# Petals as Pixels: Exploring Segmentation in the Oxford Flower Dataset

Albert Haladay
*University of Nottingham*

***Abstract*—This project aims to classify the Oxford Flower Dataset into 'flower' and 'background' labels using semantic image segmentation. We will use two distinct approaches to create our Convolutional Neural Network (CNN), a custom-made approach with skip connections, and 8 convolutional layers and 6 ReLU layers, trained using information learned from only the dataset, and a transfer learning approach where we fine-tune pre-existing weights based on the DeepLabV3+ architecture and ResNet18's pre-trained weights. The best-performing model on the testing dataset was the transfer learning model with an F1 score of $0.9634$, compared to the custom model with an F1 score of $0.9186$. Explore with us as we gain insights into the uses of computer vision for agriculture and botanical research.**

## I. Introduction

In this project, we were provided with a dataset of images from the Oxford Flower Dataset [1]. The goal was to build the best-performing semantic segmentation network in MATLAB. Semantic segmentation is an area of computer vision that classifies every pixel of an image into a category. Since semantic image segmentation identifies regions on a pixel-perfect level, it can be much more precise in identifying what an image truly contains.

The task for our model is to classify every pixel as part of a flower or the background. This holds importance because flowers exhibit elaborate shapes and colours, making them a difficult task to classify in computer vision. Generally speaking, classifying flowers is a method for computers to interact intelligently with nature, which has many real-world applications, such as botanical research since it could be used to delineate between the different parts of a flower. Similarly, it could be used in agriculture to analyse the health of a crop of flowers or even robotically harvest them [2]. Classifying flowers has plenty of potential uses, from landscape analysis to the possible development of an artificial bee, but due to the many diverse shapes of a flower, the problem can only be solved with deep learning.

We explored two solutions to this task: a custom CNN and a CNN with pre-trained weights. For the custom model, we will create a CNN from scratch that downsamples and upsamples the image, then experiment with concatenation layers, regularization and gradient clipping. Whereas, the model built using transfer learning will explore the use of fine-tuning to adapt weights from a more powerful image recognition model and use them to classify our dataset.

## II. Methodology

In this section, we will discuss the process of creating our models, but before we begin we must preprocess the data, and examine the dataset.

### A. Data Preprocessing

After investigating the target and label images in our dataset, we can see that some glaring problems need fixing.

- There are 1360 targets and 846 label images. This means that there are only 846 images that we can use to train our model, and the rest must be discarded.
- The dataset also has seventeen ground truth labels for each labelled image, but this model aims to separate the 'flower' label from all the others, so we must reclassify each pixel as 'flower' or 'background'.

To remedy these issues, we start by loading all the existing labels into memory, and every pixel which does not contain the ground truth label of 'flower' was set to 'background'. We then save these label images in a new directory with their corresponding target image.

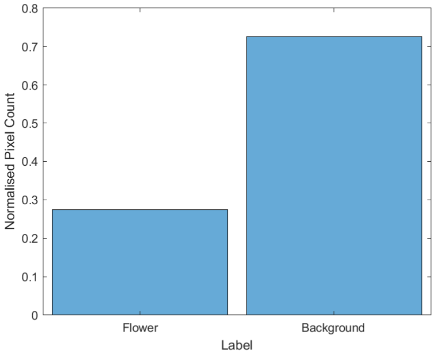

Fig. 1. The distribution of ground truth labels in the dataset.

After the data was cleaned, basic analysis in Figure 1 shows us that the dataset is heavily imbalanced, as there are significantly more background labels than flower labels. This means we have to balance the loss function and choose more balanced metrics when we evaluate the model.

### B. Model Architectures

In this section, we will describe the architectures for our semantic segmentation models, and the hyperparameters which govern the dimensions of the model and the operation of the model.

*1) The Custom Model:* The architecture of the custom model is shown in Figure 2. The model has an encoder-decoder type architecture, where the image is downsampled twice and upsampled twice. Concatenation layers were also used to add two skip connections, so less information from the encoding stage is lost when decoding, studies show that this helps train deep networks [3]. Throughout the model, six convolutional layers use 64 filters with a 3×3 kernel, since this will require fewer parameters, helping the model generalise better to unseen data. However, the two transposed convolutional layers have 8×8 kernels. There are also six ReLU layers, which are also used as outputs for the skip connections, and two max-pooling layers which scale the image down by a factor of four, and a stride of four. The final layer is a soft-max layer with two categories. In total, the custom model has approximately $300,000$ learnable parameters.

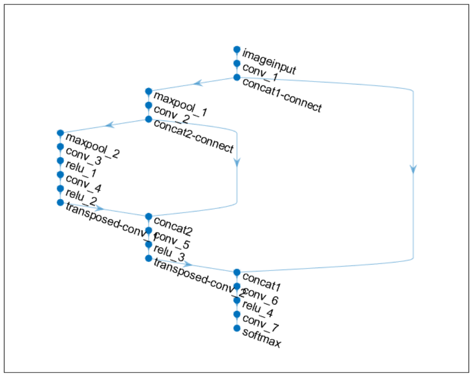

Fig. 2. The architecture of the custom model.

*2) The Transfer Learning Model:* The architecture of the transfer learning model is Google's DeepLabV3+ [4], this architecture is also an encoder-decoder type. It also has many improvements compared to the custom model, such as atrous spatial pyramid pooling, which applies dilated convolutions to learn the features at multiple scales, and depthwise separable convolutions, which combine information across channels by using a pointwise convolution between them.

The model will be initialised with pre-trained weights taken from another model, in our case that is ResNet18 [5], as it is a powerful model that has been trained on ImageNet which is a huge dataset. This allows ResNet18 to learn many generic features that will help the model generalise well. In total, this model has 20.6M learnable parameters, that begin as the weights from ResNet18.

### C. Training

Before any models can be trained, the cleaned dataset must be partitioned into training, validation and testing sets. Since there are only 846 entries in the dataset, we opted to use a $90\%$ training, $5\%$ validation, and $5\%$ testing split. We did this because we want to train the model on the greatest amount of data entries possible, with the expectation that it will help the model generalise to unseen cases more effectively. Since that dataset is heavily imbalanced, both models will use class-weighted categorical cross-entropy as the loss function to avoid bias in the model predictions.

*1) The Custom Model:* The custom model was trained using stochastic gradient descent with momentum, although it will take longer for the model to converge, some studies argue that it generalises better [6]. A learning rate schedule was used, starting with an initial learning rate of $0.005$, the learning rate is dropped by a factor of ten every three epochs. L2 regularisation was also used, with a coefficient of $0.01$, as a method to combat overfitting. Gradient Clipping was also implemented, with a threshold of $0.5$, as this will help prevent vanishing or exploding gradients, and possibly accelerate training [7]. A batch size of 16 was used, as the model had the highest performance with this size.

*2) The Transfer Learning Model:* The transfer learning model was also trained using stochastic gradient descent with momentum, as other studies have shown that combined with DeepLabV3+ and ResNet18, this method performs well [8]. It was trained similarly to the custom model, with the same regularisation, gradient clipping, and batch size hyperparameters. The learning rate schedule was modified to have a lower initial learning rate of $0.001$, and the learning rate drop factor was changed to five instead of ten.

## III. Evaluation

Since the dataset is extremely imbalanced, the metrics we use to evaluate our models must be robust to class imbalance, such as F1 score and balanced accuracy. The testing data reserved before our models were trained was evaluated using the custom and transfer learning models, the confusion matrices are shown in Figures 3 and 4 respectively.

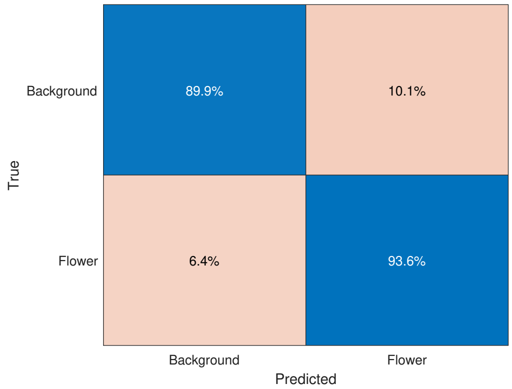

Fig. 3. The custom model's confusion matrix, using labels from the testing dataset and normalised using their true label.

From these confusion matrices, we can calculate the F1 score and balanced accuracy as $0.9186$ and $0.9177$ for the custom model, and for the transfer learning model, both metrics are $0.9634$. These matrices and metrics express that the transfer learning model performs qualitatively better than the custom model. The leading cause of this seems to be because

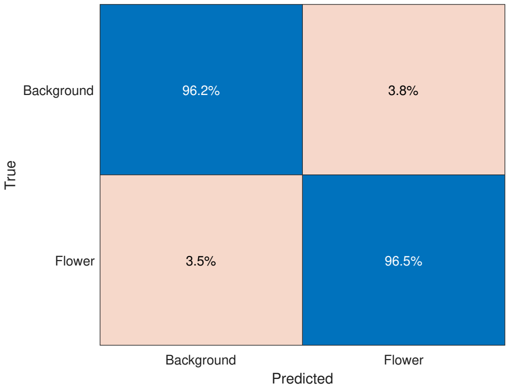

Fig. 4. The transfer learning model's confusion matrix, using labels from the testing dataset and normalised using their true label.

the custom model is classifying 10.1% of background labels as flowers.

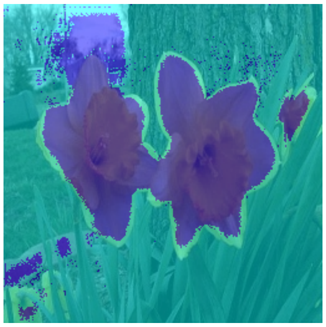

Fig. 5. The custom model results evaluated and overlaid on the example image.

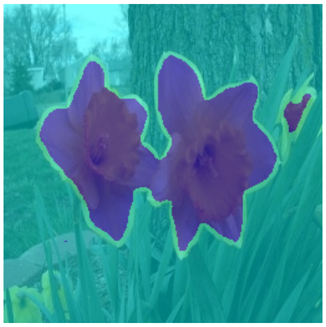

Fig. 6. The transfer learning model results evaluated and overlaid on the example image

This stands out even more in Figures 5 and 6, as the custom model has visible artefacts in its classification, whereas the transfer learning model has much less noise, and labels the daffodil test image in a more contained way.

Using the results extracted from the testing data and evaluation of the example image, it becomes clear that the transfer learning model outperforms the custom model as it has a qualitative difference in F1 score, and mean accuracy, and has visibly clearer regions. These artefacts in the custom model could likely be remedied by using larger kernel sizes, as this would expand the visual range of the model, allowing it to classify regions more accurately.

Figure 7 shows the training and validation F1 score of the custom model throughout its training. This can be seen to exhibit a rather unstable convergence, implying that the model may benefit from a lower learning rate to soothe this instability.

The training and validation F1 score of the transfer learning model during its training is visualised in Figure 8. This is converging more stably with significantly fewer oscillations than the custom model.

Figure 9 shows the training and validation history for the custom model, and it shows a sharp decrease in the loss and quickly appears to converge. This implies that the model does most of its learning in the earlier epochs, and could benefit from a lower learning rate or an increased model complexity.

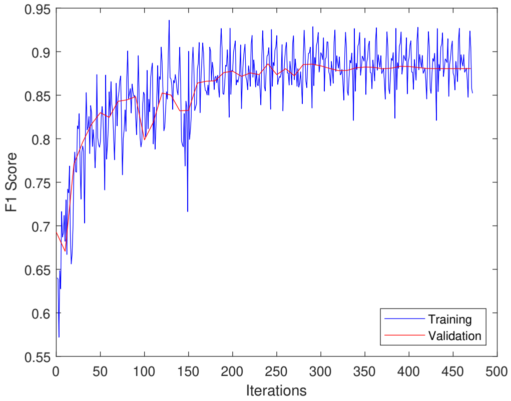

Fig. 7. The training and validation F1 score for the custom model.

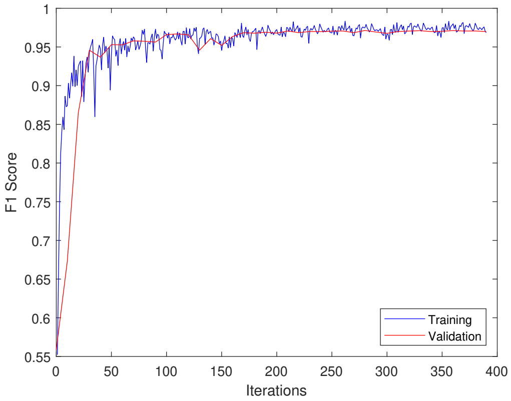

Fig. 8. The training and validation F1 score for the transfer learning model.

The training and validation history for the transfer learning model can be seen in Figure 10, and it shows a slightly more gradual decrease in the loss and appears to converge to a minimum, especially for a model with already trained weights. The stability of the loss and convergence of the model suggest that the weights are well-trained, and the model should generalise effectively to unseen data.

## IV. Conclusion

In summary, transfer learning was the approach with the best metrics when analysing the testing data, it also had clearer regions, and classified the labels to a standard that could apply to problems in the real world.

The transfer learning model could further be improved for real-world applications by adapting the training data to suit the specific task it needs to accomplish, for example, if the model needed to count the number of daisies in an area, like with using a quadrat, training the model on images of daisies could allow the model to see a significant increase in classification performance. If the task domain were to have a simpler dataset, the transfer learning model could meet a high enough standard to be used in botanical research or agriculture, as elaborated in Section I.

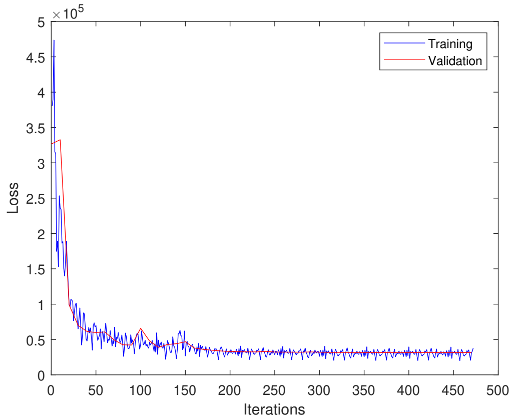

Fig. 9. The training and validation loss for the custom model.

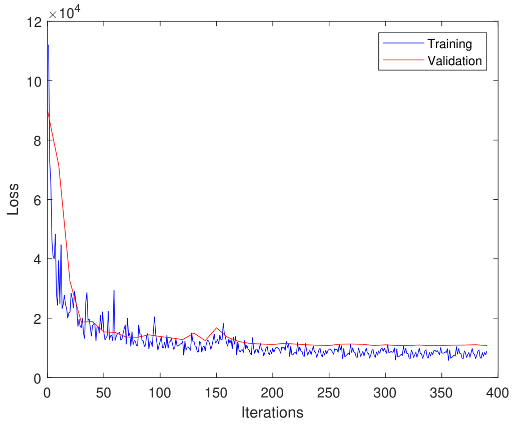

Fig. 10. The training and validation loss for the transfer learning model.

The custom model performed well, however, with an F1 score of 0.9186 and some visual artefacts, using the model in any task where classifying the regions is vital to the task's success becomes impossible.

The custom model could require further architectural improvements, such as increases in the depth of the model, or access to a broader training dataset. Another thought is that the learning rate needs to be lowered, and the schedule adjusted to improve training stability, as the F1 score is seen to wildly oscillate during training.

Overall, both models perform well, however, the transfer learning model outperforms the custom model by a noticeable margin, likely due to the generic features learned from ResNet18 and DeepLabV3+'s powerful semantic image segmentation architecture. The transfer learning model is a well-trained model with a grasp of botanical structures and could be used by machines to interact with various fauna and flora, making us one step closer to bridging the gap between petals and pixels.

## References

[1] M.-E. Nilsback and A. Zisserman, "Oxford flower dataset," https://www.robots.ox.ac.uk/~vgg/data/flowers/17/index.html.

[2] C. Narvekar and M. Rao, "Flower classification using cnn and transfer learning in cnn- agriculture perspective," in *2020 3rd International Conference on Intelligent Sustainable Systems (ICISS)*, 2020, pp. 660–664.

[3] A. E. Orhan and X. Pitkow, "Skip connections eliminate singularities," 2018.

[4] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," 2018.

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[6] M. Hardt, B. Recht, and Y. Singer, "Train faster, generalize better: Stability of stochastic gradient descent," in *International conference on machine learning*. PMLR, 2016, pp. 1225–1234.

[7] J. Zhang, T. He, S. Sra, and A. Jadbabaie, "Why gradient clipping accelerates training: A theoretical justification for adaptivity," *arXiv preprint arXiv:1905.11881*, 2019.

[8] M. Lin, S. Teng, G. Chen, J. Lv, and Z. Hao, "Optimal cnn-based semantic segmentation model of cutting slope images," *Frontiers of Structural and Civil Engineering*, vol. 16, no. 4, pp. 414–433, 2022.