COMPARATIVE ANALYSIS OF SEGMENTATION PERFORMANCE UTILISING DEEPLABV3+ AND A LIGHTWEIGHT ENCODER-DECODER NETWORK

Boey Chun Hong

University of Nottingham, School of Computer Science

ABSTRACT

This paper dives into the usages of deep learning Convolutional Neural Networks (CNN): DeepLab v3+ and a CNN that was inspired by the architecture of U-Net called Floral-Net, tasked at segmenting flower and background of images from the Oxford Flower Dataset. On top of this, the DeepLab v3+ networks had its weights initialised from the following list of pre-trained networks: ResNet18, ResNet50 Each variant of the DeepLab v3+ network would be fine-tuned and trained on the Oxford Flower Dataset.

The different variations of DeepLab v3+ networks and Floral-Net were evaluated with the following metrics: Pixel Classification Accuracy, Intersection over Union (IOU), MeanBFScore, Confusion Matrix. Additionally, the best and worst instances of flower and background segmentation on test images were visualised, based on the IOU score, to scrutinise the strengths and weaknesses of each network. Pre-trained DeepLabv3+ networks with initialised weights from ResNet 18 and ResNet 50 produced excellent results in precisely classifying flower and background pixels, with an accuracy of 98.35% and 99.61%, respectively. On the other hand, Floral-Net had a pixel classification accuracy of 97.8% for background pixels and 96.5% for flower pixels.

Index Terms— Convolutional Neural Networks (CNNs), DeepLabv3+, U-Net, ResNet 18, ResNet 50, Image Segmentation.

1. INTRODUCTION

The problem statement is as follows: Given a subset of the Oxford Flower Dataset (17 Classes) , the goal of the proposed models is to segment between two classes: flower and background. Therefore, this research would delve into comparing the segmentation capabilities between the various DeepLab v3+ networks and Floral-Net. The dataset includes a variety of flower images captured in different lighting conditions and environments across the United Kingdom (UK). Continuing from here, several flower species (e.g.: Buttercup, Colts Foot, Daffodil, Daisy etc.) are present in the dataset. There have been several approaches to the problem statement of flower segmentation. For instance, Saha et al., 2020 ventured into a quantitative evaluation surrounding a mix of pretrained supervised and unsupervised networks: Variants of FCN-VGG16, ResNet18 and their novel approach. The authors split the Oxford Flower Dataset to 80-20 for the supervised networks. For the unsupervised networks, the authors did not split the data due to the nature of the network, as it learns directly from the input images without relying on labelled data for training [1]. Following this, transfer learning was employed on the pretrained supervised networks using the aforementioned data split. The highest results of each variant were as follows: 95.75% for ResNet 18, 97.23% for FCN-VGG16 and 96.01% for their novel network. Despite the excellent results produced from the networks, the authors did mention the issue of over-segmentation [1]. The subsequent study by Yong et al. demonstrated the capability of classification accuracy of ResNet 50. Like the aforementioned study, the authors applied transfer learning to ResNet 50, training and evaluating the network on the Oxford Flower Dataset 17. The results were promising, scoring a classification accuracy of 95.29% [2].

The last piece of literature revolves around the introduction of U-Net. The architecture of U-Net included encoders and decoders that upsamples and downsamples the image depending on the layer. Each encoder layer comprises of 3x3 convolutions followed by rectified linear unit (RELU) and ends with a 2x2 max pooling operation with stride 2 [3]. The decoder layers starts with upsampling the feature map, followed by a process of up-convolution with the same parameters as in the encoders [3]. The bottleneck acts as a bridge between the encoder and deocder layers, reconstructing details from the encoder layers [3]. The network included skip connections to ensure that no important details were lost during training whilst moving forward to the decoder layers [3]. The network was trained on images of HeLa cells recorded with Differential Interference Contrast. Consequently, the model was evaluated on the test portion of the dataset, as well as an additional PhC-U373 dataset which contained human glioblastoma cells [3]. This resulted in the network scoring a 92.03% on the PhC-U373 dataset, and 77.56% on the DIC-HeLa dataset, in terms of IOU accuracy [3].

2. METHODOLOGY

The given subset of the Oxford Flower Dataset (17 Classes) was imbalanced , several images were misssing its truth labels. Due to the nature of supervised learning networks that is present in this research, it was essential to remove any images that did not truth labels. In brief, the algorithm detects all files from the specified paths of images and truth label files, and strips their respective extensions from the file. Utilising MATLAB's setdiff function, the files that were present in the images folder but not in the label folder were deleted. This resulted in the augmented dataset of 846 images and truth label files. No additional preprocessing was done to the image prior to the creation of image datastores and pixel datastores, as the required dimensions of the input layers for all networks were the same as the dimensions of the images. The pixel label datastore was initialised to only store the truth labels of the classes: flower and background, which in this case corresponds to label 1 and 3.

```
Algorithm 2 Image and Pixel Label Datastore Splitting and Combining
1: Shuffle indices
2: Calculate set sizes based on training, validation and test split ratio
3: Split indices into training, validation, and test sets
4: Create image datastore splits for training, validation and test sets
5: Create pixel datastore splits for training, validation and test sets
6: Combine datastores based on their respective sets
```

Fig 1: Algorithm to Split and Combine Image and Pixel Datastores

Derived from MathWorks tutorial on semantic segmentation, the algorithm shown in Figure 1 was used to randomly split the image and pixel label datastores into 70% for training , 20% for validation and 10% for test sets [4]. Contuining from this, the ratio adopted in this research shadows a commonly adopted split ratio in studies that adapted machine learning in their methodology [5], which have shown the effectiveness of this split ratio. From here, the respective train, validation and test image and pixel label datastores were combined.

The DeepLab v3+ network would be instantiated using the "deeplabv3plus" function that is available from the MATLAB's Computer Vision Toolbox [6]. In the subsequent phase, **part one** of the experimental setup would be as follows: ResNet 18 would be initialised as the backbone network for the initial step, followed by ResNet50 in the subsequent step. During each iteration, the DeepLabv3+ network would be initialised with the weights of the assigned network, following the procedure of transfer learning. The training options, shown in Table 1, was also derived from MathWorks tutorial on semantic segmentation [4], with the only difference being the MaxEpochs which defines the training duration.

Table 1: Training Options used in experimental setup **part one and part two**

Option	Value
Optimization Algorithm	sgdm
Learning Rate Schedule	piecewise
Learning Rate Drop Period	6
Learning Rate Drop Factor	0.1
Momentum	0.9
Initial Learning Rate	1e-2
L2 Regularization	0.005
Validation Data	dsVal
Max Epochs	10
Mini-Batch Size	4
Shuffle	every-epoch
Checkpoint Path	tempdir
Verbose Frequency	10
Validation Patience	4

Option	Value
Optimization Algorithm	Adam
Initial Learning Rate	0.001
Learning Rate Schedule	Piecewise
Learning Rate Drop Factor	0.1
Learning Rate Drop Period	10
L2 Regularization	0.0001
Maximum Epochs	10
Mini-Batch Size	16
Data Shuffling	Every Epoch
Validation Data	dsVal
Validation Frequency	10
Plots	Training Progress

Lastly, "trainnet" was utilised to initiate the training process for each step of the experimental setup. **Part two** of the experimental setup revolves around constructing a unique approach for the problem statement. This research encompasses a CNN called Floral-Net which follows the structure of U-Net's encoder and decoder blocks with modifications and additions done to the layers, and then removing all the skip connections. This results in an encoder-decoder network based on the tuned parameters of U-Net's encoder and decoder layers.

Floral-Net mirrors the ideas of improvement from two specific papers. Starting off with Xiao-Yun et al.'s research, which introduced the idea of adding a batch normalisation layer inbetween the set of 3x3 convolutions and RELU layers [7]. This resulted in a higher Dice Similarity Coefficient (DSC) whilst segmenting several medical images. The addition of a dropout layer is credited to Li et al.'s research, the authors proposed to add dropout layers to a type of U-Net called MResU-Net. The exact location was after each batch normalisation layer in the residual blocks [8]. Floral-Net adapts this idea, but dropout is added as the last operator of each encoder and decoder block. In Floral-Net incremental and decremental dropout probabilities are intialised, the dropout probability would increment by 0.1, and would reset once it reaches the bottleneck block. From the bottleneck to the decoder block, the dropout probability would reset to 0.4, and would decrement by 0.1, up until the final output layer. This ensures that the network is not overfitted whilst extracting higher-level features.

Regular U-Net architectures would follow a certain pattern, where after every downsampling step, the number of filters of any convolutional operator would double. On the contrary, after every upsampling step, the number of filters will halve. Whereas in Floral-Net, only the last two encoders would have their filters double in numbers, this trend in consistent in the decoders as well, where the last two would have their number of filters halve in numbers, capturing more details whilst reducing the chances of the network learning an unimportant feature. The bottleneck would maintain the number of filters that is present in the last encoder block. Floral-Net removes all skip connections as the chances of details being lost is low, due to the small size of the dataset. The upsampling operator would be removed from the first decoder block, the following decoder block would be reinitialised with 64 filters, halving in the block and quadrupling in the last decoder block for reduced complexity. The last difference can be observed in the pixel classification layer, which was initialised with the class weights that was derived by calculating the inverse of the frequency of occurrence for each class. This mechanism compensates for the class imbalance that can occur due to overpowering number of background pixels compared to flower pixels [9]. To construct this architecture, a layerGraph object was intialised, adding the encoder, bottleneck, decoder and final layers, connecting and creating a network graph. From here, trainNetwork function would be used to train the compilted network graph with the options stated in Table 1 (**part two**).

3. EVALUATION

The networks in part one and two of the experimental setup would be evaluated with the following metrics: pixel classification accuracy, IoU, MeanBFScore, Confusion Matrix, Image Mean IoU across the test set. To spot the weaknesses and strengths, the best and worst predicted images would be visualised, based on the mean IoU score. Classifcation accuracy is defined by:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

This metric calculates the ratio of the number of correctly predicted positive cases and the total number of real positive cases [10]. Therefore in this case, pixel classification accuracy would be the ratio of the number of correctly predicted positive pixels and the total number of real positive pixels for each class: flower and background. Contuining, IoU is defined by:

$$\text{IoU} = \frac{\text{TP}}{\text{TP} + \text{FP} + \text{FN}}$$

IoU calculates the the ratio of accurately classified pixels to the total count of ground truth and predicted pixels within that class, and it penalises wrongly classified pixels [11]. Boundary F1 Score or BFScore is defined by:

$$\text{BF score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

The goal of this metric is to find a good balance between precision and recall, which concludes that the model is able to effectively classify pixels between the two classes within the boundary region [11]. On the other hand, a confusion matrix, is a N x N matrix used to assess the effectiveness of a classification model, where N is the number of target classes. The matrix compares truth pixel labels to those predicted by the network. Mean IoU would represent the average IoU score among all classes in an image [11]. Segmentation inference would be performed on images present in the test set. From here, the mean IoU would be recorded for each image and would be represented in a histogram graph.

4. RESULTS AND DISCUSSIONS

Table 2: Performance of Networks on various metrics.

Network	Accuracy		IoU		BFScore	
	Flower	Background	Flower	Background	Flower	Background
DeepLabv3+ Initialised with ResNet 18 weights	98.35%	99.61%	97.48%	98.91%	87.54%	97.66%
DeepLabv3+ Initialised with ResNet 50 weights	98.63%	99.63%	97.77%	99.03%	89.01%	98.28%
Floral-Net	96.64%	97.83%	91.84%	96.36%	81.15%	93.09%

Table 3: Confusion Matrices for Trained Networks

DeepLabv3+ Initialised with ResNet 18 weights

		Predicted Class	
		Background	Flower
True Class	Background	99.6%	0.4%
	Flower	1.6%	98.4%

DeepLabv3+ Initialised with ResNet 50 weights

		Predicted Class	
		Background	Flower
True Class	Background	99.6%	0.4%
	Flower	1.4%	98.6%

Floral-Net

		Predicted Class	
		Background	Flower
True Class	Background	97.8%	2.2%
	Flower	3.5%	96.5%

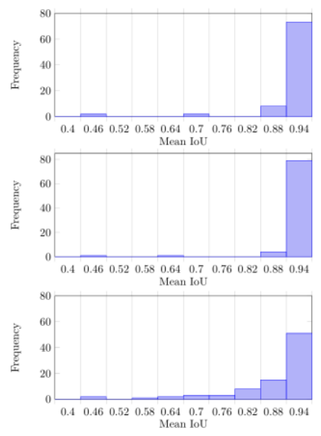

Fig 2: Image Mean IOU of ResNet 18 (Top), ResNet 50 (Middle) and Floral-Net (Bottom)

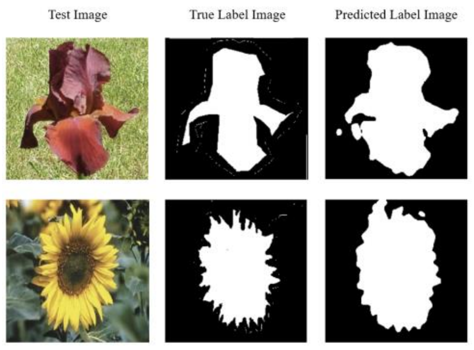

Fig 3: Image Mean IoU of Best and Worst Predicted Label Images from DeepLabv3+ (ResNet 18)

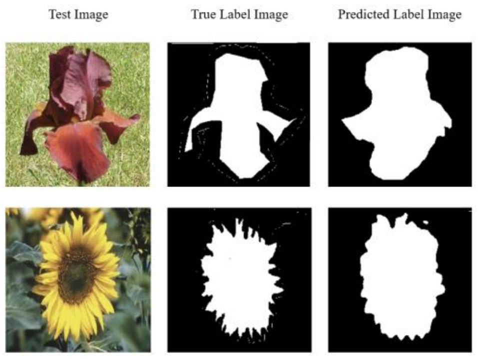

Fig 4: Image Mean IoU of Best and Worst Predicted Label Images from DeepLabv3+ (ResNet 50)

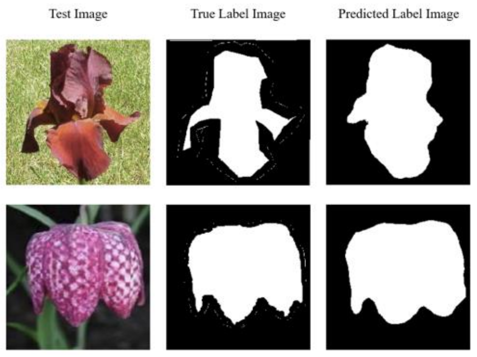

Fig 5: Image Mean IoU of Best and Worst Predicted Label Images from Floral-Net

The results of the aforementioned metrics can be visualised in Table 2. There is a distinctive trend, where the scores would be higher for background than flower, this is due to the higher pixel count for background than flowers. Floral-Net had the greatest disadvantage from the pixel class imbalances, achieving a BF Score of **81.15%** for the class: flower. DeepLabv3+ initialised with ResNet 50 weights consistently outperforms other networks in all metrics listed in Table 2. Despite the difference in architectural complexity between ResNet 50 and ResNet 18, the difference of performance between both networks are negligible. The difference between learnable parameters of DeepLabv3+ (ResNet50) and Floral-Net is 42.7 million, and considering the pixel class imbalance, Floral-Net was only behind of ResNet50 by **1.76%** in mean accuracy, **4.1%** in mean IoU and **5.48%** in mean BFScore. The confusion matrices in Table 3 reflects the observation on the metrics, DeepLabv3+ with ResNet 50 weights initialised misclassified **1.4%** of flower pixels (ground truth) as background, and **0.4%** of background pixels(ground truth) as flowers. Floral-Net on the other hand; **2.2%** of flower pixels as background, and **3.5%** of background pixels as flowers.

Figure 2 reflects past observations made on the networks, DeepLabv3+ initialised with ResNet 18 and 50, were consistent at producing high quality predictions, scoring high mean IoU scores. The anomalies of all networks can be visualised in Figure 3 to 5. The top row represents the worst predicted image from each network, and the bottom row represents the worst. Performing worse on average, all networks struggled with extreme variations in petal shape and lighting conditions. Scrutinising further, it can be observed that the curved petal with dark lighting conditions in the bottom left of the worst predicted flower image was inaccurately segmented by the networks. This concludes that occlusions and inconsitent lighting variations of petals in the flower images were one the sole reason of inaccurate segmentations.

ResNet-50 intialised network and Floral-Net suffered the worst from this, where it completely disregarded the right petal and blended the left bottom portion of the flower image with the background. The networks performed best when the petals of the flower were consistent and when there is little lighting variation in the petals, but it reflects the observation made by Saha et al., where a slight over-segmentation can be observed for ResNet 18 and ResNet 50 networks. Compared to the DeepLabv3+ networks, Floral-Net handled complex variations of petals worse. There were 9 instances where the mean IoU was ranging between 0.58 and 0.76, which signifies the tradeoff between model complexity and segmentation performance.

5. CONCLUSION

In brief, the DeepLabv3+ network initialised with ResNet50 weights triumph amongst all networks, and a close second was the ResNet18 initialised network. Floral-Net was still a strong contender due to its lightweight architecture compared to DeepLabv3+ networks. In future works, more experiments and augmentations would be done to Floral-Net to push the extent of this lightweight architecture.

6. REFERENCES

[1] S. Saha, N. Sheikh, B. Banerjee and S. Pendurkar, "Self-supervised Deep Learning for Flower Image Segmentation," 2020 14th International Conference on Innovations in Information Technology (IIT), Al Ain, United Arab Emirates, 2020.

[2] Y. Wu, X. Qin, Y. Pan, and C. Yuan, Convolution Neural Network based Transfer Learning for Classification of Flowers. 2018. doi: 10.1109/siprocess.2018.8600536.

[3] O. Ronneberger, P. Fischer, and T. Brox, “U-NET: Convolutional Networks for Biomedical Image Segmentation,” arXiv.org, May 18, 2015. https://arxiv.org/abs/1505.04597

[4] MathWorks, “Semantic segmentation using Deep Learning - MATLAB & Simulink - MathWorks United Kingdom.” https://uk.mathworks.com/help/vision/ug/semantic-segmentation-using-deep-learning.html

[5] “Figure 2. Data division, using a 70:10:20 split into training, “ResearchGate. https://www.researchgate.net/figure/Data-division-using-a-701020-split-into-training-validation-V-and-testing-test_fig16_360897069

[6] “Create DeepLab v3+ convolutional neural network for semantic image segmentation -MATLAB deeplabv3plus- Math Works United Kingdom.” https://uk.mathworks.com/help /vision/ref/deeplabv3plus.html

[7] X.-Y. Zhou and G.-Z. Yang, “Normalization in training U-NeT for 2D Biomedical semantic segmentation,” arXiv.org, Sep. 11, 2018. https://arxiv.org/abs/1809.03783

[8] “Residual U-Net for retinal vessel segmentation,” IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/abstract/document/8803101?casa_token=DlizmKKVm8cAAAAA:InwycMcFhovXLo5WthTJqA_WG3knAEP10-wRorNFpncdpB7LwpD8__1GaRN-XBTBOhFBKlhQBmI

[9] “Count occurrence of pixel or box labels - MATLAB countEachLabel
- MathWorks United Kingdom.” https://uk.mathworks.com/help/vision/ref/pixellabelimagedatastore.counteachlabel.html

[10] “Classification: accuracy,” Google for Developers. https://developers.google.com/machine-learning/crash-course/classification/accuracy

[11] “Evaluate semantic segmentation data set against ground truth - MATLAB evaluateSemanticSegmentation
- MathWorks United Kingdom.” https://uk.mathworks.com/help/vision/ref/evaluatesemanticsegmentation.html