IMAGE SEGMENTATION FOR FLOWER CLASSIFICATION DATASET USING DEEP LEARNING METHODS

Nikhil Raj Ravi Singh

University of Nottingham

ABSTRACT

This project studies the segmentation of image on the oxford flower dataset using the deep learning methods. I implemented two approaches, an advanced U-Net convolutional neural network and a custom convolutional neural network built from scratch. Our objective was to effectively separate flowers from their backgrounds while handling the common issues such as noise, variability and overfitting. In data pre-processing steps, I employed a custom read function along with Gaussian filter to reduce the noise in the data. Additionally, I used data augmentation techniques to overcome overfitting problems. Using the convolutional neural network models, I segmented the input images into two categories flower segment and the background segment. The background segment includes boundaries, sky and leaves within the input data. This project is evaluated using the confusion matrix and Intersection over Union (IoU) metric which demonstrates the segmentation accuracy of the flower on the input images. Furthermore, the developed models exhibit robustness across diverse imaging conditions, including agricultural and medical images.

Index Terms— Image segmentation, Deep learning methods, Oxford Flower dataset, U-Net, IoU.

1. INTRODUCTION

Image segmentation is a fundamental aspect of computer vision, it involves dividing an image into distinct segments and changing the representation of an image to learn the useful features from the image. This domain is crucial for many applications ranging from medical imaging to agricultural monitoring. The primary focus in this project is to accurate segmentation of flower from different background situations. Segmentation of specific objects like flowers from the background will face to many challenges like the complexity of scenes, variations in colour, texture and noise in the images. The Traditional methods of image segmentation have included thresholding techniques, edge-based methods, and region-based methods. These methods did not perform precisely with different lightning conditions, textures and complex backgrounds. The goal in this project is to build two deep learning models which segments the flower. One of the model should be pre-trained convolutional neural network and another one should be a neural network model built from scratch. I need to improve the model's performance across different background scenarios and it is evaluated using metrics such as accuracy, confusion metrics and Intersection over Union (IOU). These metrics will help us understand how well the models can correctly identify pixels and outline the boundaries of flowers. Recent advancements in deep learning especially convolutional neural network (CNN) made a huge leap in the approach of image segmentation [10]. These CNN model can learn features from the images which leads to the robust segmentation in various conditions, therefore leading to the best precision. For instance, Long et al. introduced the Fully Convolutional Network (FCN), a critical development in using deep learning for pixel-wise classification, which has been foundational for subsequent innovations in segmentation networks [1]. However, there exists some issues in the application of these models like U-Net, FCN. They lack in segmenting the objects for highly variable images especially in the botanical studies. While models with large parameters and deep architecture provide capabilities to handle complex scenarios but there is still some modifications required to train the flower segmentation in natural environments. This project will address these issue by using the existing deep learning approaches and modifying them to precisely segment the flower from the background to help in botanical studies and agricultural research.

In our project, the input data comprises of a collection of high resolution images from the Oxford Flower Dataset, which is specifically designed for image segmentation tasks. This dataset consists of 17 categories flower images with 80 images in each class, in total the input data has 1360 flower images each with corresponding label images. The images are categorized under various flower types, each represented with light variations, overlapping objects, unique texture, colour, and structural characteristics [4] [5]. The complexity of the dataset is similar to real word conditions which helps the model to perform well in segmenting the flower across different situations. The provided dataset has two types of files, they are images and labels. The image file consists of 1360 RGB flower images in JPEG format, each image is resized to 256x256 pixels.

The labels file consists of segmentation ground truth labels of the images as colour maps. It has only 846 images in PNG format, which indicates there are some images which do not have segmentation labels. These label maps are formatted with different colour maps where each colours represents different classes such as flowers (1), leaves (2), background (3) and sky (4). In our project the labels are simplified to focus on two classes, those are flower class (1) and background class (3). So, we assumed flower as one class and rest all as one class. This simplification was done to improve the model's performance to segment the flower regions from the background effectively.

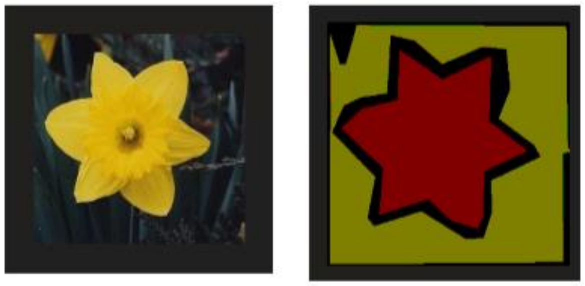

Figure 1.1 Input and Corresponding Segmentation Mask for Image Segmentation

The figure 1.1 presents the example of input images utilized in the project. The image on the left is taken from the image file which shows a flower in its natural environment. The image on the right is from the label file which represents a segmentation mask created for the corresponding flower image. It highlights the flower in a different colour compared to the background colour which helps identify the flower separately from its background. Our aim is to create a model that automatically generate such segmentation masks for given flower images with varying backgrounds.

Data pre-processing is a crucial step in our project preparing the meaningful image data for successful training and to improve the overall performance of the deep learning model. I have reviewed the input data, the label file had images without the corresponding segmentation labels. Therefore I created a custom function to go through the input file and separate out images without segmentation labels. This function ensured that the model training phase utilized only those images that had corresponding labels, thus improving the training performance of the model.The input images had some noises which will affect the accuracy of the model. To handle this issue I tried two filter methods. Gaussian filter and Median filter. Median filter is a non-filtering technique used to remove the impulsive noise. It replaces the noisy pixels with the median value of neighbouring pixels making it robust to outliers and preserving image details [8]. On the other hand, Gaussian filter works by calculating weighted average for each pixel based on the values of neighbouring pixel [7]. It works well to reduce high frequency noise in the image. It blurs the image based on the standard deviation value and it is more computationally efficient then the median filter. I opted to use the Gaussian filter because it works better in smoothing the images while preserving edge details which is very important as it helps in accurately segmentation of the flower. This improves the overall quality of the image without changing the features that define the shape of the flower. Once the unannotated images and the noises are removed from the input data, I have left with only 846 images to train the model which is very less. To enhance the dataset size and also improve the robustness of the model I utilized data augmentation techniques. This technique performs random rotations, flips and shifts on the input images. Such transformations improve the diversity of training dataset without the need for acquiring additional data [6]. This step is crucial for improving the models performance and its capability to accurately segment flowers across different natural scenarios that are not captured in the dataset.

The goal of the project is to build two convolutional neural network models, a pre-trained model and a custom model that segments the flower from varying complex backgrounds. At first let's look into the custom CNN model. In the construction of my custom CNN network, a structured and layered approach was used to build the network. The network consists of three section, downsampling section, upsampling section and fully convolutional layer section. The downsampling section of the network is crucial for feature extraction, where the size of the input is reduced while the depth is increased to capture essential features. This section of the network has several key layers such as Convolutional2DLayer, Batch Normalization Layer, ReLU Layer and Max Pooling Layer [9]. The Convolutional2DLayer will take a part of the input image and apply many different filters on that part and will process these images to give output. The filter size I used is 3x3 pixels and it creates a feature maps that summarizes the important features of the input images. Max pooling function selects the maximum pixel value from the region of the feature map covered by the filter. It is performed on the convolutional layer extracting the most important feature from the input. The Batch Normalization Layer normalizes the input by transforming the input values such that their mean is zero and standard deviation is one. ReLU is an activation function in the network, this decides whether a neuron should be activated or not by calculating the weighted sum and adding bias to it. It is defined by maximum of zero and input that is the neuron will be activated only if it has positive value [12]. The upsampling section consists of ReLU, Batch Normalization Layer and TransposedConv2DLayer. Batch Normalization Layer and ReLU layers are same as in downsampling section. The TransposedConv2DLayer perform the reverse of convolutional operations, this enlarges the feature maps and recovering spatial dimensions lost during downsampling. The fully connected layer section indicates the final stages of the architecture. This section consists of Convolutional2DLayer, Softmax Layer and PixelClassification Layer. The Softmax layer is another activation function used only in the fully connected layer, it transforms the raw outputs of the neural network into a vector of probabilities. This layer calculates the probability distribution of each class at every pixel position which is crucial for classifying each pixel into specific categories. The PixelClassificationLayer interprets the softmax probabilities by assigning each pixel to the most likely class. Therefore producing the final segmented image using those probabilities. Finally all the sections are stacked on one another ensuring a seamless flow from input to output. The second approach is to build a model which utilises pre-trained CNN models. There are many models available in the field of convolutional neural network, I chose two networks U-Net and VGG 16 network. U-Net network is one of the convolutional neural network that works well in segmentation of images due to its symmetric structure and the incorporation of skip connections which link the downsampling path directly to the upsampling path [2]. These connections transfer feature information from early layers of the network to later layers, combining high-resolution features from the downsampling path with upsampled outputs. The downsampling layer consists of repeated sequence of convolutional layer each followed by a ReLU layer and a max pooling operation. In the upsampling section the feature map is enhanced by a convolution that halves the number of feature channels and is then concatenated with the correspondingly cropped feature map from the downsampling path, followed by two more convolutions. VGG16 is another convolutional neural network which is famous for its deep architecture and robustness. It is characterized by its depth and the use of 3x3 convolutional layers stacked on top of each other in increasing depth [3]. This network consists of input layer which accepts only image size of 224x224 pixels, convolutional layers, activation layers, pooling layers and fully connected layers. The total number of parameters in VGG16 is approximately 138 million. Its architecture is straight forward and it is effective in capturing critical features which makes it suitable for wide range of image segmentation tasks. Initially, I chose VGG16 model for this project of flower segmentation due to its robustness and its efficiency in image recognition tasks. However, I found it unsuitable for my project needs. One of the primary issue in VGG16 is its large number of parameters which require heavy computational resource to train those parameters. And also VGG16 model requires the input size resized to 224x224 pixels, despite changing the size. I was not able to fully adapt the VGG16 architecture to achieve the desired output proved to be complex. So after careful consideration, I chose U-Net network architecture which works highly efficient in learning from a limited amount of data and also it offers a balance between performance and resources efficiently.

2. METHODOLOGY

In previous sections, I have discussed about the different data pre-processing methods and CNN networks I explored and explained why I selected specific methods for this project. In this section, let's look into how I implement those chosen methods. Initially, I developed a custom function to remove the unannotated images within the image file. This function uses a for-loop that iterates through the image file and checks whether each flower image has a corresponding segmentation map in the label file by comparing the names in both files. Images with matching segmentation maps are then stored in a separate folder. I will use only these images in the further process. To remove the noise in the images, I utilized Gaussian filter to smooth out the noise. To implement this I used a custom read function that reads the image from the directory and applies the Gaussian filter with standard deviation of 2. Then I set up the image datastore for flower images and pixel label datastore for label images in the dataset. I set the pixel label id as 1 for flower and 3 for background. After this I performed data augmentation which included random rotation where the images are randomly rotated within a range of -2 to 2 degrees, random scale to randomly scale the images by a factor between 0.9 and 1.1 and finally random translation with 10 pixels that shift the images both horizontally and vertically. Using the pixelLabelImageDatastore, I configured it with the original image and pixel label datastores by applying all the above parameter of data augmentation.

Now that the data is prepared to use in the model let’s build the CNN model, first approach is through custom CNN model and second by using, a pre-trained CNN model. At first I defined the input layer of 256x256 pixels matching our input size. Then I defined the downsampling layer section of the model which consists of three blocks of sequential layers. In the first block I defined a convolutional layer with 3x3 filter size, 64 number of filters and padding as same, followed by a batch normalization layer and a ReLU layer. Repeat this sequence once and then add a max pooling layer at the end of first block. In the second block I repeated the same sequence but increased the number of filters in the convolutional layers from 128 filters. Similarly in the third block, I further increased the number of the filter to 256 filters. After the construction of down sampling layers, I build the upsampling layers section using transposed convolutional layer, batch normalization layer and ReLU layer. This section begins with the transposed convolutional layer with 4x4 filter, 256 number of filters, a stride of 2 and cropping of 1. This is followed by similar layers with decreased number of filters 128 and another layer with 64 filters. Each transposed convolutional layer is paired with a batch normalization layer and a ReLU layer. Finally, I constructed the fully connected layer with convolutional layer of 1 filter size and number of out classes which is 2, followed by a softmax layer and pixel classification layer [11]. The model layers, including the input layer, downsampling layer, upsampling layer, and fully connected layer, are then stacked to complete the construction of my custom CNN model. To train my model, I utilized trainingOptions function with Adam optimizer [13], a learning rate with 0.001, max epochs to 10, and mini batch size to 128. After successful training I saved my model in my directory. For the second approach, I used a pre-trained U-Net network to segment the flowers utilizing the same dataset as in the first approach. This model was built by importing the U-Net architecture by setting the input size to 256x256 pixels and number of output classes to 2. I trained the model using the trainingOptions function with the Adam optimizer, a learning rate of 0.001, a maximum of 10 epochs, and a mini-batch size of 64, and then saved it in my directory. To evaluate my model, I used standard accuracy metrics, confusion matrix and the Intersection over Union (IoU) metric, using the confusionchart and metrics functions available in MATLAB. The evaluation results will be discussed in detail in the results section.

3. RESULT

The evaluation of the model is the crucial step in this project, it serves as the proof that my proposed model works precisely in the real time natural images. The evaluation metrics I considered include standard metrics such as accuracy, confusion matrix and Intersection over Union [15]. Global Accuracy measures the overall accuracy of the model across all classes and it is calculated as the total number of correct predictions divided by the total number of predictions made. Mean Accuracy is the average accuracy obtained on each class individually. The confusion matrix will provide detailed performance of the models for each class. Intersection over Union (IoU) measures the overlap between predicted segmentation and ground truth. A higher IoU indicates better segmentation accuracy [14]. Mean IoU calculates the average IoU across all classes in the dataset. F1 score is a measure of the harmonic mean of precision and recall [16].

Models	Global Accuracy	Mean Accuracy	Mean IoU	Mean Boundary F1 Score
Pre-trained Model	0.8986	0.8774	0.7918	0.5539
Custom Model	0.9567	0.9535	0.9062	0.8386

Table 1.1 Results of CNN Models.

The comparison between the pre-trained and custom CNN models shows that each model has its own strengths and weaknesses when it comes to segmenting flower images. The pre-trained model is quite good overall, achieving a Global Accuracy of 89.86% and a Mean IoU of 79.18%. This makes it reliable for general tasks where perfect detail is not crucial. However, it struggles with accurately outlining the exact edges of objects, as shown by its lower Mean Boundary F1 Score of 55.39%. In contrast, the custom model performs exceptionally well with a very high accuracy of 95.67% and excellent detail in edge definition, shown by an F1 Score of 83.86%. This proves the custom model is better suited for this tasks where capturing precise details is important.

4. CONCLUSIONS

In conclusion, I tested whether specific data pre-processing techniques and the utilization of both custom and pre-trained CNN models could enhance the accuracy and efficiency of flower segmentation from complex backgrounds. The results I obtained shows that our methods significantly enhanced the accuracy of our image segmentation. The use of a custom CNN model optimized for our dataset, coupled with the pre-trained U-Net model network, provides the robust solutions adaptable to varying imaging conditions. The key takeaways from this study are the effectiveness of utilizing multiple convolutional network architectures and pre-processing techniques to handle complex segmentation tasks and the critical role of data augmentation in enhancing model generalization. Additionally, I also learned to balance the computational efficiency with model is crucial. For future research, it would be interesting to explore the combining the strengths of both custom and pre-trained models together, possibly by using newer and more efficient network structures. Future efforts could focus on utilizing the images that were initially set aside due to the lack of corresponding segmentation labels. Integrating these left-out images could expand the dataset significantly, potentially leading to even more robust models capable and further improving the accuracy and efficiency of the segmentation process.

5. REFERENCES

[1] Shelhamer, Evan & Long, Jonathon & Darrell, Trevor. (2016). Fully Convolutional Networks for Semantic Segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence. 39. 1-1. 10.1109/TPAMI.2016.2572683.

[2] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in Medical Image Computing and Computer-Assisted Intervention (MICCAI), Springer, Cham, 2015, pp. 234-241.

[3] Simonyan, Karen & Zisserman, Andrew. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv 1409.1556.

[4] M. . -E. Nilsback and A. Zisserman, "A Visual Vocabulary for Flower Classification," 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), New York, NY, USA, 2006, pp. 1447-1454, doi: 10.1109/CVPR.2006.42.

[5] Nilsback, Maria-Elena & Zisserman, Andrew. (2007). Delving into the Whorl of Flower Segmentation.. BMVC. 1. 10.5244/C.21.54.

[6] Perez, Luis & Wang, Jason. (2017). The Effectiveness of Data Augmentation in Image Classification using Deep Learning.

[7] J. S. Lim, "Two-Dimensional Signal and Image Processing," Prentice Hall, 1990, pp. 479-480.

[8] D. Zhang and Z. Wang, "Image information restoration based on long-tail distribution noise modeling," in Journal of Visual Communication and Image Representation, vol. 21, no. 3, pp. 263-271, April 2010.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), 2016, pp. 770-778.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in Communications of the ACM, vol. 60, no. 6, pp. 84-90, 2017.

[11] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-Based Learning Applied to Document Recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, November 1998.

[12] V. Nair and G. E. Hinton, "Rectified Linear Units Improve Restricted Boltzmann Machines," in Proceedings of the 27th International Conference on Machine Learning (ICML-10), 2010, pp. 807-814.

[13] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in 3rd International Conference for Learning Representations, San Diego, 2015.

[14] J. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal Visual Object Classes (VOC) Challenge,

[15] K. Kohavi and F. Provost, "Glossary of Terms," Machine Learning, vol. 30, no. 2-3, pp. 271-274, 1998.

[16] C. D. Manning, P. Raghavan, and H. Schütze, Introduction to Information Retrieval, Cambridge University Press, 2008, pp. 260-264