**Steel Defects Classification**

In this section we built a multi-class classifier to classify the type steel defect shown on an image. To mitigate the class-imbalance, during training we will augment the images and for every batch augmented we will over sample the minority classes to have better representation. To evaluate the performance of the model, we expect the precision, recall and f1 scores to be similar in training, validation and testing datasets. Class number 3 is approximately 75 percent of the whole dataset. In other words, just by predicting class 3 the model can achieve an accuracy of 75 percent. Therefore, we need the model to generalize among each class as much as possible.

The images were preprocessed into a shape of 64 by 64 to a gray scale, as a way to reduce the number of features. To reduce the memory footprint, to train the model we will use an image generator from Keras. As mentioned above, an additional piece of code was added to balance the image generated during training. To augment the training data, the image generator will flip the image horizontally and vertically, and also tweak the brightness from a range of 02. to 0.8. To ensure reproducibility, the random number generator was seeded at 42.

To reduce overfitting due to the class imbalance, the model was created with several dropouts. After the first convolutional layer we used a batch normalization layer which improved the learning of the algorithm. The following layers are two blocks of convolution, 10 percent spatial dropout and max-pooling layer, then another convolution and a max-pool layer. The classifier has a ten percent dropout with 512 neurons in the first dense layer and 4 neurons in the output layer. This model architecture gives a total of 5,109,060 trainable parameters. Due the image data generator will act randomly, to enhance the learning and avoid learning cycles the optimizer used is stochastic gradient descent. The loss function is categorical entropy and the target metrics is accuracy. The activations functions for the convolutional and the first layer was Relu and SoftMax for the output layer.

A close up of a device

Description automatically generated

Figure 1. Multi-Class Classifier Architecture.

Training convolutional models takes time and sometimes after a few epochs it can start overfitting. One strategy we used was to record the best weights after every epoch based on the accuracy of the validation. Additionally, we added early stopping and reduce plateau to monitor the model's learning and stop the process when the model has already learned enough.

A screenshot of a cell phone

Description automatically generated A screenshot of a cell phone

Description automatically generated

Figure 2. Classifier Accuracy per Epoch on the left and Loss per Epoch on the right during training.

Initially for every epoch the accuracy of the validation increased while its loss was not stable. After 20 epochs the accuracy of the training and validation plateaus out but the loss in the validation does not show a significant change as in the training set. This trend in training and the confusion matrix and classification report of the trained model shows the model does generalize for class 1, 2 and 3 but it enables to properly classify class 4.

|  |  |  |  |
| --- | --- | --- | --- |
| Training Classification Report | Precision (%) | Recall (%) | F1-Score (%) |
| 1 | 68 | 70 | 69 |
| 2 | 80 | 70 | 75 |
| 3 | 89 | 91 | 90 |
| 4 | 47 | 37 | 41 |
| Accuracy |  |  | 83 |
| Macro Avg | 71 | 67 | 69 |
| Weighted Avg | 82 | 83 | 83 |

|  |  |  |  |
| --- | --- | --- | --- |
| Validation Classification Report | Precision (%) | Recall (%) | F1-Score (%) |
| 1 | 43 | 47 | 45 |
| 2 | 50 | 47 | 49 |
| 3 | 84 | 85 | 84 |
| 4 | 12 | 10 | 11 |
| Accuracy |  |  | 74 |
| Macro Avg | 47 | 47 | 47 |
| Weighted Avg | 73 | 74 | 73 |

|  |  |  |  |
| --- | --- | --- | --- |
| Testing Classification Report | Precision (%) | Recall (%) | F1-Score (%) |
| 1 | 54 | 49 | 51 |
| 2 | 48 | 63 | 55 |
| 3 | 84 | 88 | 86 |
| 4 | 23 | 13 | 17 |
| Accuracy |  |  | 76 |
| Macro Avg | 52 | 53 | 52 |
| Weighted Avg | 74 | 76 | 75 |

Training convolutional neural networks with a small unbalanced dataset is not an easy task. After implementing several strategies to build the model, due to the given dataset, the model overfits. Ideally, a larger dataset with more class representation would do the work. Another option would be to try and drop class 4, this might decrease the overfit and improve the validation scores.