**Assignment 2: Convolution**

The basic models demonstrated increasing accuracies with increasing training sample numbers in training samples of 1000, 2000, and 5000 while test size and validation size are kept constant at 500. To construct the model, all the above models were run with the following parameters of metrics as accuracy, optimizer as adam and loss function as binary\_crossentropy. Also optimizer rmseprop is used for pretrained model of training size 3000 to know better about the model’s performance

**Summary:**

* Training size of 1000, Test sample size 500 and validation sample size 500 showed loss value as 0.6202 and Accuracy as 0.6420. Throughout the course of epochs, training accuracy improved with each epoch.
* Training size of 3000, Test sample size 500 and validation sample size 500 showed loss value as 0.5253 and Accuracy as 0.7740. Throughout the course of epochs, training accuracy improved with each epoch.
* When training samples grew to 1000, 2000, and 3000, the base models' accuracies improved as well.
* As we have the highest accuracy training size of 3000, I have chosen the same training size to pretrain the model and run the model with data augmentation and without data augmentation.
* Among all the model’s run, pretrained model with data augmentation and training size of 3000 has the highest accuracy.
* As we have highest accuracy for 3000 training size I have changed the optimizer to rmseprop to know better about the model’s performance.

The model is run using various training sizes, both with and without data augmentation, and with a pretrained model (both with and without data augmentation) in the tabular format that is shown below.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Optimizer: Adam | | | | | |
| S.no | Training Set | Validation and Test Size | Data augmentation | Pretrained Model | Loss and Accuracy on Test |
| 1 | 1000 | 500, 500 | No | No | loss: 0.5659 - accuracy: 0.6980 |
| 2 | 2000 | 500, 500 | No | No | loss: 0.5604 - accuracy: 0.7360 |
| 3 | 3000 | 500, 500 | No | No | loss: 0.5253 - accuracy: 0.7740 |
| 4 | 1000 | 500, 500 | Yes | No | loss: 0.5944 - accuracy: 0.6840 |
| 5 | 2000 | 500, 500 | Yes | No | loss: 0.5832 - accuracy: 0.7280 |
| 6 | 3000 | 500, 500 | Yes | No | loss: 0.5433 - accuracy: 0.7600 |
| 7 | 3000 | 500, 500 | No | Yes | loss: 0.3322 - accuracy: 0.9955 |
| 8 | 3000 | 500, 500 | Yes | Yes | loss: 3.4010 - accuracy: 0.9720 |
| Optimizer: rmseprop | | | | | |
| 9 | 3000 | 500, 500 | Yes | Yes | loss: 3.4010 - accuracy: 0.9720 |

Conclusion:

The size of the training sample and the network selection for image classification are directly correlated. For image classification applications, deeper networks are needed to get improved performance as the training sample size grows. Larger training sample sizes further mitigate the problem of overfitting, enabling the construction of more complicated models without compromising their ability to generalize on unobserved data. Simpler networks can occasionally outperform more complex ones on image classification tasks, even with smaller training sample sizes. This may happen if there are few visual cues or patterns in the categorized photos that are simple for shallower networks to identify, or if the network design has been precisely tailored and optimized for the image classification issue.

Recommendations:

* Larger training sample sizes allowed for the extraction of more features and helped reduce overfitting.
* The approach of pre-trained convolutional neural networks trained on the ImageNet dataset significantly improved model's performance.
* With pre-trained CNNs and data augmentation, the model’s ability significantly increased accuracy especially with smaller training sample sizes. This implies that even with a small amount of training data, using pre-trained models can be advantageous.
* The choice of optimizer and hyperparameters can significantly impact training.