Weed Classification

Group project for CS3244

Presented by PG21:
Yun Jihyun
Lee Chaeyeon
N Mufiz Ahmed
Son Seongyeon
Low Jia Jun

Contents

01

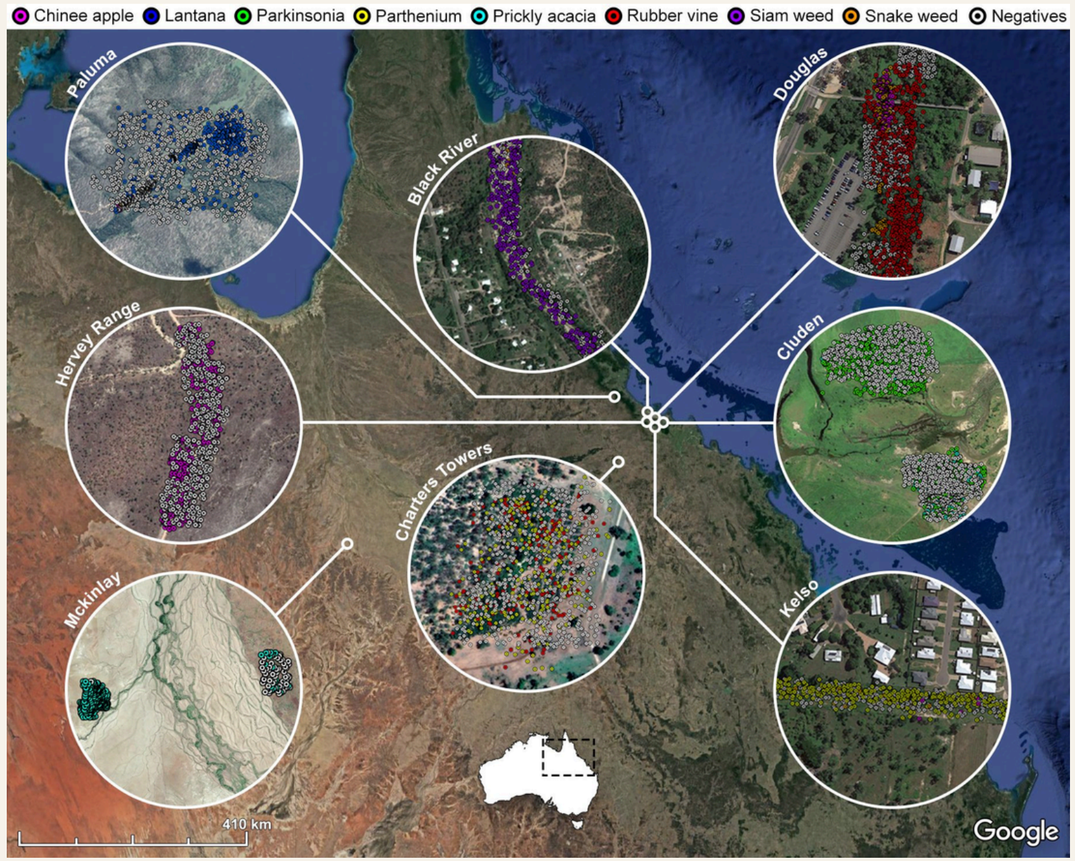
Introduction

Context

Economic Costing Australian farmers billions annually

Technological Growing need for intelligent systems

Dataset Description



The dataset contains
over 17500 images of
8 different weed species
native to Australia

Models used

01.

Linear Models (kNN, Random Forest, SVM)

02.

Linear Models with Autoencoders

03.

Convolutional Neural Network

Motivation

01.	Which model delivers the best accuracy for real–world weed classification?
02.	How do preprocessing techniques impact model performance?

Baseline Model

```
Macro-averaged Precision: 0.06
Macro-averaged Recall: 0.11
Macro-averaged F1 score: 0.08
Classification Report for Logistic Regression:
                precision    recall  f1-score   support

 Chinese Apple       0.00      0.00      0.00       213
       Lantana       0.00      0.00      0.00       215
   Parkinsonia       0.00      0.00      0.00       199
    Parthenium       0.00      0.00      0.00       215
Prickly acacia       0.00      0.00      0.00       210
   Rubber vine       0.00      0.00      0.00       200
     Siam weed       0.00      0.00      0.00       223
    Snake Weed       0.00      0.00      0.00       191
     Negatives       0.52      1.00      0.69      1836

      accuracy                           0.52      3502
     macro avg       0.06      0.11      0.08      3502
  weighted avg       0.27      0.52      0.36      3502
```

Logistic Regression with macro-averaged F1 score of 0.08

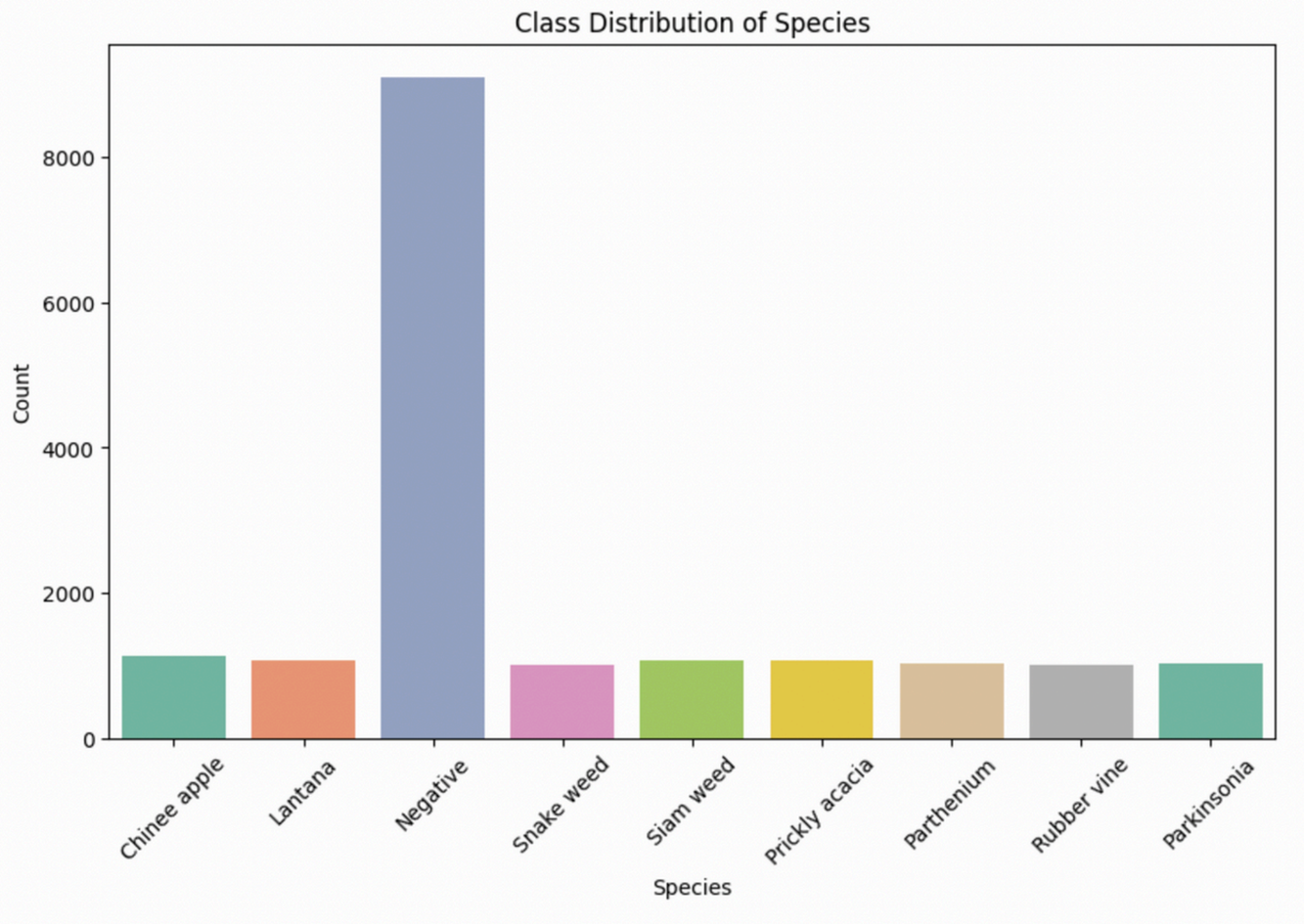
02

Data Preprocessing

Preprocessing

01.	Inspection and class imbalance
02.	Splitting and preprocessing
03.	Feature extraction

Inspection and class imbalance



Negative class
is overwhelmingly
dominant which will
reduce its ability
to accurately classify
other weed species

Inspection and class imbalance

Oversampling?
will lead to increased dataset size and potentially cause overfitting

Undersampling
Randomly selected a subset of the negative class.
Opted for this and achieved a balanced dataset

Splitting and Feature Extraction

Data splitting

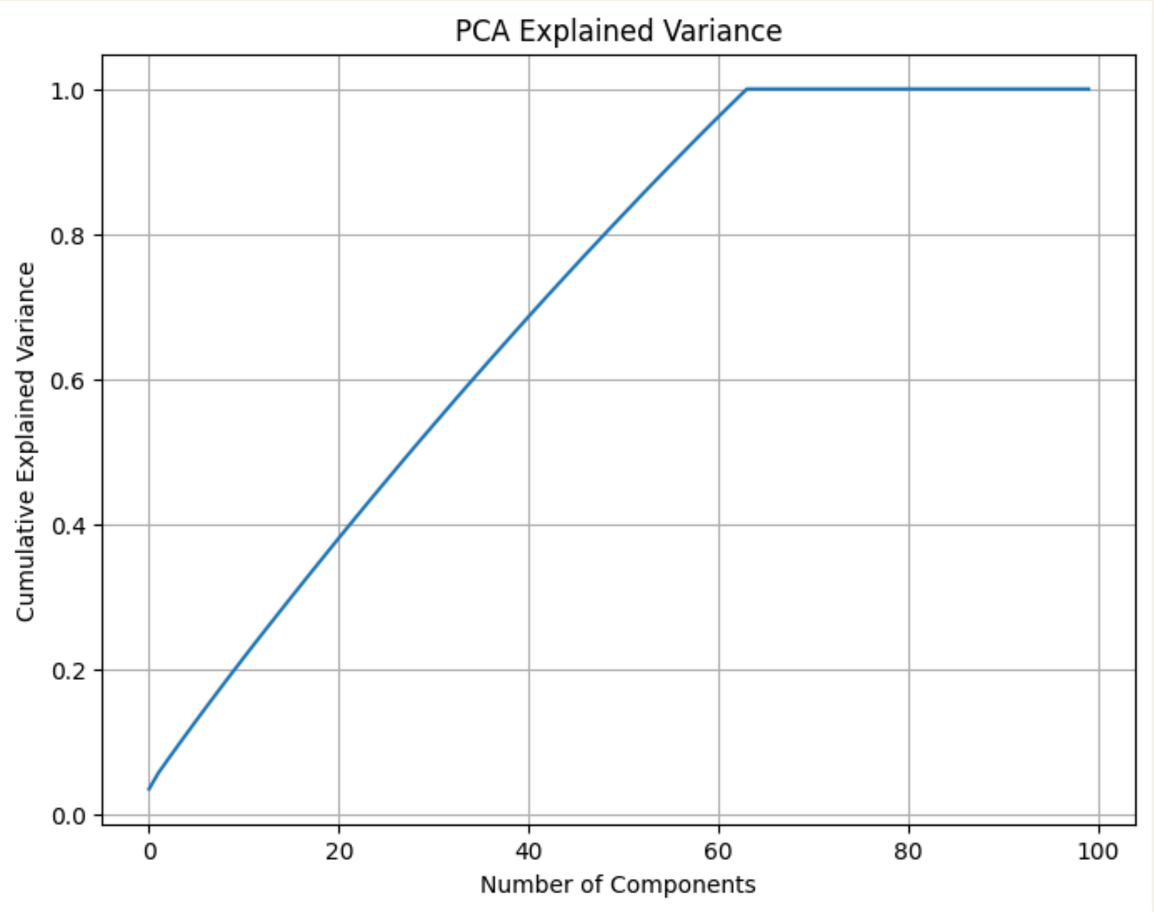
- Split the data into training and test sets to prevent data leakage
- Resized images to 224 x 224 pixels and normalized pixel values to [0,1]

Features extracted

- Mean RGB Values: Capture overall color, key for distinguishing weed species.
- Local Binary Patterns (LBP): Encode texture, useful for species classification.
- Laplacian Mean Value: Highlights sharp intensity changes, emphasizing leaf edges.
- Edge Mean Value: Quantifies edge intensity, reinforcing structural features.
- Harris Corner Mean Value: Detects key points and corners, capturing unique weed shapes.

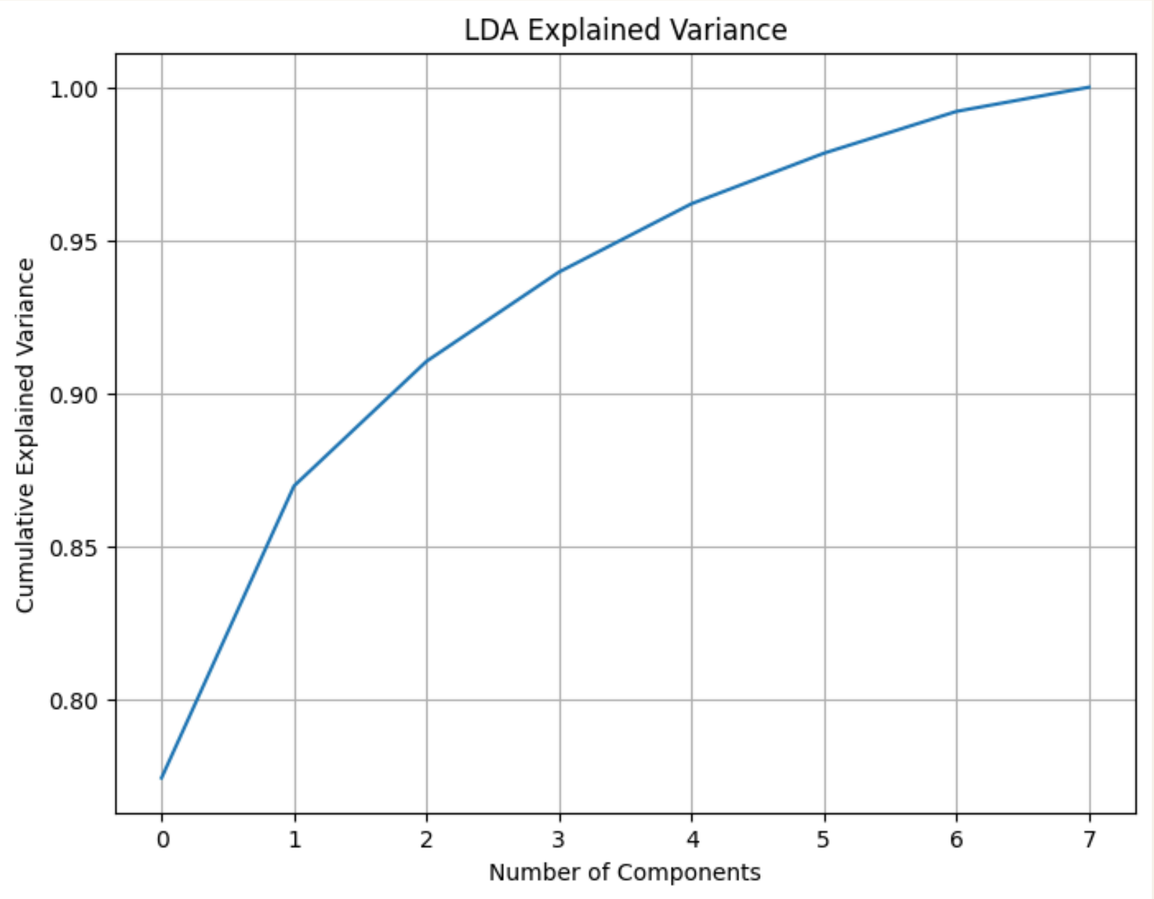
Dimensionality reduction (PCA)

150,535 features reduced to **64**!



Dimensionality reduction (LDA)

64 features
reduced to **8**!!



03

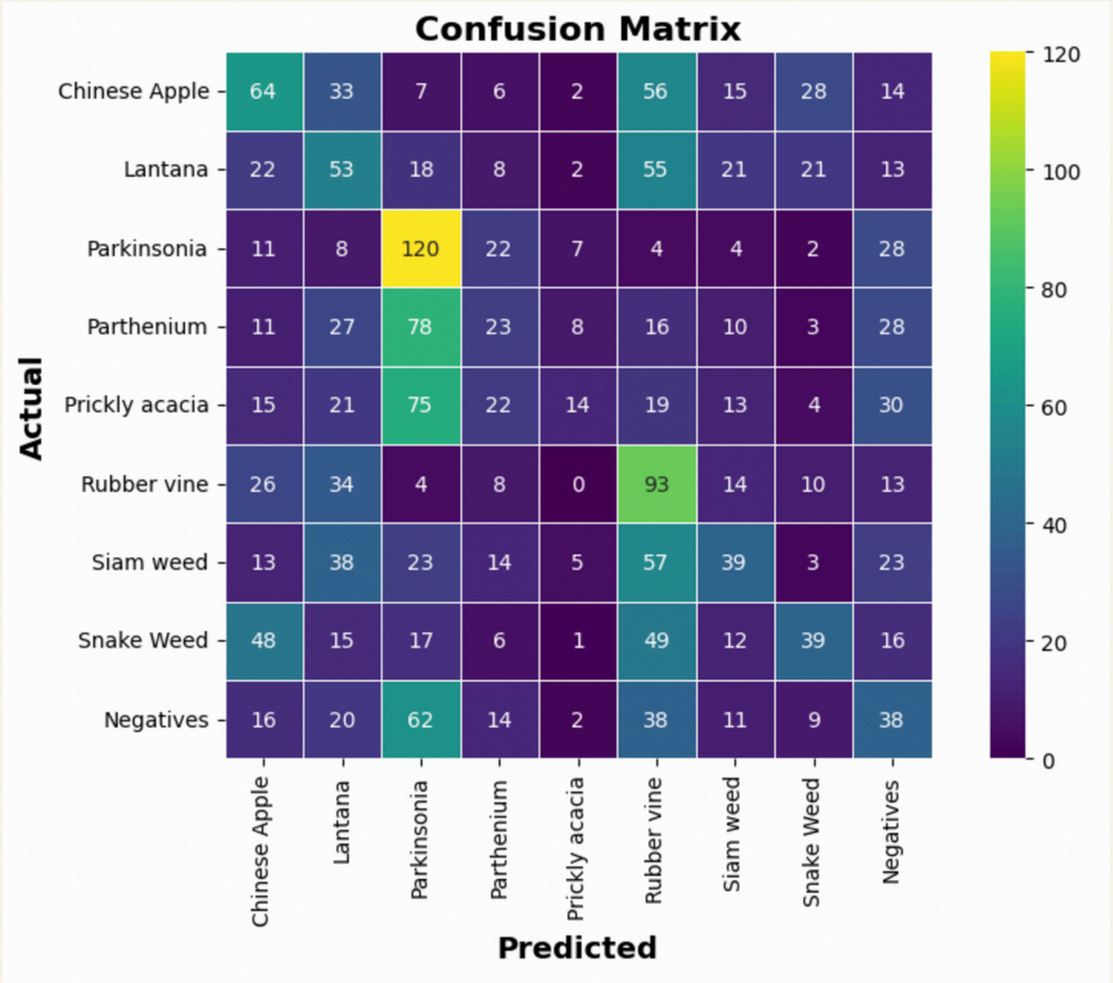
Linear Models

Linear Models

01.	k-Nearest Neighbours (kNN)
02.	Random Forest Classifier
03.	Support Vector Machine (SVM)

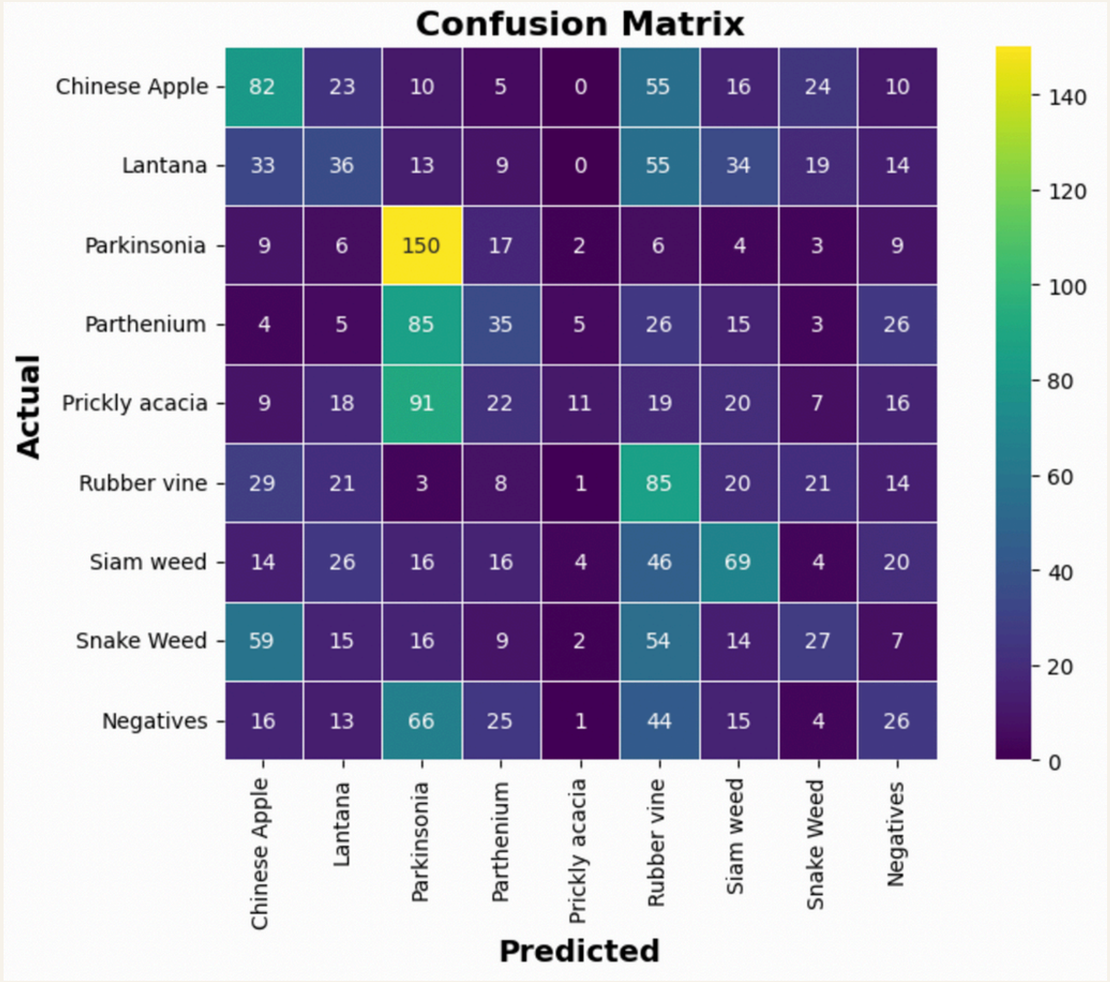
k-Nearest Neighbours (kNN)

```
Micro-averaged Precision: 0.22
Micro-averaged Recall: 0.22
Micro-averaged F1 score: 0.22
Classification Report for kNN:
                precision    recall  f1-score   support

 Chinese Apple       0.31      0.22      0.26       225
       Lantana       0.15      0.13      0.14       213
   Parkinsonia       0.13      0.75      0.22       206
    Parthenium       0.14      0.14      0.14       204
Prickly acacia       0.00      0.00      0.00       213
   Rubber vine       0.13      0.57      0.21       202
     Siam weed       0.25      0.16      0.20       215
    Snake Weed       0.14      0.13      0.13       203
     Negatives       0.66      0.18      0.29      1821

      accuracy                           0.22      3502
     macro avg       0.21      0.25      0.18      3502
  weighted avg       0.42      0.22      0.23      3502
```



Random Forest Classifier

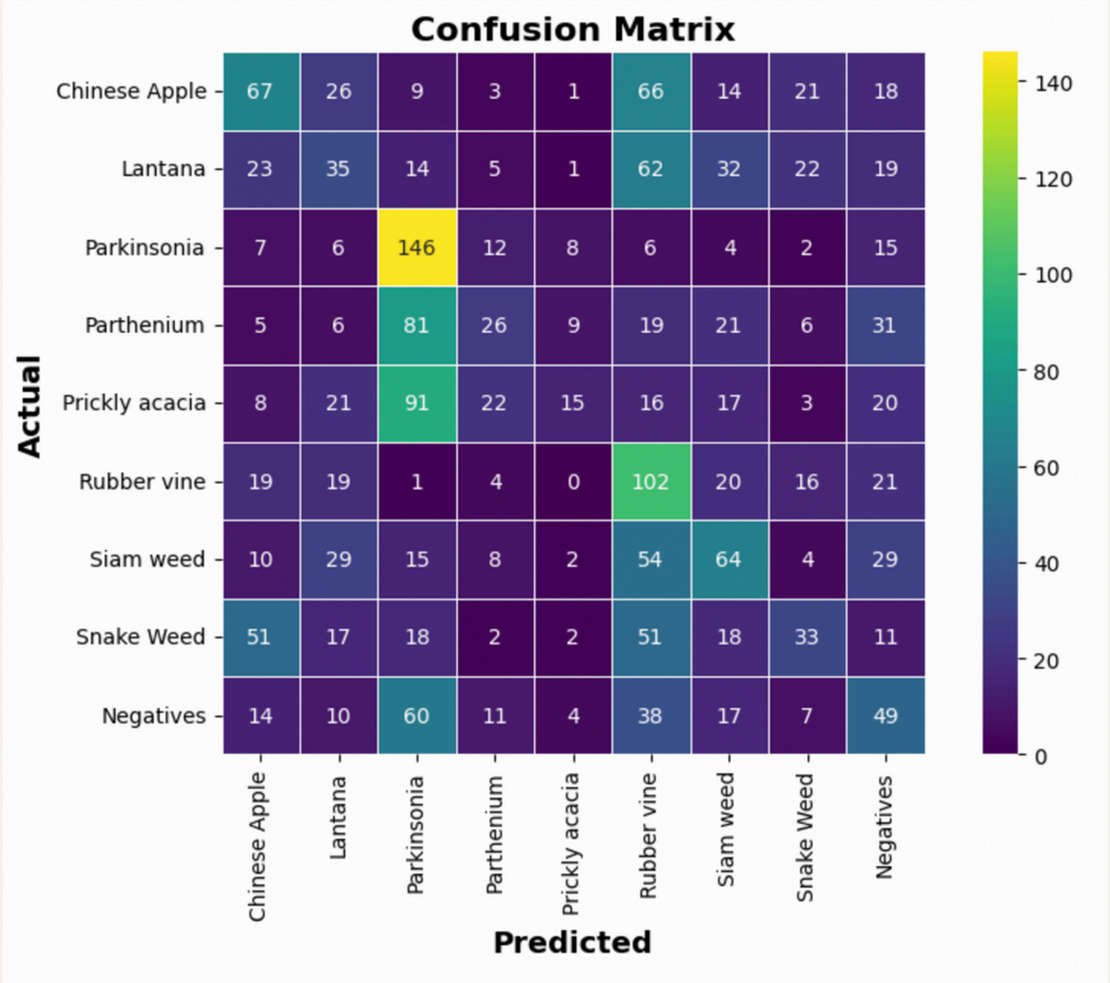
```
Micro-averaged Precision: 0.20
Micro-averaged Recall: 0.20
Micro-averaged F1 score: 0.20
Classification Report for Random Forest:
                precision    recall  f1-score   support

 Chinese Apple       0.26      0.29      0.27       225
       Lantana       0.15      0.15      0.15       213
   Parkinsonia       0.16      0.72      0.26       206
    Parthenium       0.11      0.18      0.14       204
Prickly acacia       0.18      0.03      0.05       213
   Rubber vine       0.11      0.48      0.18       202
     Siam weed       0.22      0.24      0.23       215
    Snake Weed       0.15      0.22      0.17       203
     Negatives       0.66      0.12      0.21      1821

      accuracy                           0.20      3502
     macro avg       0.22      0.27      0.18      3502
  weighted avg       0.43      0.20      0.20      3502
```



Support Vector Machine (SVM)

Micro-averaged Precision: 0.26
Micro-averaged Recall: 0.26
Micro-averaged F1 score: 0.26
Classification Report for SVM:

	precision	recall	f1-score	support
Chinese Apple	0.25	0.26	0.25	225
Lantana	0.18	0.19	0.19	213
Parkinsonia	0.17	0.68	0.27	206
Parthenium	0.17	0.11	0.14	204
Prickly acacia	0.19	0.03	0.05	213
Rubber vine	0.14	0.55	0.23	202
Siam weed	0.18	0.33	0.23	215
Snake Weed	0.13	0.12	0.12	203
Negatives	0.68	0.25	0.36	1821
accuracy			0.26	3502
macro avg	0.23	0.28	0.20	3502
weighted avg	0.44	0.26	0.28	3502



Evaluation of linear models

Micro-Averaged

	Precision	**Recall**	**F1**
kNN	0.22	0.22	0.22
Random Forest	0.20	0.20	0.20
SVM	0.26	0.26	0.26

04

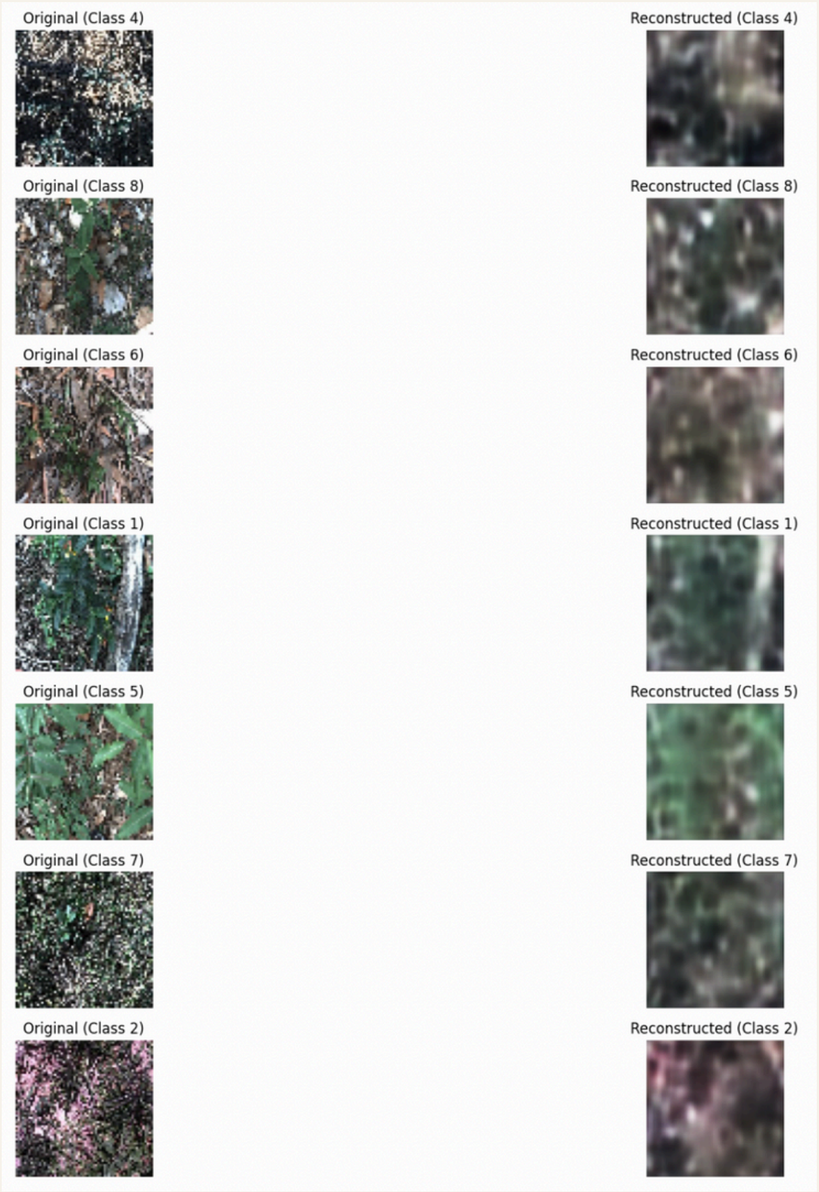
Linear Models with Autoencoders

Autoencoders with linear models

Employed autoencoder to extract more meaningful features

Operated in two stages:

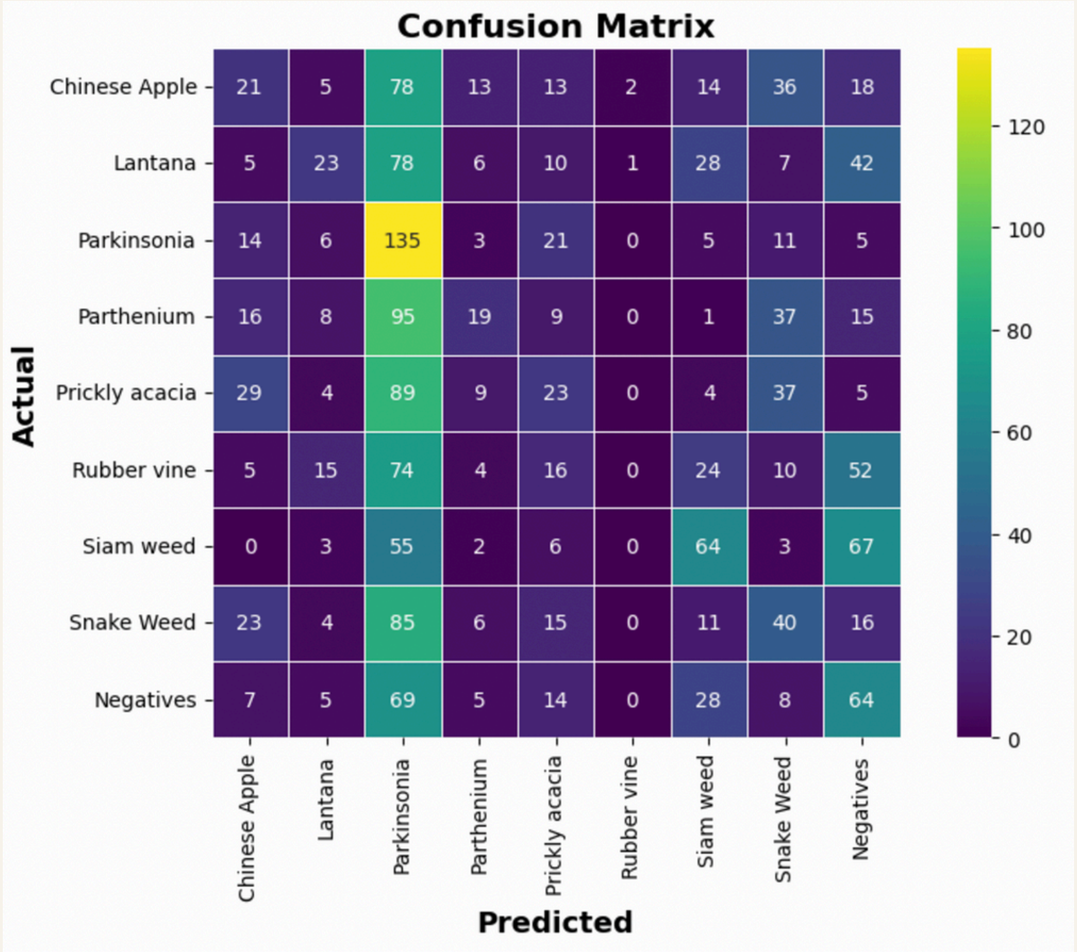
1. compresses images into lower-dimensional
2. Decoder constructs images from the compressed version



kNN using autoencoder

Macro-averaged Precision: 0.21
Macro-averaged Recall: 0.22
Macro-averaged F1 score: 0.19
Classification Report for kNN:

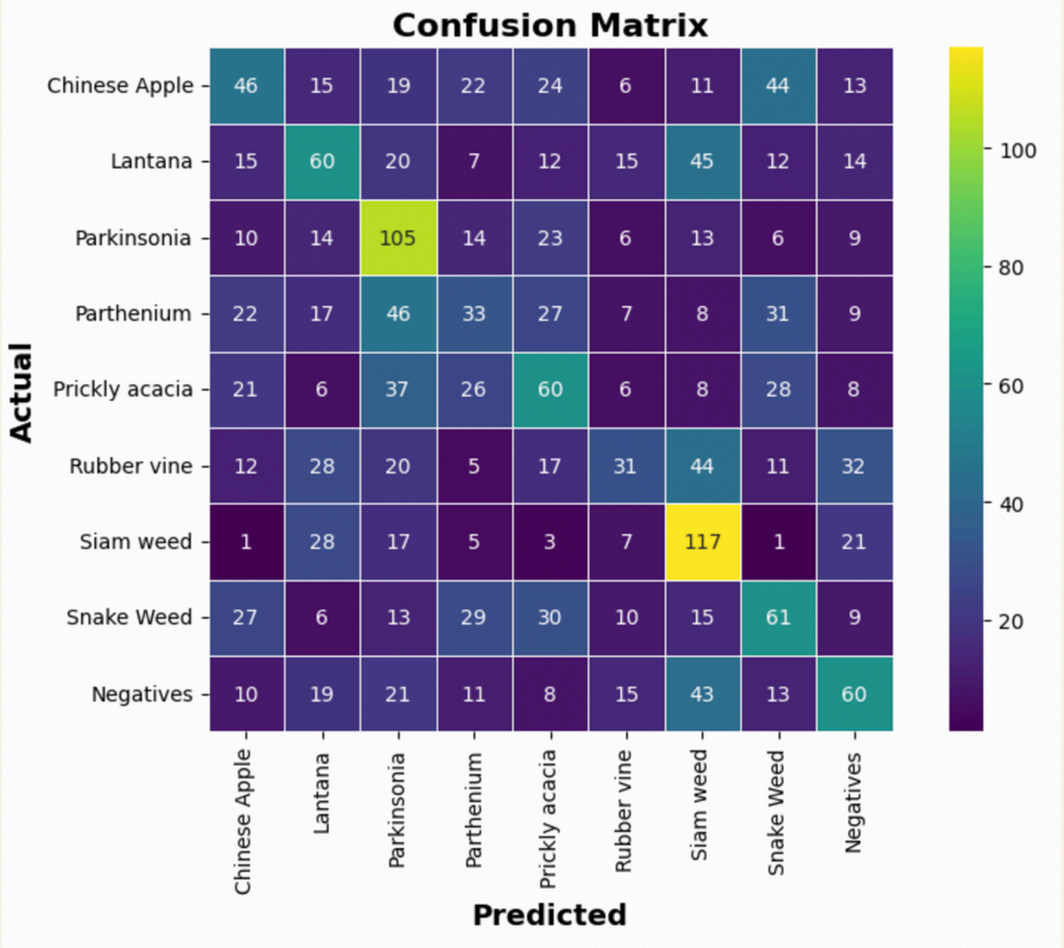
	precision	recall	f1-score	support
Chinese Apple	0.17	0.10	0.13	200
Lantana	0.32	0.12	0.17	200
Parkinsonia	0.18	0.68	0.28	200
Parthenium	0.28	0.10	0.14	200
Prickly acacia	0.18	0.12	0.14	200
Rubber vine	0.00	0.00	0.00	200
Siam weed	0.36	0.32	0.34	200
Snake Weed	0.21	0.20	0.21	200
Negatives	0.23	0.32	0.26	200
accuracy			0.22	1800
macro avg	0.21	0.22	0.19	1800
weighted avg	0.21	0.22	0.19	1800



Random Forest using autoencoder

Macro-averaged Precision: 0.31
Macro-averaged Recall: 0.32
Macro-averaged F1 score: 0.31
Classification Report for Random Forest:

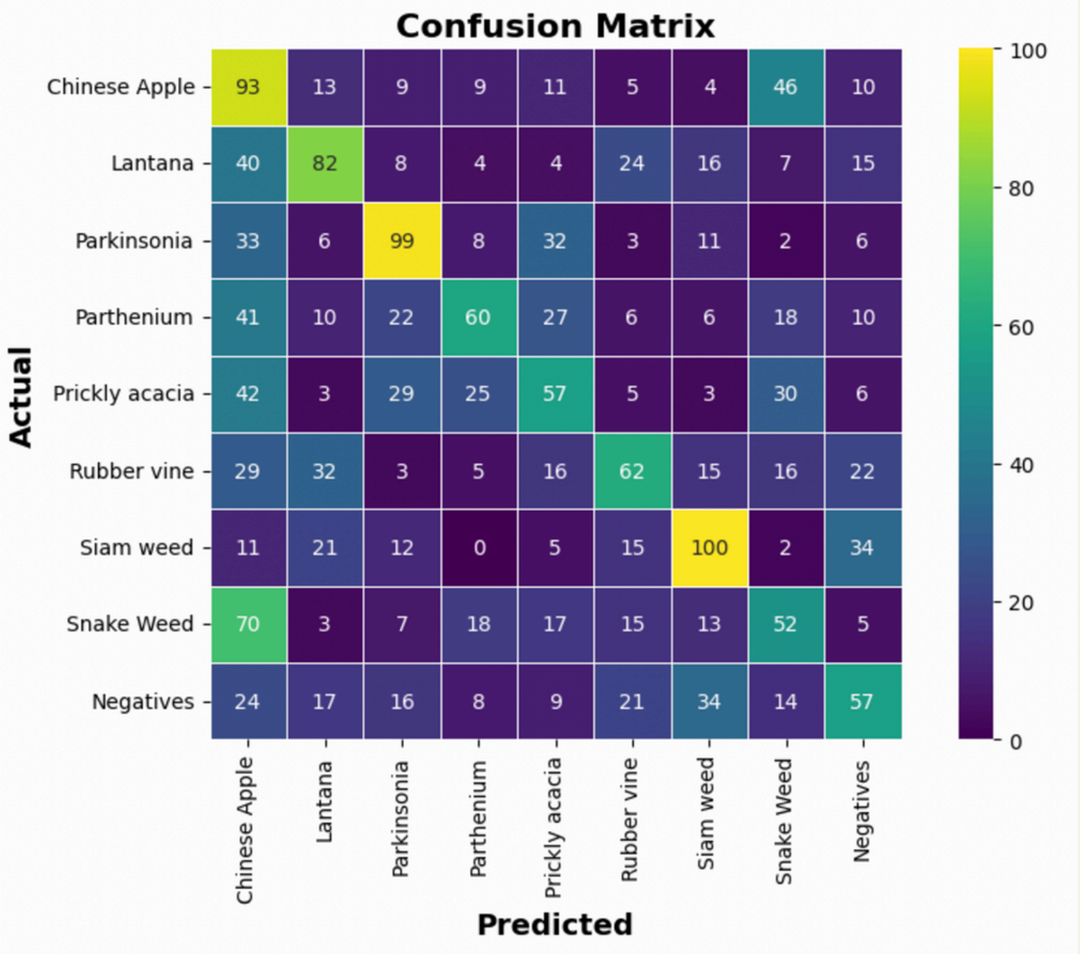
	precision	recall	f1-score	support
Chinese Apple	0.28	0.23	0.25	200
Lantana	0.31	0.30	0.31	200
Parkinsonia	0.35	0.53	0.42	200
Parthenium	0.22	0.17	0.19	200
Prickly acacia	0.29	0.30	0.30	200
Rubber vine	0.30	0.15	0.20	200
Siam weed	0.38	0.58	0.46	200
Snake Weed	0.29	0.30	0.30	200
Negatives	0.34	0.30	0.32	200
accuracy			0.32	1800
macro avg	0.31	0.32	0.31	1800
weighted avg	0.31	0.32	0.31	1800



SVM using autoencoder

Macro-averaged Precision: 0.38
Macro-averaged Recall: 0.37
Macro-averaged F1 score: 0.37
Classification Report for SVM:

	precision	recall	f1-score	support
Chinese Apple	0.24	0.47	0.32	200
Lantana	0.44	0.41	0.42	200
Parkinsonia	0.48	0.49	0.49	200
Parthenium	0.44	0.30	0.36	200
Prickly acacia	0.32	0.28	0.30	200
Rubber vine	0.40	0.31	0.35	200
Siam weed	0.50	0.50	0.50	200
Snake Weed	0.28	0.26	0.27	200
Negatives	0.35	0.28	0.31	200
accuracy			0.37	1800
macro avg	0.38	0.37	0.37	1800
weighted avg	0.38	0.37	0.37	1800



05

CNN and GradCAM

Convolutional Neural Network (CNN)

```
# Define the model
inputs = Input(shape=(224, 224, 4))

x = layers.Conv2D(64, (3, 3), padding="same", strides=2)(inputs)
x = layers.Conv2D(64, (3, 3), padding="same", strides=2)(x)
x = layers.BatchNormalization()(x)
layer1 = layers.ReLU()(x)

x = layers.Conv2D(64, (3, 3), padding="same")(layer1)
x = layers.Conv2D(64, (3, 3), padding="same")(x)
x = layers.BatchNormalization()(x)
layer2 = layers.ReLU()(x)

x = layers.Conv2D(128, (3, 3), padding="same")(layer2)
x = layers.Conv2D(128, (3, 3), padding="same")(x)
x = layers.BatchNormalization()(x)
layer3 = layers.ReLU()(x)

x = layers.Conv2D(128, (3, 3), padding="same")(layer3)
x = layers.Conv2D(128, (3, 3), padding="same")(x)
x = layers.BatchNormalization()(x)
layer4 = layers.ReLU()(x)

x = layers.Conv2D(256, (3, 3), padding="same", strides=2)(layer4)
x = layers.Conv2D(256, (3, 3), padding="same", strides=2)(x)
x = layers.BatchNormalization()(x)
layer5 = layers.ReLU()(x)

x = layers.GlobalMaxPooling2D()(x)
x = layers.Dropout(0.5)(x)
outputs = layers.Dense(9, activation="softmax")(x)
```

```
Epoch 1/20
197/197 [==============================] - 738s 4s/step - loss: 2.5979 - accuracy: 0.2319 - precision: 0.2587 - re
call: 0.1178 - val_loss: 4.6115 - val_accuracy: 0.1211 - val_precision: 0.1278 - val_recall: 0.1200
Epoch 2/20
197/197 [==============================] - 685s 3s/step - loss: 1.8098 - accuracy: 0.3300 - precision: 0.4214 - re
call: 0.1208 - val_loss: 2.6872 - val_accuracy: 0.2422 - val_precision: 0.3079 - val_recall: 0.1656
Epoch 3/20
197/197 [==============================] - 671s 3s/step - loss: 1.6615 - accuracy: 0.3732 - precision: 0.5052 - re
call: 0.1532 - val_loss: 1.7049 - val_accuracy: 0.3711 - val_precision: 0.4748 - val_recall: 0.1467
Epoch 4/20
197/197 [==============================] - 682s 3s/step - loss: 1.5899 - accuracy: 0.4068 - precision: 0.5437 - re
call: 0.1897 - val_loss: 1.8303 - val_accuracy: 0.3711 - val_precision: 0.4619 - val_recall: 0.2356
Epoch 5/20
197/197 [==============================] - 645s 3s/step - loss: 1.5404 - accuracy: 0.4243 - precision: 0.5645 - re
call: 0.1986 - val_loss: 1.6364 - val_accuracy: 0.4011 - val_precision: 0.5466 - val_recall: 0.2344
```

1st Epoch

validation set

- **precision: 0.1278**
- **recall: 0.1200**

```
Epoch 15/20
197/197 [==============================] - 636s 3s/step - loss: 1.1302 - accuracy: 0.5924 - precision: 0.7173 - re
call: 0.4433 - val_loss: 1.5581 - val_accuracy: 0.4478 - val_precision: 0.6359 - val_recall: 0.2833
Epoch 16/20
197/197 [==============================] - 653s 3s/step - loss: 1.0938 - accuracy: 0.6052 - precision: 0.7321 - re
call: 0.4637 - val_loss: 1.4341 - val_accuracy: 0.5589 - val_precision: 0.6026 - val_recall: 0.4633
Epoch 17/20
197/197 [==============================] - 24642s 126s/step - loss: 1.0498 - accuracy: 0.6192 - precision: 0.7401
- recall: 0.4895 - val_loss: 3.1679 - val_accuracy: 0.2600 - val_precision: 0.2845 - val_recall: 0.2200
Epoch 18/20
197/197 [==============================] - 649s 3s/step - loss: 1.0525 - accuracy: 0.6219 - precision: 0.7330 - re
call: 0.4859 - val_loss: 1.2306 - val_accuracy: 0.5678 - val_precision: 0.6558 - val_recall: 0.5144
Epoch 19/20
197/197 [==============================] - 611s 3s/step - loss: 1.0122 - accuracy: 0.6378 - precision: 0.7526 - re
call: 0.5065 - val_loss: 1.0664 - val_accuracy: 0.6167 - val_precision: 0.7210 - val_recall: 0.4767
Epoch 20/20
197/197 [==============================] - 640s 3s/step - loss: 0.9935 - accuracy: 0.6398 - precision: 0.7514 - re
call: 0.5149 - val_loss: 1.1733 - val_accuracy: 0.5922 - val_precision: 0.7003 - val_recall: 0.5089
```

Last 20th Epoch

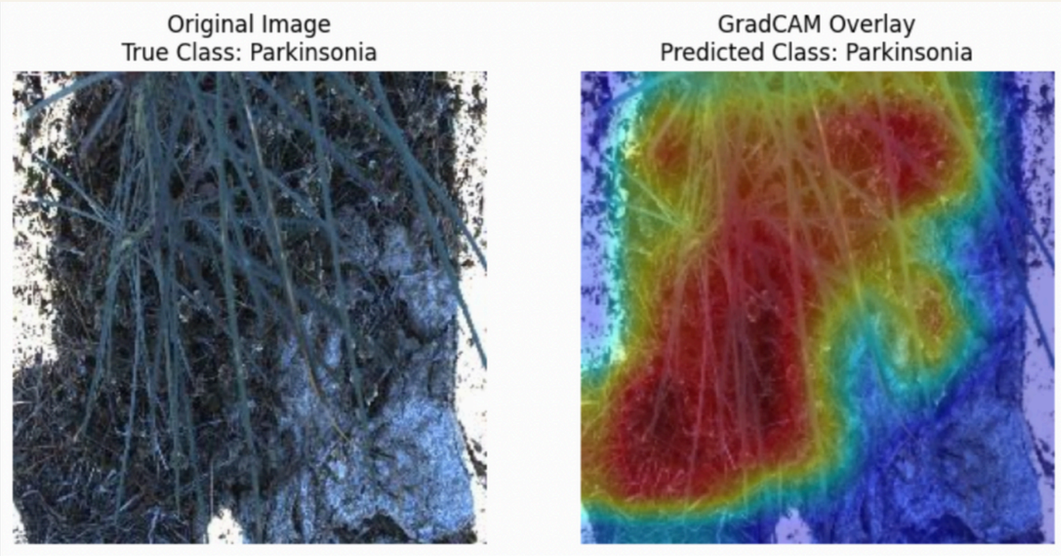
validation set

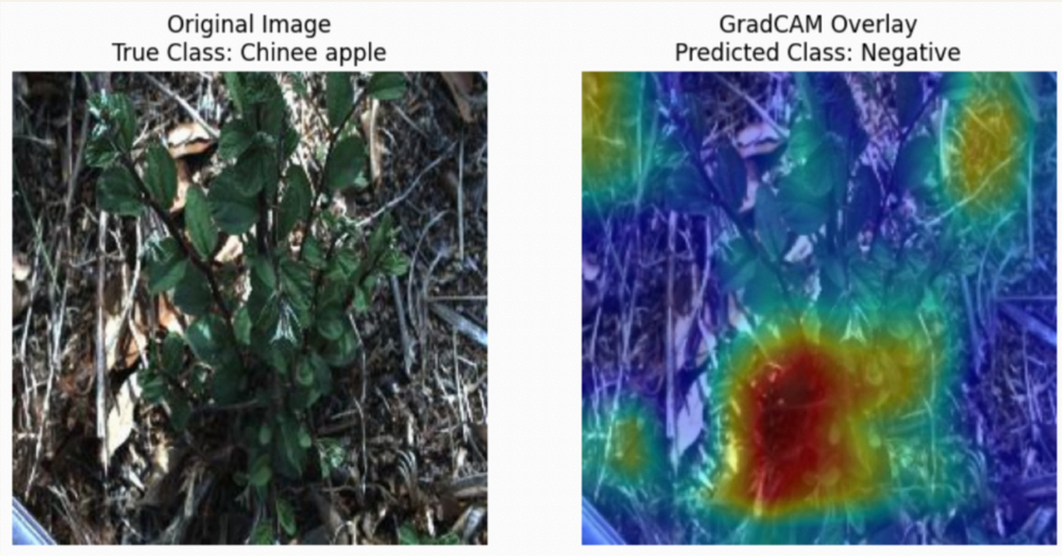
- **precision: 0.7003**
- **recall: 0.5089**

Convolutional Neural Network (CNN)

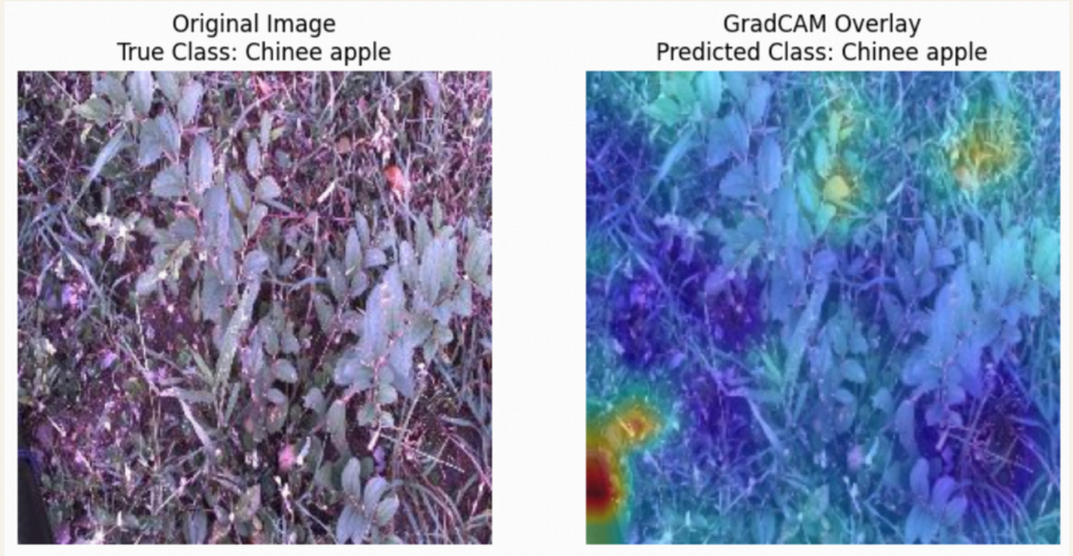
```
Micro-averaged Precision: 0.62
Micro-averaged Recall: 0.62
Micro-averaged F1 score: 0.62
                precision    recall  f1-score   support

 Chinese Apple       0.75      0.04      0.08       225
       Lantana       0.37      0.77      0.50       213
   Parkinsonia       0.56      0.67      0.61       206
    Parthenium       0.41      0.75      0.53       204
Prickly acacia       0.63      0.75      0.69       213
   Rubber vine       0.45      0.83      0.59       202
     Siam weed       0.50      0.77      0.60       215
    Snake Weed       0.41      0.52      0.46       203
     Negatives       0.90      0.60      0.72      1821

      accuracy                           0.62      3502
     macro avg       0.55      0.63      0.53      3502
  weighted avg       0.72      0.62      0.62      3502
```

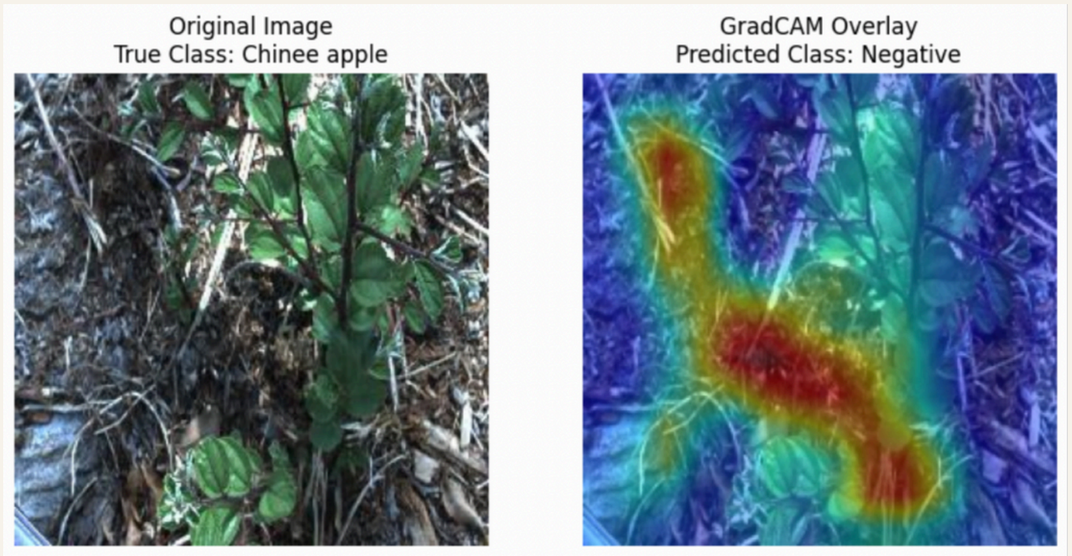
GradCAM (correctly identified)





GradCAM (wrongly identified)





06

Conclusion

Conclusion

01.

This project focused on improving weed classification in Australian agriculture using machine learning, overcoming challenges like class imbalance.

02.

While traditional models had limitations, Convolutional Neural Networks (CNNs) outperformed them by capturing complex patterns.

03.

Grad-CAM insights confirmed the CNN's effectiveness, offering a transformative solution for weed management in farming.

Thank you