1 Question 1: Image Classification

1.1 Data Pre-Processing

Images show the entire person, with the person roughly centred in the image. Given the task is concerned with torso colour and limited computational resources are available, images are cropped to focus on the torso region. Images are loaded at a size of 100×60 and cropped to the region (in numpy notation) `[20:60, 10:50, :]`, resulting in 40×40 images. These crop bounds are sufficiently conservative to ensure that none of the torso region is lost, though are sufficient to exclude a substantial portion of the image. No further resizing is performed, and images are left in colour given the task is recognition of clothing colour.

Figure 1: Data Samples: The top row shows the samples after cropping to focus on the torso. The bottom row shows an example of augmentation. Captions at the top of each column denote the class label.

Noting that very limited data is available (516 training samples, 190 test samples), data augmentation is also used. Random horizontal flipping, small random rotations (up to 5%), small random zooms (up to 5%) and small translations (of $\pm 2.5\%$ in horizontal and vertical directions) are used. All ground truth labels remain valid after these augmentations, Figure 1 shows examples of the cropped and augmented images input to the network.

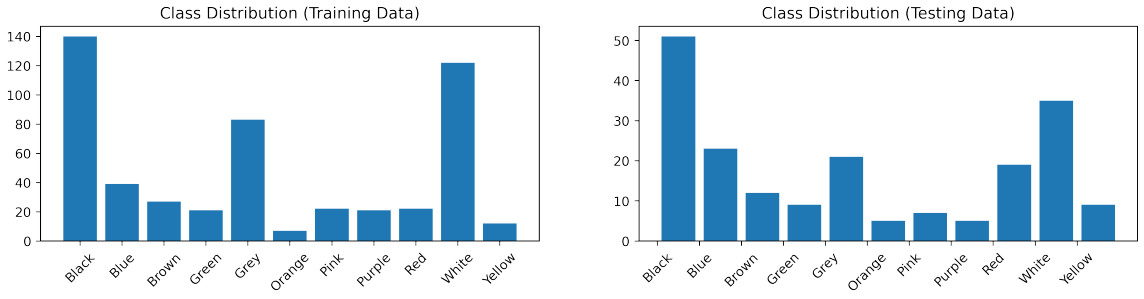

Figure 2: Class distributions of training (left) and testing (right) sets. Imbalance is present in both sets, though distributions across the two sets are similar.

Class balance issues are also identified, with the *Black* and *White* classes in particular being most common, as shown in Figure 2. While classes are imbalanced, an approximately similar class distribution is observed in both the training and testing split.

1.2 Network Design and Training

A ResNet network is used given the limited available data, lower parameters count obtained by ResNet due to the use of global average pooling prior to the classification layer, and increased representative power of ResNet when compare to VGG. Specifically, the `resnet_v1` function from the `CAB420_DCNNs_Example_5_ResNet.ipynb` example is used, with 3 residual stages with 16, 32 and 64 filters in each stage, and a single residual block in each stage. The output is an 11 neuron layer (for the 11 colour classes). In total, the network has 78,731 parameters. Appendix A shows a diagram of the implemented network.

A batch size of 32 is used, and the network is trained for 50 epochs using Adam, with weight decay ($\lambda = 1e-5$) used to improve training stability and help curb overfitting. The limited available data means partitioning off a validation set is not desirable, thus the network is simply trained for fixed number of epochs (determined experimentally). As noted in Section 1.1, class imbalance is present in the data and as such, class weights are also incorporated during training.

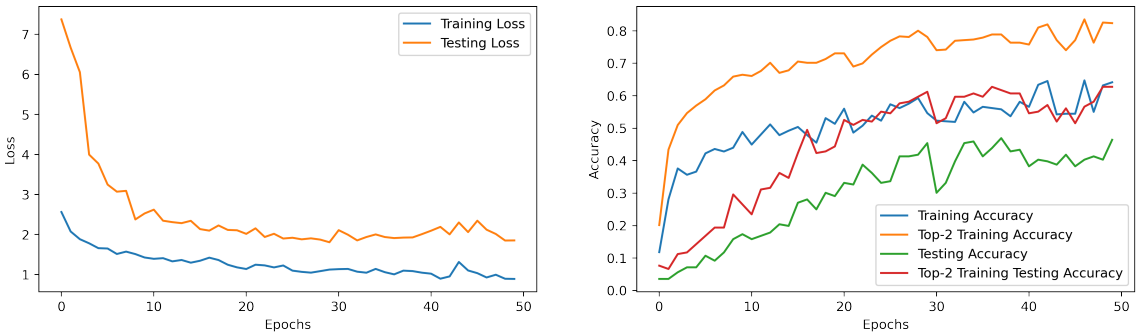

Figure 3: Model training performance, showing loss (left) and accuracy (right). Accuracy shows both top-1 and top-2 (i.e. the true value is one of the top two predicted classes) accuracy. Overfitting is observed, and training has converged after 50 epochs of training.

Figure 3 shows the results of model training. Over-fitting is observed, though it can be seen that training has roughly converged after 50 epochs.

1.3 Evaluation

Confusion charts for the training and testing sets are shown in Figure 4. Weighted F1 scores of 0.623 and 0.451 are obtained for training and testing sets. The impact of the class imbalance can be clearly seen in Figure 4, where the model clearly performs better on the more populous classes. The overfitting observed in Figure 3 is also clearly visible, with the performance on the testing set well below that of the training set.

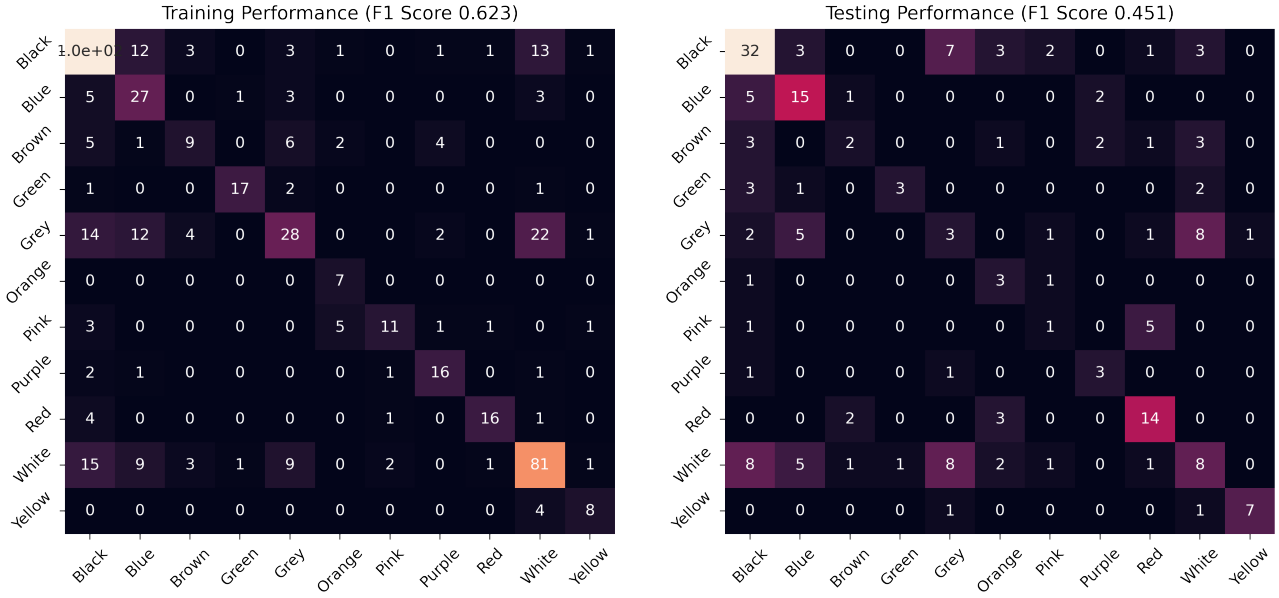

Figure 4: Confusion charts for training (left) and testing (right) sets. Overfitting is clearly visible, and confusion is often seen between similar colours (i.e. red and pink, black and grey).

Figure 5 explores failure cases, plotting the first 20 incorrect predictions on the test set. From this we observe that:

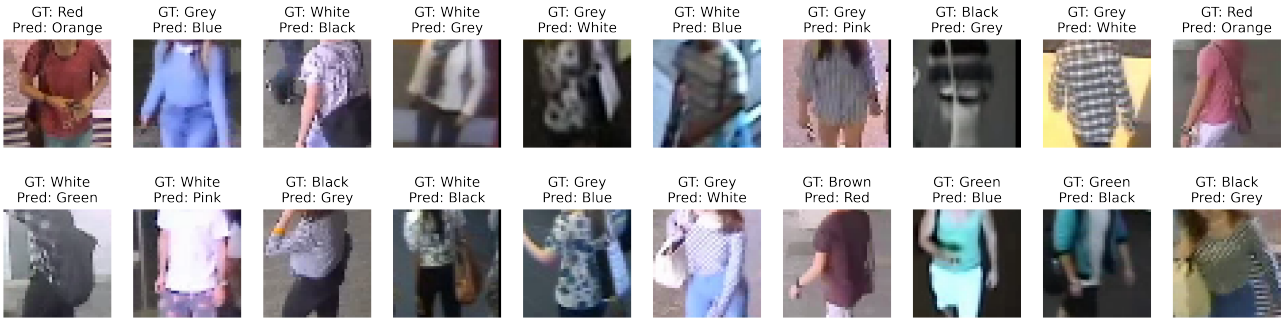

Figure 5: Failure cases on the test set. Ground truth and predicted classes are shown above individual images. Errors are caused by a number of factors including of similar colours being confused (first row, first column), incorrect annotation (first row, second column) and textured garments (top row, 9th column; bottom row, last column).

- Confusion between similar colours is common, for example mis-classifying red as orange, or white as grey. This is supported by the Top-2 accuracy reported in Figure 3, which shows the percentage of cases in which the correct prediction is in the top two most likely results.
- Some annotations are incorrect (see Figure 5, top row, second column), or ambiguous (such as examples with patterned shits in Figure 5).
- The approach struggles in situations where a garment contains multiple colours. Similar problems occur when other objects (such as bags) overlap the region of interest, obscuring part of the garment.
- Comparing Figure 1, which shows examples from the training set, and Figure 5 which shows errors on the test set, a domain shift between the two datasets is evident, with the training set containing darker, less vibrant, images. While the class distributions across the two datasets are similar (see Figure 2), the capture conditions vary. Similarly, the test set contains more samples with visible textures (i.e. chequered patterns) than is seen in the training set.

Overall, the poor performance arises from a mix of the small dataset, the need to train a model from scratch (rather than fine-tune an existing model), the ambiguities and errors in the dataset and the domain shift between the training and testing sets. While the use of data augmentation, weight decay and the ResNet model are able to help address issues pertaining to the small dataset, they are unable to compensate for challenges within the data itself.

A Network Diagram

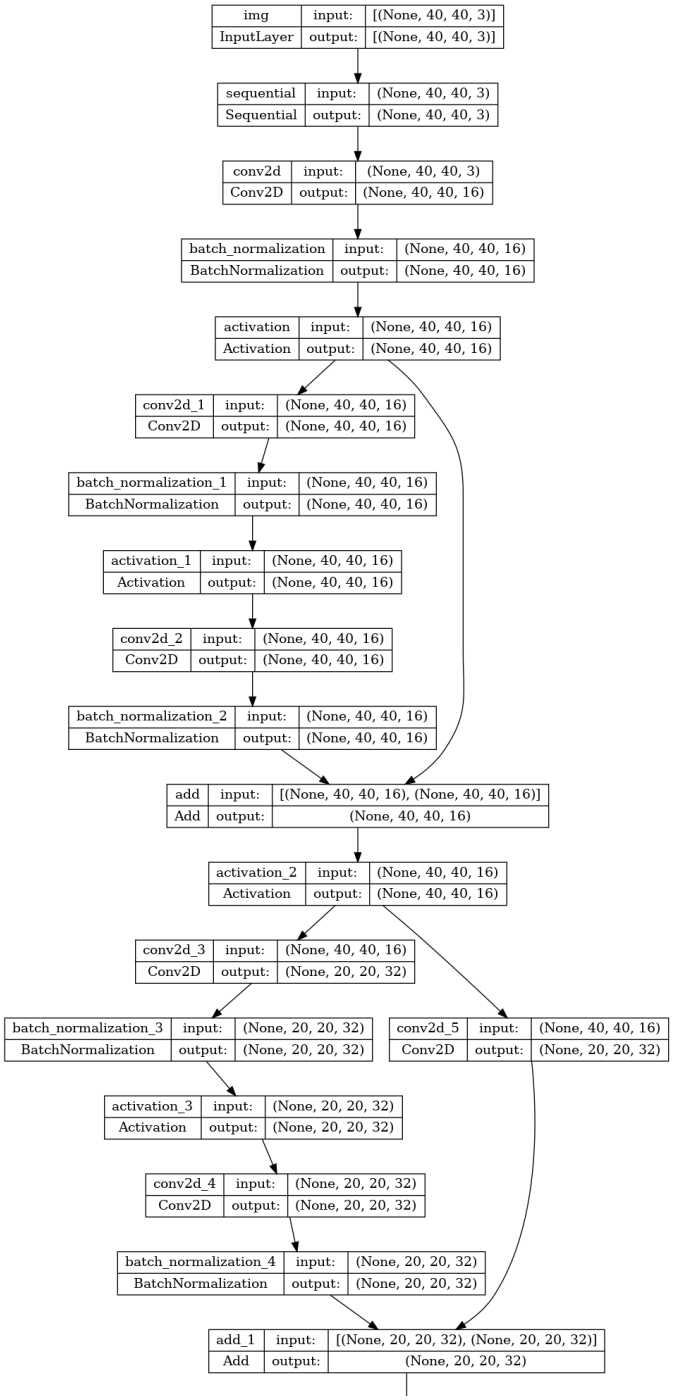

Figure 6: Model used in solution. Note model diagram is truncated and continues on the next page.

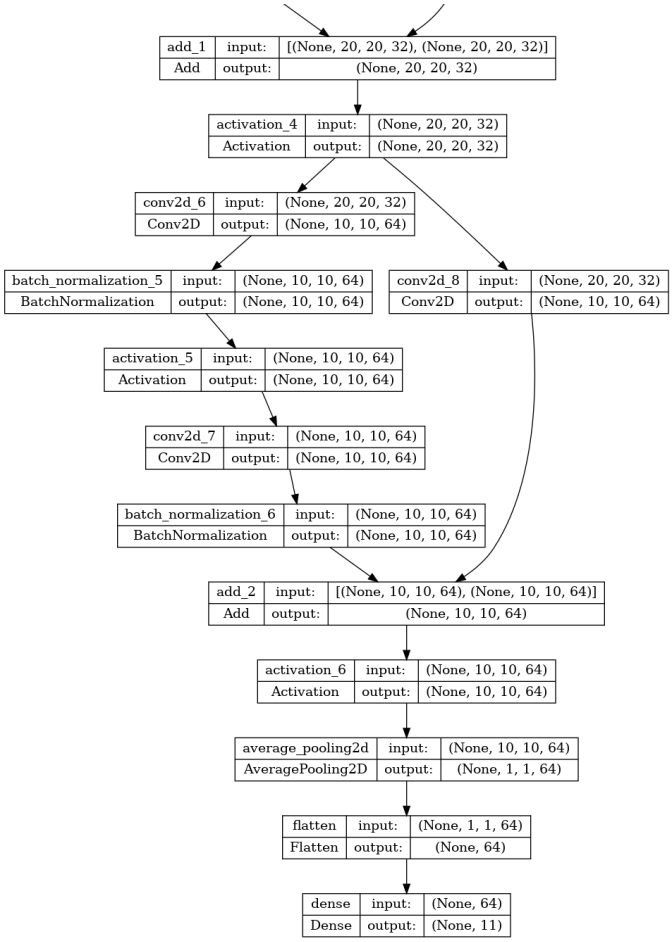

Figure 7: Model used in the solution. Model diagram continues from previous page, and the `add_1` layer is repeated in this portion of the.