INT2 Open Assessment Report

Group 3: Yaseen Khan, Teddy Seddon, Johnathon Snelgrove, Rebecca Stone, Ellie White

***Abstract*—We present a model based on the fundamental concept of the depthwise separable convolution operation to satisfy the task of achieving high classification accuracy on the oxford_flowers102 dataset. An examination of literature surrounding this more recent form of convolution, as well as concepts such as residual connections introduced in other image classification architectures allowed us to create a deep, high performance architecture. Trained on the standard training split, our model was able to correctly classify examples at a rate of 65%.**

I. Introduction

IMAGE classification has been a fundamental problem in the field of computer vision, with numerous applications ranging from object recognition to medical diagnosis. In recent years, deep learning techniques, especially convolutional neural networks (CNNs), have proven highly effective in solving various image classification tasks [1]. One specific application of image classification is flower species recognition, which has significant implications in ecology, agriculture, and botany. The flowers-102 dataset [2] presents a challenge in classifying 102 different flower species with limited training data. Although various studies have addressed this problem using different machine-learning techniques, there remains room for improvement in classification accuracy. Historically, researchers have employed methods such as handcrafted feature extraction but with the advent of deep learning, CNN based approaches have emerged as the best performers.

The objective of this study is to develop a CNN architecture with a heavy focus on minimising computational cost, while maintaining performance and depth in order to accurately classify flower species from the flowers-102 dataset. We aim to consolidate modern network techniques and optimisation strategies into a single model which is able to fulfil the intricate requirements of classifying images into classes that can have very nuanced differences in features between them.

II. Method

The network is composed of 13 separable convolutional layers arranged sequentially with ascending filter counts, with each layer's output batch-normalised. Between these layers are ReLU layers to achieve non-linearity, and the overall sequential model contains the non sequential elements of residual connections. There are also several max-pooling layers arranged throughout the architecture, as well as a final global average pooling before the output layer. The loss function is sparse categorical cross entropy. The data was prepared for the network by cropping to a 1:1 aspect ratio, then resizing to 299x299 pixels, then normalising the pixel values between 0 and 1. The training set was augmented by 5 layers: a random flip, random rotate, random zoom, random brightness and a random translation; necessary steps due to the small number of training examples per class.

The foundation of the model is a series of sequential depthwise separable convolutional (DSC) layers. In a DSC, the convolution is divided into depthwise and pointwise convolutions. We chose this form of convolution because it is computationally less expensive and requires less memory than a normal convolution [3]. These properties allowed us to create a deeper model for the same computational demands than would be possible with regular convolutions. This property is also exploited in MobileNets [4]. Model depth is a key property for learning complex feature sets in images [5], especially for models with very slight differences in features between classes like those present in oxford_flowers102.

All of the convolutional layers are followed by batch normalisation due to the problem of covariate shift (identified by Ioffe and Szegedy [6]), which allows us to mitigate this effect from the internal activations of the network. This also removed the need for any dropout layers within the network. We chose ReLU layers for non-linearity because it is less computationally intensive than the sigmoid function [7]. In addition, as a neural network increases in size, the cost of adding ReLU layers scales linearly, which was a key consideration as increasing model depth was of importance. The filter sizes of the convolutional and DSC layers also increased through the model, this was based on the intuition that the model would progressively learn more and more complex patterns in the image which would have greater variation and would require more filters to capture their increasing complexity. The model begins with two normal convolutional layers because they result in a higher number of parameters compared to DSC, which aids learning, especially when the training set is limited.

The residual connections present in the architecture were based on the technique present in residual networks [8] which improves convergence as this allows some information to skip layers and generally improves training [8]. In our model some of these connections have 1X1 convolutional layers which are present to change the dimension of the tensors so they can be added. The max pooling layers reduce the number of parameters and computation to reduce overfitting, and Bera and Shrivastava [9] found it to outperform other pooling strategies which is why we chose it. We used small kernel sizes to reduce computational cost.

We used sparse categorical cross entropy as the loss function and Adam as the optimizer due to proven performance in multi-class classification with both as well as evaluations of alternative strategies (Fig. 3, 4). We used a learning rate of 0.001 which decayed on plateau to improve convergence in the later stages of the training process.

III. Diagram of network architecture

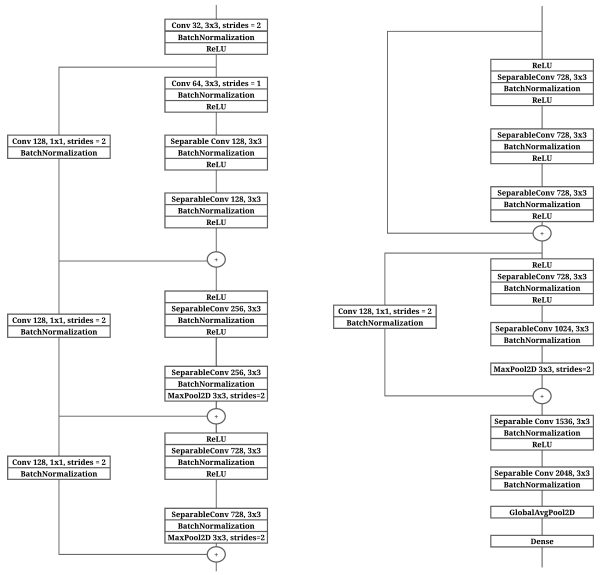

Figure 1: Illustration of the network architecture, no. of filters, kernel size and strides given where relevant. Add function given by +.

IV. Results and evaluation

In order to evaluate our architecture and the decisions made in our method, we undertook a rigorous process of evaluating different decisions both before and after each iteration of our model.

Experimental Results:

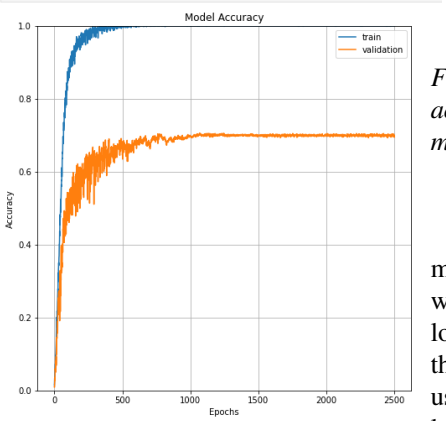

Figure 2: Validation accuracy of our final model

The evaluation metrics we chose were accuracy and loss, as measured on the test set. We also used accuracy and loss on the training and validation set to view the performance model in real-time. We utilised graphs to visualise and extrapolate model behaviour during experiments, as well as checking final metrics against the test data at the end of each experiment. We achieved a final test (top 1) accuracy of 65%, and a loss value of 4.34. This was achieved after a 9 hour model run over 2500 epochs on an NVIDIA RTX 4080 with 16GB of VRAM, connected via runpod.io to a Jupyter Notebooks instance. The framework used was TensorFlow. The time per epoch was 13 seconds.

Evaluating architectural decisions:

To validate the hypotheses and findings from literature made in our method, we constructed tests against our architecture (referred to as "Control" below). Each experiment was performed over 300 epochs on the same hardware and conditions as our final model. The evaluation functions and raw data outputs are included in the GitHub repository.

Model	Test Acc (%)
Control	54%
No Augmentation	28%
Shallow Model	25%
Constant Conv. Filter Size	36%
Stochastic Grad. Desc.	31%

Figure 3: Tabulated experimentation results

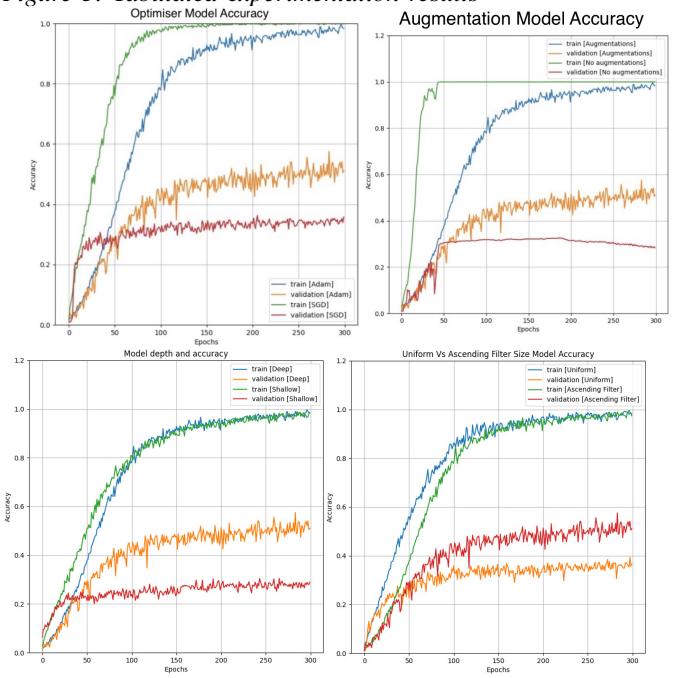

Figure 4: Parameter and hyperparameter evaluation

We thus verified the choice of the Adam optimiser over stochastic gradient descent and the importance of augmentations due to the limited size of the training set. The significance of depth in a network was also highlighted, as well as ascending filter size to capture complex patterns.

V. Conclusion and further work

We were pleased with the results considering the limited training time available as well as the very limited number of examples per class in *oxford_flowers102*. The architecture was founded upon a strong theoretical base which we empirically verified upon this data-set, and will serve as an effective foundation for further development. The model did suffer from over-fitting, and the loss did not decrease significantly, so these are behaviours of the model that we were not able to tune out during the assessment period. An area for improvement would be to explore more advanced image augmentation techniques, as we believe this is the key to unlocking even higher classification accuracy due to the limiting factor of the small training set.

REFERENCES

[1] Krizhevsky, A., Sutskever, I., Hinton, G. E, "ImageNet Classification with Deep Convolutional Neural Networks", *Neural Information Processing Systems*, 2012.

[2] Nilsback, M. E., Zisserman, A. Automated Flower Classification over a Large Number of Classes. *Indian Conference on Computer Vision, Graphics Image Processing*, 2008

[3] Chollet, F. Xception: Deep Learning with Depthwise Separable Convolutions, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017

[4] Howard, A. G. et al., MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications, *arXiv*, 2017

[5] Zhong, G. et al, From shallow feature learning to deep learning: Benefits from the width and depth of deep architectures, *Data Mining and Knowledge Discovery*, 2019

[6] Ioffe, S., Szegedy, C., Batch normalization: accelerating deep network training by reducing internal covariate shift, *Proceedings of the 32nd International Conference on International Conference on Machine Learning*, 2015

[7] Glorot, X. et al,, Deep Sparse Rectifier Neural Networks, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011

[8] He, K. et al., Deep Residual Learning for Image Recognition, *IEEE Conference on Computer Vision and Pattern Recognition*, 2016

[9] Bera, S., Shrivastava, V. K., Effect of pooling strategy on convolutional neural network for classification of hyperspectral remote sensing images, *IET Image Processing*, 2020