**Introduction**

This report summarizes the design process and results for a CNN used to classify different kinds of skin defects found on aircraft. The model was developed using TensorFlow and Keras.

**Data Processing**

The sample dataset was generated using Keras’s image dataset from directory function. Due to the limited samples available for training, data augmentation was performed by defining a shear and zoom range for the training images. This served to increase the amount of data available to the model, for it to have better generalization. The input shape was set to (100, 100), which is smaller than the required (500, 500) shape. Attempting to use the required shape led extremely long training times, so a reduced shape was required.

**Network Design**

The model was based on the LeNet architecture, which comprises successive blocks of convolution and pooling layers for feature extraction, before flattening and using a fully connected layer for classification. This structure was chosen as it can achieve reasonable results without the significant computational requirements of more complex networks. Batch Normalization layers were used to improve performance. The ReLU activation function was used for its non-linearity and its avoidance of the vanishing gradient problem. The number of filters and kernel size were increased until significant overfitting was observed, implying the model was sufficiently complex. The model was then regularized to increase its ability to generalize to unseen data. This was done in two ways. The first was using dropout layers, and the second through L2 regularisation.

**Hyperparameter Tuning**

To have immediate feedback when adjusting the hyperparameters, the input shape was set to (50, 50), allowing for rapid model convergence. The hyperparameter that seemed to have the greatest effect on model performance was the learning rate. To optimize this parameter, a learning schedule was used, to have a larger rate in the earlier epochs, and then gradually reduce it as training progressed. The leakyReLU activation function was tested, but didn’t provide significant changes in model performance. Additionally, the number of neurons in the dense layer was adjusted, and it the model was able to achieve decent results with a limited size.

A graph with blue and orange lines

Description automatically generatedA graph of a number of objects

Description automatically generated with medium confidence**Model Evaluation**

Figure 1: Training and validation loss Figure 2: Training and validation accuracy

Figure 1 demonstrates a drastic reduction in both training and validation loss for the first 10 epochs, and then a much more gradual decrease until the training stops just after 80 epochs. It can also be seen the loss oscillates, especially after 50 epochs. This indicates that the learning rate is perhaps too high. Attempts were made to reduce it, but that resulted in the model taking an unreasonable amount of time to converge. Figure 2 compares the training and validation accuracy. Oscillations are once again present in this figure and are more pronounced on the validation set. This is due to the smaller size of the validation set compared to the training set, meaning that the slightly different weights used for each epoch results in more variation for the predictions on the validation set.

**Model Testing**

A hole in a metal surface

Description automatically generatedUpon testing the model on the test dataset, it achieved an accuracy of 85.71%, and a loss of 0.46.

Figure 3: Results of model applied to sample images

From figure 3, the model was able correctly predict all 3 sample images. An interesting point is it had complete confidence in its prediction for the missing screw head, but less so for the crack and paint-off defects. This makes sense, as the screw head appears visually distinct from the other two. This implies that the feature extraction portion of the model can distinguish certain key features of the training images.

When initial testing of the model was done with the input shape of (50, 50), the model couldn’t really tell the crack and paint-off classes apart, and its predictions for both classes was around 50% for the two samples. At that resolution, it was also difficult to tell them apart visually. To correct this, the resolution of the images was increased until they could be discerned by the human eye. Training at this new input resolution of (100, 100) increased the model performance significantly, highlighting that the initial issue was not enough data to discern the unique features of the two defects.

**Conclusion**

This project was able to demonstrate a real-world scenario where neural networks could play a critical role in automating repetitive inspection tasks and improving the efficiency of aircraft maintenance operations. Further improvements should be made to improve the accuracy of the model, as in its current state, human inspection would probably still outperform it. To do this, higher resolution images should be supplied to the model for training, and improved computer hardware should be harnessed to facilitate this more robust training load.

**GitHub Submission**