FLOWER SEGMENTATION WITH U-NET AND RESNET

Alfie Rushby

ABSTRACT

This paper focuses on using two models to segment a small flower dataset. The first method involves using a simple U-Net architecture, and it achieved good performance, with a slight tendency to over-extend its flower boundaries. The second method uses a pre-trained ResNet model with 18 layers, and uses the first few layers to pre-train a U-net-like network. This method achieves better performance with tighter boundaries, but is noisier in its boundaries.

1. INTRODUCTION

This paper focuses on the aspect of two attempts to segment flowers in a small dataset called Oxford Flower Dataset [1]. The specific dataset is of around 750 flowers, with various species, where the images themselves each contain one flower. The difficulty, then, is to create a consistent solution to segment these flowers with the background. These flowers have varying viewpoints, colours, lighting, posing, shape, etc, and so the act of 'knowing' what a flower is with so much variance is very difficult.

2. LITERATURE REVIEW

This specific dataset has seen classical attempts at segmenting the flowers. The method uses an approximate segmentation with learned distributions for the background and foreground, and then it iterates on improving the foreground segmentation through using a geometric model of what a flower should look like, until it can converge onto some flower segmentation [2]. This specific method achieved its performance through a rudimentary learning method, which is by using a set of labelled sample images to create 'learned' distributions.

The method is affine invariant, but is still susceptible to colour/lighting variations. To circumvent this, a deep learning approach can be used to expand on the early uses of learning from data, specifically a U-net [3]. This expands on the principle by iteratively learning a set of filters that extract progressively more abstract features as you go down the first half of the network via lowering spatial information. The idea then is that these features would only encode the flower and background, or something useful, to then be upscaled with another set of filters into an output segmentation, specifically with up-convolutions.

This method removes the need for any assumptions on how a flower is structured, as the entire 'structure' and the concept of a 'flower' is learnt in the fast half. The only downside with this method is its dependence on data, where the quality and quantity of this data affects the performance to a great extent.

Another method, SegNet [4], exists, with the main differentiator being that it doesn't use convolutions to upscale directly, and uses pooling indexes to help with the upsampling. This isn't as popular as U-Net because it isn't as direct in learning the upscaling sequence with convolutions.

All of these methods requires data, and the learning of the encoder section (the downsampling that lowers the spatial resolution progressively) is similar to any classification CNN, like AlexNet or ResNet [5], [6]. Training these networks often uses methods like Batch Normalization, which prevents neuron outputs from spiralling into massive values [7], and Dropout, which make the network more robust and lessen overfitting by disabling parts of the network during training [8].

Smaller dataset training in particular has popular configurations for maximizing training efficiency. Along with what has been spoken of, Data Augmentation is another method to expand the dataset by applying affine transformations, along with possible colour variations. This has been shown to reduce overfitting [9] by increasing the model's generalization (through a larger dataset).

These methods can be used together in specific ways to maximize the usefulness in small dataset applications, where the additional concept of injecting this data augmentation technique in the middle of training can maximize its effectiveness [10].

It is also possible to transfer learn, or fine tune pre-trained models to use their more general convolutions in the encoder as a boost in training, where a pre-trained set of weights for the down-convolution can help it lock into a useful area of search on the onset. It has been shown that pre-training helps with generalizing performance by finding a better 'minima' in loss [11].

3. METHODOLOGY

3.1. Data pre-processing

Flower Label Mismatch. There exists flowers without labels, so these are deleted, resulting in 846 labelled flower images to work with.

Image Labelling. The dataset by default has more than 2 labels for the segmentation, and has a number of flowers that are unlabelled. The possible classes for the labelled pixels are null, flower, background, leaves, and sky. Null and flower are often just the inner flower and its boundary, which can be combined. The other labels are all 'background' and can also be combined into the background class.

These two modifications have been directly applied to the dataset with MATLAB and saved so that it doesn't need to be re-applied on every run of training.

Dataset Split. The dataset is split to 60:20:20, with it being the training, validation and testing set respectively, using random shuffling.

Image Resizing. For the transfer learning method, it is also important to note that the images and labels will be resized to 224x224 to fit the ResNet input dimension. This is done with MATLAB's resize algorithm.

Image Noise. The images of the flowers are also noisy due to the resizing algorithm used, and so a minor blur has been applied to smooth out the artefacts.

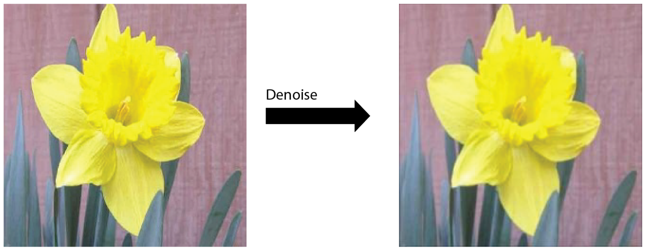


Fig. 1. Denoising of an image in the dataset

This method does add a blur to the image, but it smooths out the artefacts which seems to favour the training if you look at the **Evaluation** section.

Data Augmentation. Along with pre-processing, an active data augmentation method has been utilized during training, where a set of random affine transformations are applied to the images. This includes X and Y translations between -10 and 10 pixels, an X reflection, and a random scaling factor between 1 and 1.5.

3.2. Segmentation with a 'from scratch' model

From what has been discussed in the literature review, a U-Net along with Batch Normalization and Dropout has been chosen:

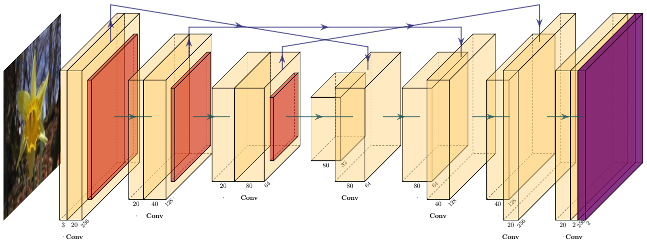


Fig. 2. Proposed U-Net Model Architecture

Relu and batch-normalization layers have been left out of the diagram to save space, but you can assume that every up-convolution has a batch-normalization and every convolution has a ReLu layer.

The network was trained with a maximum epoch of 256, with a linear learning rate of 0.001 with the 'Stochastic Gradient Descent with momentum' optimizer. The simple method was chosen so that a focus could be put on the architecture, as there was not a need for optimizing the time taken to train for such a small model.

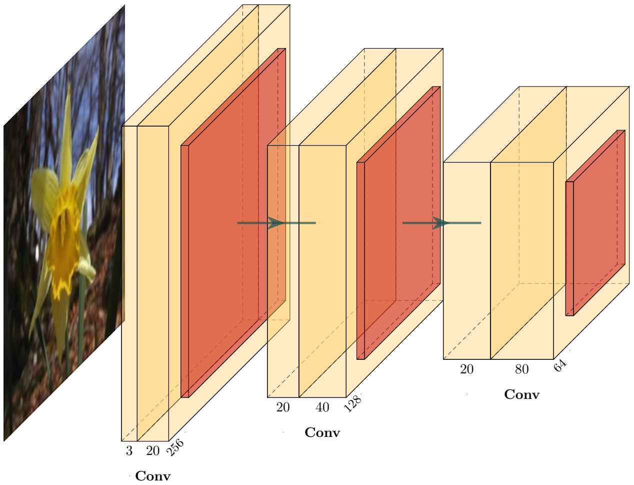


Fig. 3. U-Net Encoder Architecture

The encoder has been set up to start by creating 20 filters from the first convolution, without decreasing the spatial resolution. From this, it follows the pattern of doubling the filters and halving the resolution to 32x32 pixels.

No more layers were added as it added complexity and made training difficult. It uses 3x3 filters all the way through, with the lowering spatial resolution negating the need for larger ones.

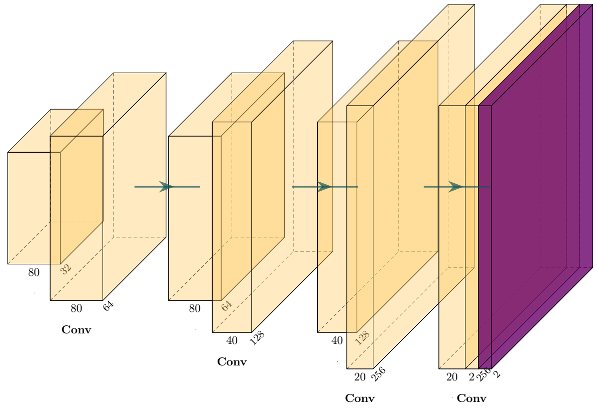


Fig. 4. U-Net Decoder Architecture

The decoder works similarly, except that it uses convolutions to upscale, not max-pooling. The end layer is a softmax layer. It uses 4x4 filters.

3.3. Segmentation with a ’Pre-trained’ model

Resnet-18 [12] was used as the base to pre-train off of. This was chosen because it is a relatively small classifier trained on a 1000 class image dataset [6]. It is hypothesized that the first few layers are useful from this to fine tune towards the flower segmentation problem.

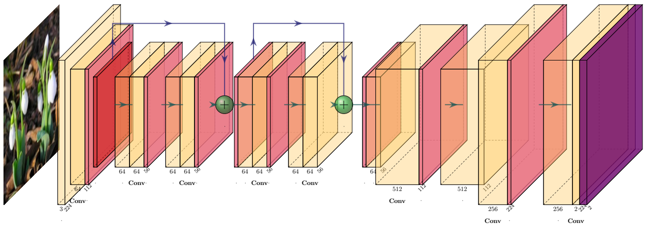


Fig. 5. Proposed ResNet + U-Net Model Architecture

As ResNet is a classifier a number of modifications were done to make it segment images. A number of the ResNet convolutions were removed to make training easier, leaving just two sets of branched ’layers’. The pink layers are ReLu layers, and they are included because they hold a more important role here. Batch Normalization is not shown, but appears after every convolution in the downsample area. The U-Net skip connections can be assumed.

Most of the ResNet layers were removed to save on training time and memory, as the model did not show any signs of improving without removing most layers.

An interesting quirk from this is that there is only one pooling of downsampling, which is at the start. It also halves the resolution at the start with a convolution. This is due to removing the ResNet layers past the 2 pairs of convolutions.

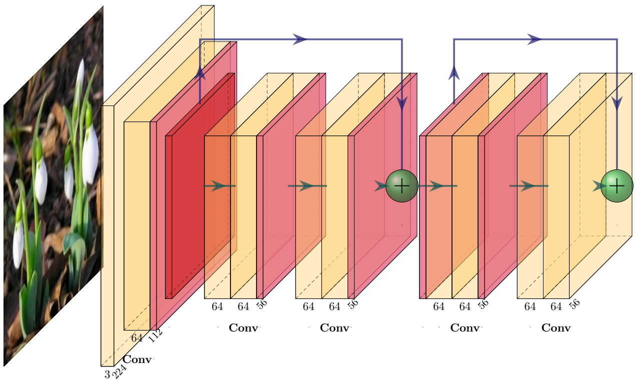


Fig. 6. ResNet + U-Net Encoder Architecture

There are two skip connections that sum the previous convolution outputs, where it is done in pairs. This part is taken from ResNet-18 and is pre-trained.

When training, the learning rate of these layers was set to 0.4. Instead of stopping learning completely, a smaller learning rate will allow the training to adapt the first layers, and with the fact that they are already trained from another dataset, gain an advantage when learning.

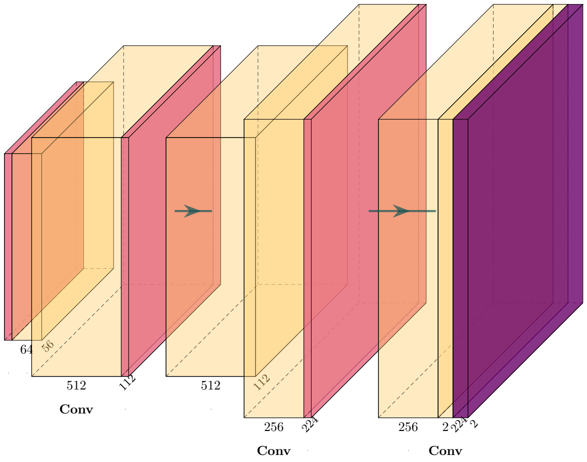


Fig. 7. ResNet + U-Net Decoder Architecture

The decoder follows a similar structure to the U-Net used in the ’from-scratch’ model. The main difference is there is one less convolution pass, as the dimension inputted to it only requires 2 2x2 filter passes on up-convolutions, from 56x56 to 224x224. The filter sizes were chosen to accommodate the input dimension. All of these layers have initialized weights as they are added onto the ResNet network.

4. EVALUATION & DISCUSSION

The paper will be comparing the two methods directly.

Evaluation Datasets. Evaluation of all methods are done using the Oxford Flower Dataset [1], on an unseen set of 169 flower images.

Implementation Details. The model was trained using a GTX 970 GPU, and follows the implementation details show in the methodology. It is set to train for 128-256 epochs and returns the best model according to the validation set's accuracy.

4.1. 'From scratch' model

Mean Accuracy	Mean IoU	Mean BF Score
90.61%	82.44%	58.31%

The high IoU score means it is good at not classifying background pixels as flowers, but its low boundary F1 (BF) contour matching score means that it doesn't align with the true boundary well. It tends to overextend from the flower.

From this the specific classes are:

Class	Accuracy	IoU	Mean BF Score
Flower	90.01%	79.80%	46.02%
Background	90.95%	85.08%	70.60%

This shows that its major weakness is properly adhering to the flower boundaries as shown with the flower's low BF score.

Its confusion matrix is:

Class	Flower	Background
Flower	90.13%	9.86%
Background	9.04%	90.95%

The confusion matrix shows a rather even confusion with flowers and background. Around 10% of each class is confused, and this shows there is not a great disparity in its handling of both classes. **Fig 8** shows this, where it has quite thick boundaries.

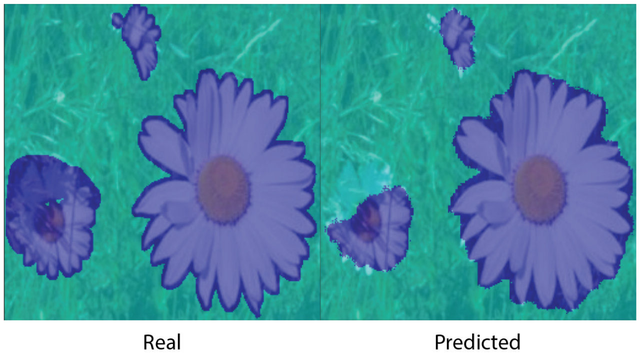


Fig. 8. Hard Example Image

4.2. 'Pre-trained' model

Mean Accuracy	Mean IoU	Mean BF Score
92.22%	85.2%	73.96%

Compared to the 'from scratch' model it achieves a much higher BF score, along with marginal improvements in average and IoU. This means that it should be much tighter in segmenting the flowers, with less excess boundary.

From this the specific classes are:

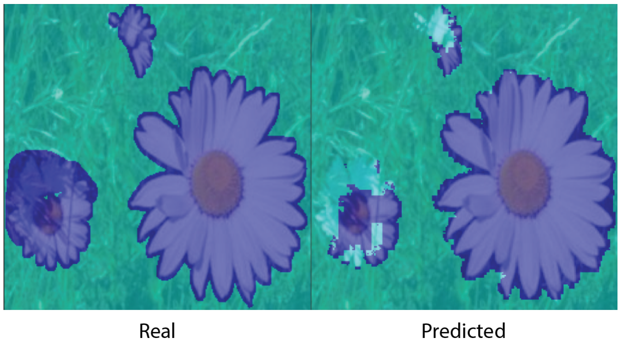
Class	Accuracy	IoU	Mean BF Score
Flower	92.94%	83.1%	67.78%
Background	91.73%	87.43%	80.147%

This shows a similar picture, with the BF score improving in both aspects.

Its confusion matrix is:

Class	Flower	Background
Flower	92.94%	7.05%
Background	8.26%	91.73%

The boundaries are indeed tighter, but a certain amount of noise is present in separated flowers.



An ablation study with the 'from scratch' model shows an improvement with blurring the images to remove noise.

Setup	Mean Accuracy	IoU	Mean BF Score
*	89.90%	81.686%	58.34%
Dropout	89.86%	81.98%	57.31%
Dropout+Blur	90.5%	82.4%	58.31%

5. CONCLUSION

The pre-trained model performs better in almost all aspects, except for the fact that it is more noisy in certain scenarios. This noise may be due to the ResNet downsampling not removing sufficient spatial resolution. It only pools once with a downsampling convolutional network with the only other layer that lowered spatial resolution being a convolution. This might entail a less abstracted set of filters which can make it more sensitive to image changes.

6. REFERENCES

[1] M.-E. Nilsback and A. Zisserman, "A Visual Vocabulary for Flower Classification," presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, Feb. 1, 2006, pp. 1447–1454, ISBN: 978-0-7695-2597-6. DOI: 10.1109/CVPR.2006.42.

[2] M.-E. Nilsback and A. Zisserman, "Delving deeper into the whorl of flower segmentation," *Image Vision Comput.*, vol. 28, pp. 1049–1062, Jun. 1, 2010. DOI: 10.1016/j.imavis.2009.10.001.

[3] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds., Cham: Springer International Publishing, 2015, pp. 234–241, ISBN: 978-3-319-24574-4. DOI: 10.1007/978-3-319-24574-4_28.

[4] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation.," Dec. 2017, ISSN: 0162-8828. [Online]. Available: https://www.repository.cam.ac.uk/handle/1810/271007 (visited on 05/05/2024).

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems*, vol. 25, Curran Associates, Inc., 2012. [Online]. Available: https://papers.nips.cc/paper_files/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html (visited on 05/05/2024).

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90. [Online]. Available: https://ieeexplore.ieee.org/document/7780459 (visited on 04/29/2024).

[7] S. Ioffe and C. Szegedy. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift." arXiv: 1502.03167 [cs]. (Mar. 2, 2015), [Online]. Available: http://arxiv.org/abs/1502.03167 (visited on 04/30/2024), preprint.

[8] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014, ISSN: 1533-7928. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html (visited on 04/30/2024).

[9] C. Shorten and T. M. Khoshgoftaar, "A survey on Image Data Augmentation for Deep Learning," *Journal of Big Data*, vol. 6, no. 1, pp. 1–48, 1 Dec. 2019, ISSN: 2196-1115. DOI: 10.1186/s40537-019-0197-0. [Online]. Available: https://journalofbigdata.springeropen.com/articles/10.1186/s40537-019-0197-0 (visited on 05/05/2024).

[10] P. Thanapol, K. Lavangnananda, P. Bouvry, F. Pinel, and F. Leprévost, "Reducing Overfitting and Improving Generalization in Training Convolutional Neural Network (CNN) under Limited Sample Sizes in Image Recognition," in *2020 - 5th International Conference on Information Technology (InCIT)*, Oct. 2020, pp. 300–305. DOI: 10.1109/InCIT50588.2020.9310787. [Online]. Available: https://ieeexplore.ieee.org/document/9310787 (visited on 04/29/2024).

[11] D. Erhan, A. Courville, Y. Bengio, and P. Vincent, "Why does unsupervised pre-training help deep learning?" In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, Y. W. Teh and M. Titterington, Eds., ser. Proceedings of Machine Learning Research, vol. 9, Chia Laguna Resort, Sardinia, Italy: PMLR, 13–15 May 2010, pp. 201–208. [Online]. Available: https://proceedings.mlr.press/v9/erhan10a.html.

[12] "Resnet-18." [Online]. Available: https://uk.mathworks.com/help/deeplearning/ref/resnet18.html.