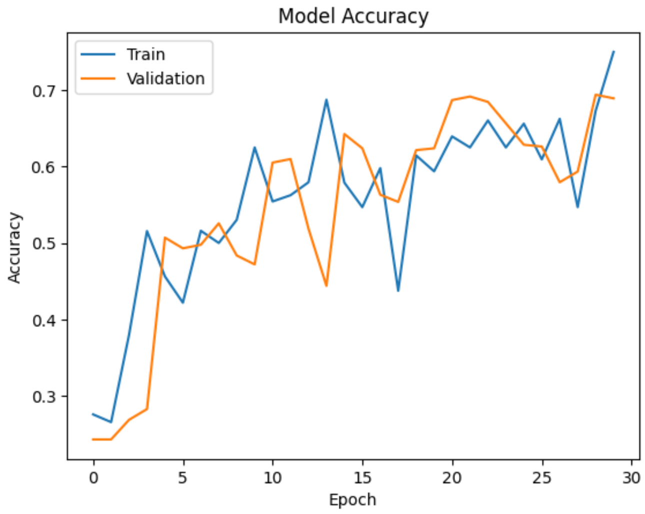
Model Building (model2)

- Sequential method was used to build model2 using Adam optimizer.
- The model2 was fit on the train normalized and encoded data for 30 epochs. . Highest validation accuracy was 69% with number of trainable parameters: 151,612 parameters.

Model: "sequential"

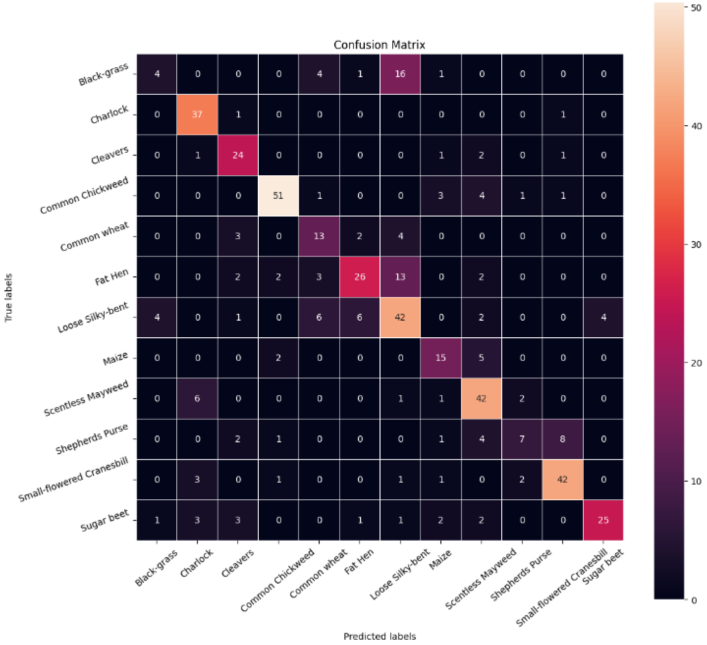
Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 64, 64, 64)	1,792
max_pooling2d (MaxPooling2D)	(None, 32, 32, 64)	0
conv2d_1 (Conv2D)	(None, 32, 32, 32)	18,464
max_pooling2d_1 (MaxPooling2D)	(None, 16, 16, 32)	0
batch_normalization (BatchNormalization)	(None, 16, 16, 32)	128
flatten (Flatten)	(None, 8192)	0
dense (Dense)	(None, 16)	131,088
dropout (Dropout)	(None, 16)	0
dense_1 (Dense)	(None, 12)	204

Total params: 151,676 (592.48 KB)
Trainable params: 151,612 (592.23 KB)
Non-trainable params: 64 (256.00 B)



Model Building (model2) continued...

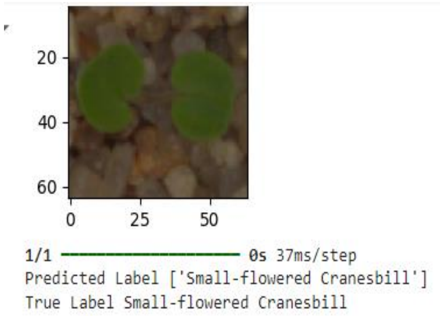
- Sequential method was used to build model2 using Adam optimizer.
- “Confusion matrix” with true labels and predicted labels and “Classification report” are below:

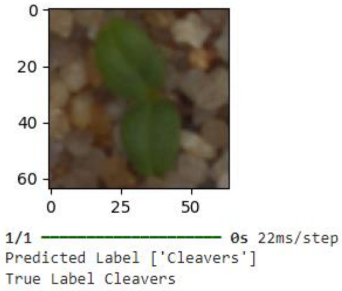


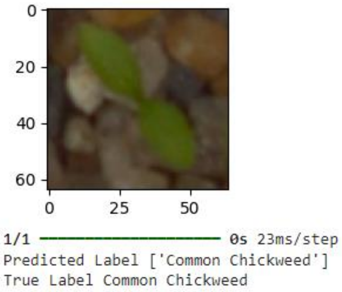
	precision	recall	f1-score	support
0	0.00	0.00	0.00	26
1	0.61	0.79	0.69	39
2	0.46	0.38	0.42	29
3	0.89	0.80	0.84	61
4	0.67	0.09	0.16	22
5	0.63	0.65	0.64	48
6	0.54	0.95	0.69	65
7	0.56	0.82	0.67	22
8	0.58	0.65	0.61	52
9	0.60	0.26	0.36	23
10	0.87	0.82	0.85	50
11	0.74	0.61	0.67	38
accuracy			0.65	475
macro avg	0.60	0.57	0.55	475
weighted avg	0.63	0.65	0.62	475

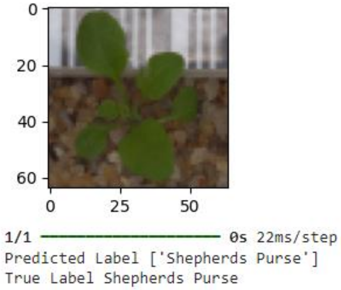
Model Performance Summary

- Summary of the final model for prediction: final selected model is model2. It demonstrated a performance of 100% accuracy on :Visualizing the prediction” where True labels matched the predicted labels one 100%.









Model Performance Summary continued...

- Summary of the final model for prediction: final selected model is model2. It performed 100% accuracy on :Visualizing the prediction” where True labels matched the predicted labels 100%. Model2 performance was better on all metrics.

Model 1 / model2	Test dataset	Train dataset	Validation dataset	Number of trainable params
model1	64.84%	58.89%	66.36%	128,828
model2	69.04%	75%	72.66%	151612

Conclusion – Actionable insights and Business recommendations

- Summary of the final model for prediction: final selected model is model2. It performed 100% accuracy on :Visualizing the prediction" where True labels matched the predicted labels 100%
- The model2 was the final models with more number of trainable parameters and better accuracy in both test dataset and validation dataset. Hence model2 is chosen to help predict the labels from images of plant species.

APPENDIX

Happy Learning !