dortmund university

Image Recognition for Prediction of Forest Health

Outline

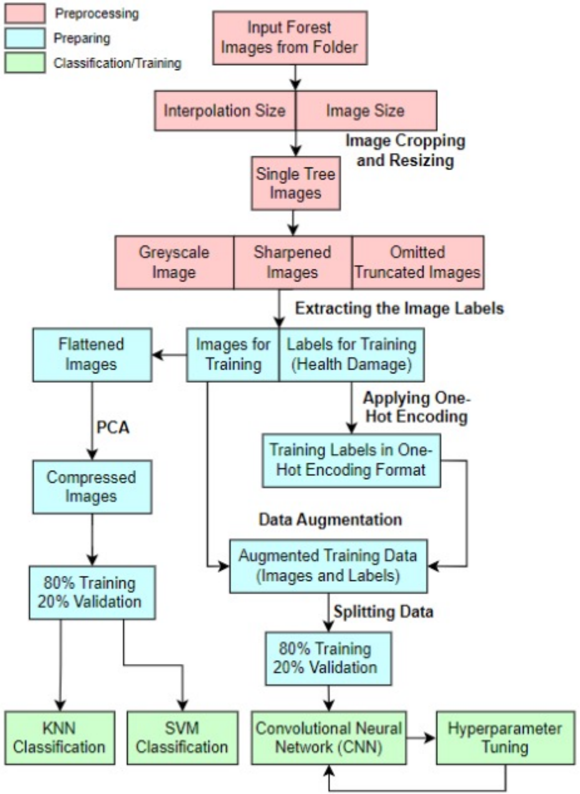
- Introduction
- Overview
- Data Preprocessing
- Principal Component Analysis
- Classification models:
  - Support Vector Machine
  - K-Nearest Neighbour
  - Convolutional Neural Network
- Comparative Study
- Conclusion
- References

Introduction

- Image Classification is one of the application of machine learning
- **Objective of project:** To predict the forest health of given images using different machine learning algorithms

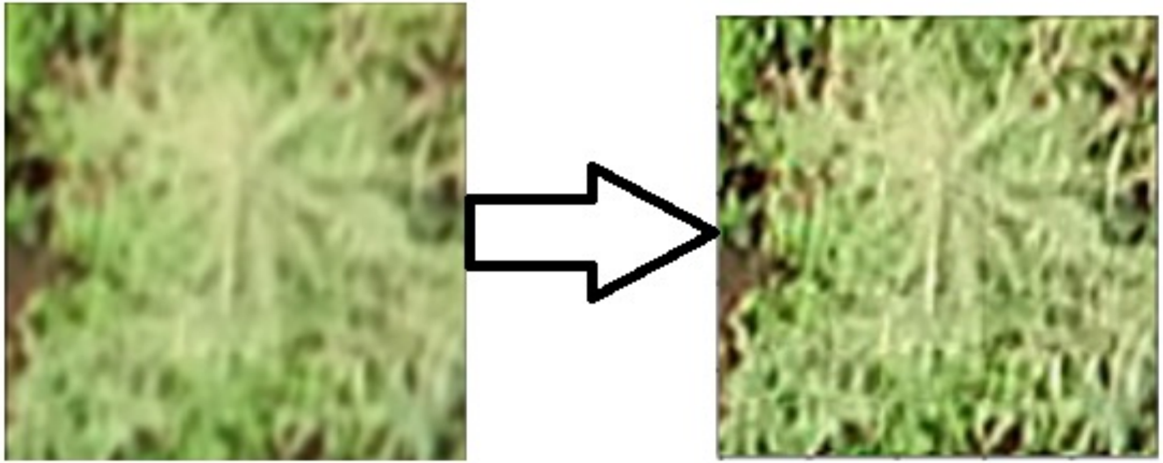
Overview

- Stage 1: Data Pre-processing
- Stage 2: Data Preparing
- Stage 3: Training and Classification

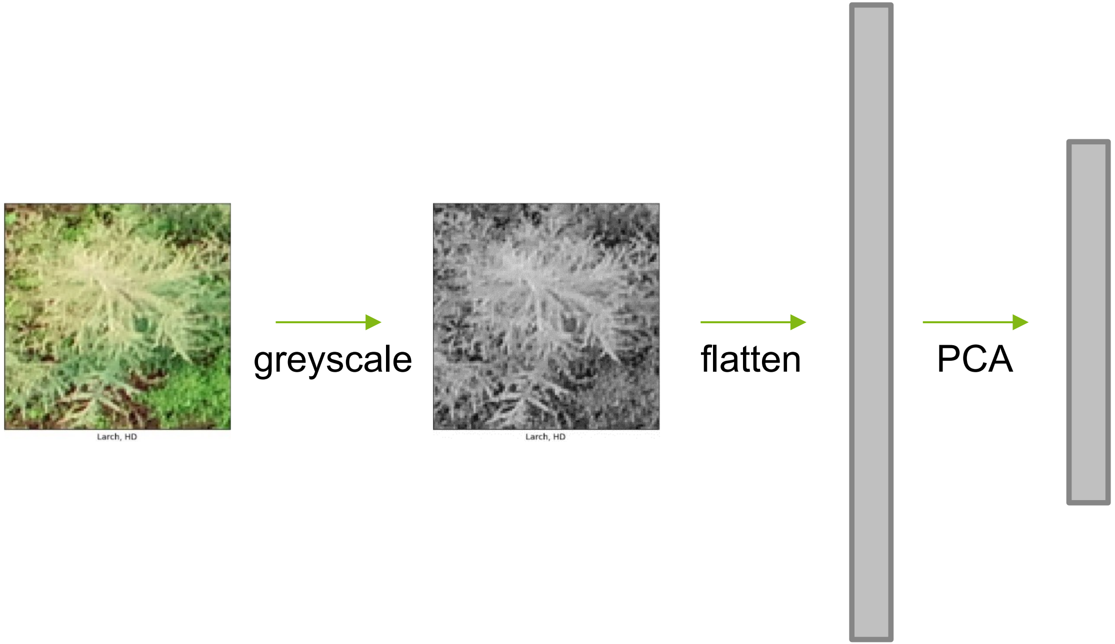


Data Preprocessing - Part 1

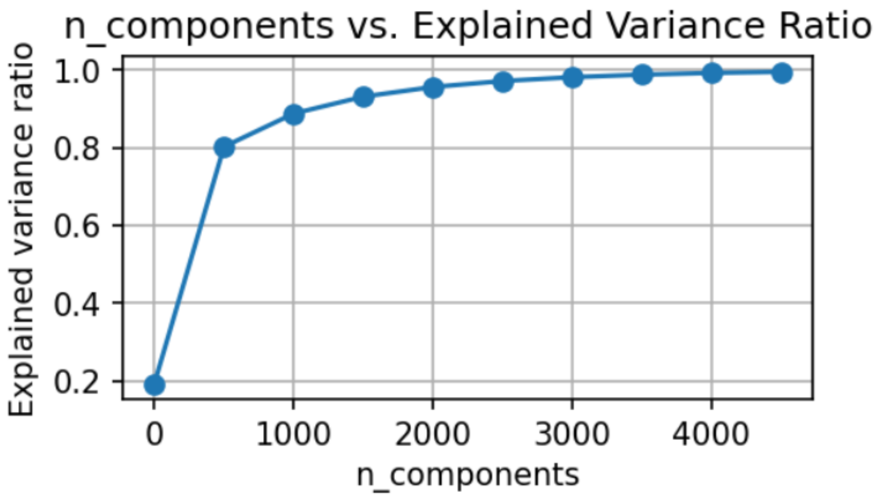
Data Preprocessing - Part 2



PCA – Implementation



PCA – Results



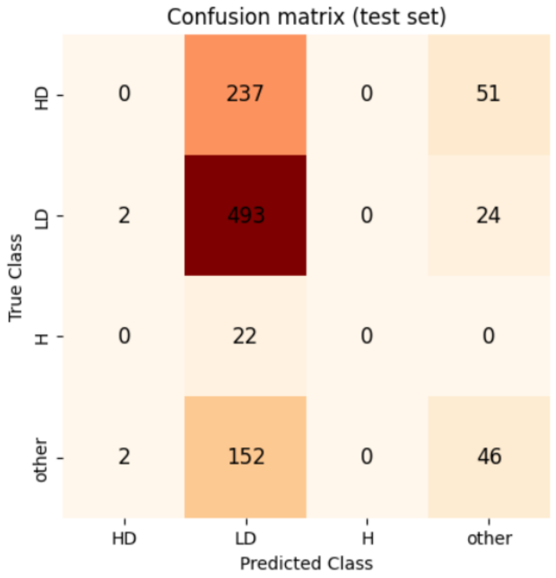
n_components vs. Explained Variance Ratio
Explained variance ratio
1.0
0.8
0.6
0.4
0.2
0
1000
2000
3000
4000
n_components

SVM - Implementation

- Library: *sklearn.svm.SVC*
- Methods for multiclass: ovo, ovr
- Class_weight = balanced
- 500 Principal Components
- 80% explained variance ratio
- 95% compression

SVM - Results

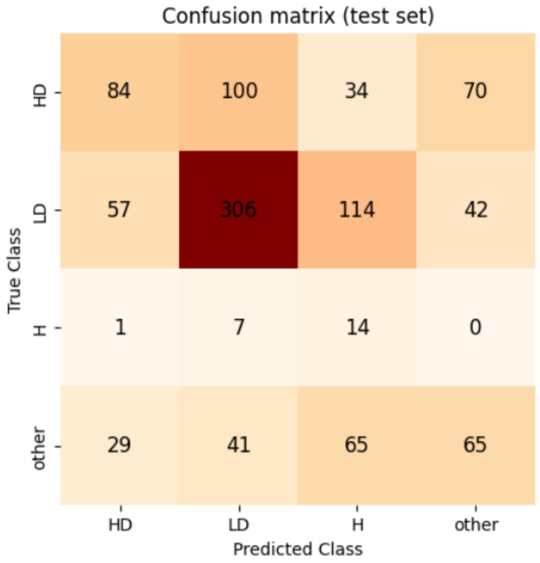
- without regularization
- Validation accuracy: 53,8%
- Test accuracy: 52,7%



SVM is not fitted to the features

SVM - Results

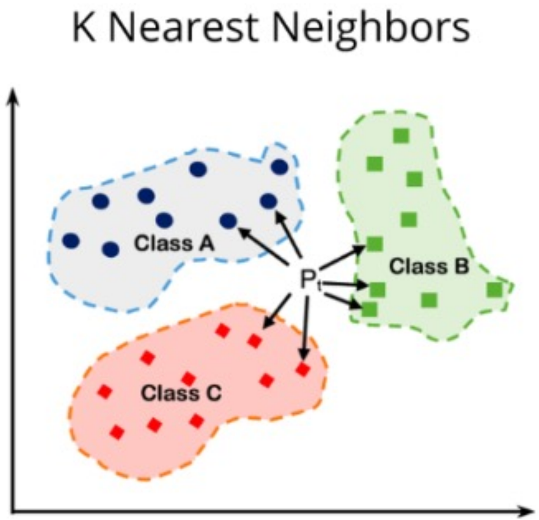
- with regularization
- Validation accuracy: 53,6%
- Test accuracy: 44,5%



SVM is fitted to the features, but overall bad performances -> information loss

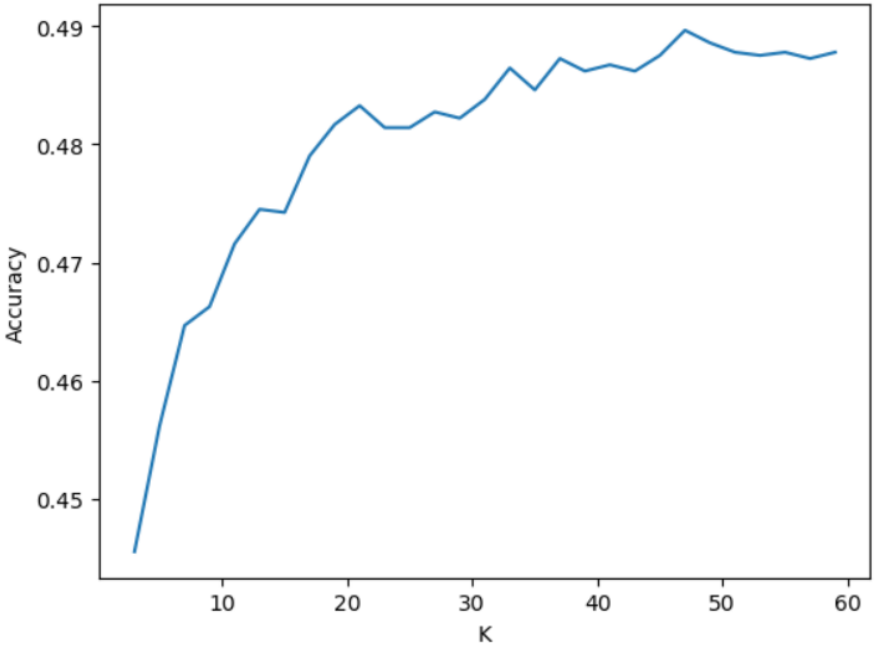
KNN - Implementation

- Libraries used: sklearn.model_selection, sklearn.neighbors, and sklearn.metrics
- 2 PCA
- 24.5% explained variance
- 99.98% compression



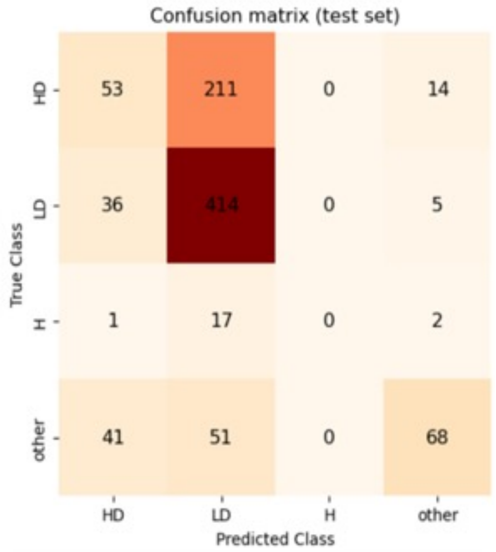
KNN - Accuracy Graph

- Graph of Accuracy (%) vs K nearest neighbours
- Accuracy stops improving after at around K = 40 (Optimal K value = 40)



KNN - Results

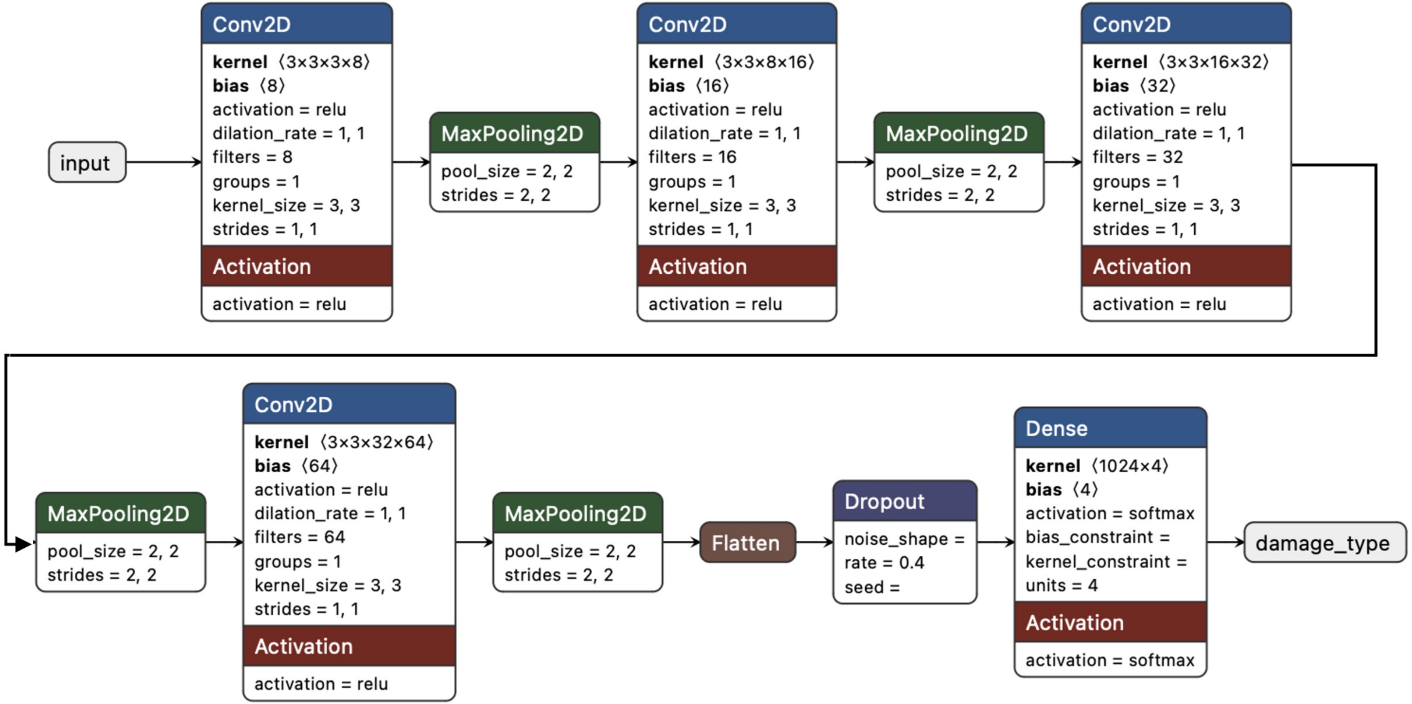
- Most predicted classes are Low Damage (LD), followed by High Damage (HD).
- No prediction for Healthy (H)
- Struggles to predict HD and Other



CNN - Implementation

- Library: *tensorflow.keras*
- Functional API
- Optimizer: Adam
- Loss Function: Categorical Cross-Entropy
- During training:
  - Custom Model Checkpoint
  - Shuffling of the training data

CNN - Architecture



CNN - Hyperparameter Tuning (1)

- Method - **Grid search**

- **Structural** hyper-parameters: number of layers and filters/neurons, dropout rate.
- **Training** hyper-parameters: learning rate, batch size, number of epochs.
- **Pre-processing** parameters: image size, omitting truncated images, interpolation method for resizing

CNN - Hyperparameter Tuning (2)

Learning Rate	Dropout Rate	Batch Size	Epochs	Image Size	Omit Truncated	Interpolation Type	Sharpen Image
0.0003	0.4	32	50	100 x 100	No	Cubic	Yes

- Optimal Hyper-parameters chosen to train the CNN after grid search

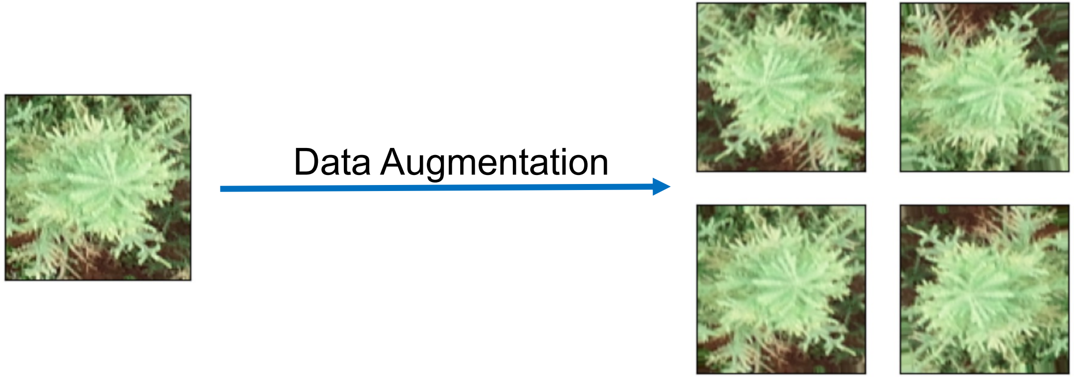
CNN - Analyzing the Training Data

Class (Damage Type)	HD	LD	H	other
# of training images	3,215	12,707	1,659	2,941

- Clearly the data is unbalanced!
- Solution => Data Augmentation

CNN - Data Augmentation

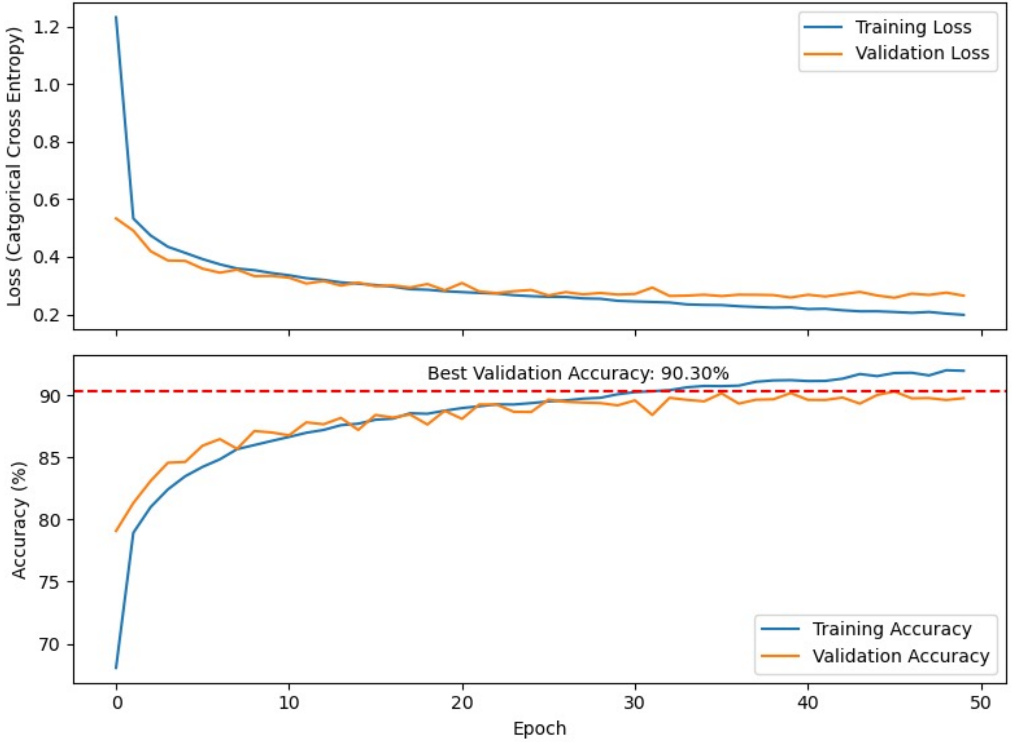
- Generate new images from the original ones
- *ImageDataGenerator* class in *tensorflow.keras.preprocessing.image*
- Rotate and Flip!
- Avoiding aggressive augmentation



CNN - Training

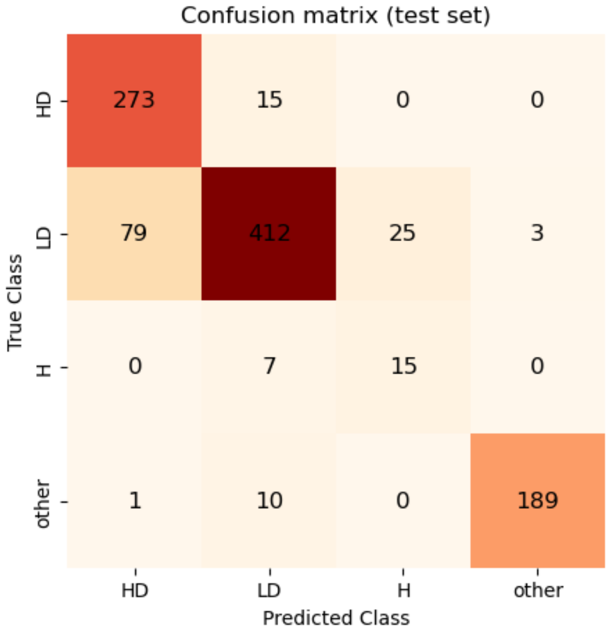
- Training/Validation loss and accuracy for 50 epochs using balanced and augmented data:

  Best Validation Accuracy: 90.3%



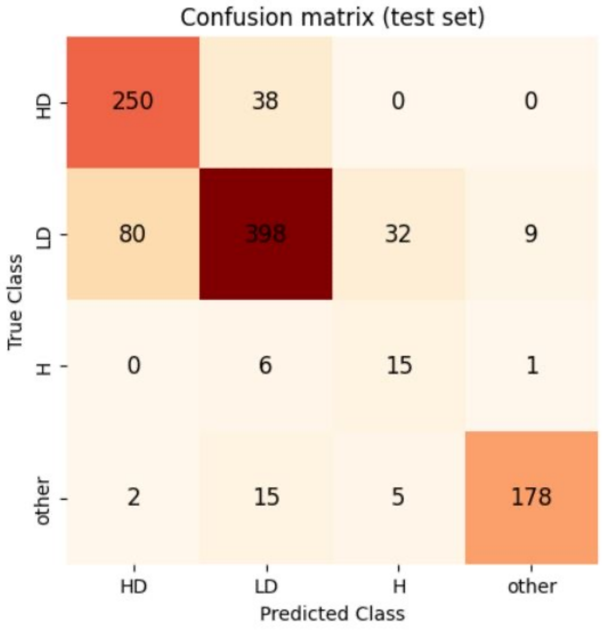
CNN - Results

- Trained with balanced and augmented data
- Test Accuracy: 87.3% (unseen data)



CNN - Results

- Trained using weighted sampling
- Class Weights:
  - HD: 1.8
  - LD: 1
  - H: 10
  - other: 3
- Test accuracy: 81%



Comparative Study

Model	K-NN	SVM	CNN
Validation Accuracy (%)	53	44.5	90.3
Generalization	Bad	Moderate	Very Good
Model Complexity	Low	Low	High

Conclusion

- Employing ML Classification for forest image recognition is very efficient
- Pre-processing and preparing training data is crucial
- Data should be balanced, diverse, extensive, and representative of actual problem
- CNNs are the most reliable for image classification
- Hyper-parameter tuning is challenging but necessary

References

- The Importance of Data Splitting https://mlu-explain.github.io/train-test-validation/, visited on 17.07.2024
- scikit-learn documentation https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html, visited on 15.07.2024
- Image Data Generator - tensorflow documentation https://www.tensorflow.org/apidocs/python/tf/keras/preprocessing/image/ImageDataGenerator, visited on 03.07.2024
- Support Vector Machine (SVM) Algorithm https://www.geeksforgeeks.org/support-vector-machine-algorithm/, visited on 17.07.2024
- scikit-learn documentation https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html, visited on 15.07.2024
- Convolutional Neural Networks (CNNs) in Computer Vision https://medium.com/@AIandInsights/convolutional-neural-networks-cnns-in-computer-vision-10573d0f5b00, visited on 17.07.2024
- CNN Functional API - tensorflow documentation https://www.tensorflow.org/guide/keras/ functional_api, visited on 10.06.2024
- Model Checkpoint - keras documentation https://keras.io/api/callbacks/ model checkpoint/, visited on 15.06.2024
- Dablain, D., Jacobson, K.N., Bellinger, C. et al. Understanding CNN fragility when learning with imbalanced data. Mach Learn 113, 4785–4810 (2024).https://doi.org/10.1007/s10994-023-06326-9