WEAKLY SUPERVISED IMAGE SEGMENTATION WITH MULTIPLE INSTANCE LEARNING NEURAL NETWORKS

By

GUOHAO YU

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2019

TABLE OF CONTENTS

page

LIST OF TABLES . . . 5

LIST OF FIGURES . . . 6

LIST OF SYMBOLS . . . 7

1 INTRODUCTION . . . 8

1.1 Weak Label in Multiple Instance Learning . . . 9
1.2 Minirhizotron Image Dataset . . . 10
1.3 Remote Sensing Dataset . . . 11
1.4 Visual Object Classes Challenge 2012 dataset . . . 12
1.5 Motivation . . . 12

CHAPTER

2 LITERATURE REVIEW . . . 15

2.1 Semantic Segmentation . . . 15
2.1.1 Supervised Semantic Segmentation . . . 15
2.1.2 Weakly Supervised Image Annotation . . . 16
2.1.3 Weakly Supervised Image Segmentation . . . 18
2.1.3.1 Graph Cuts Based Approach . . . 18
2.1.3.2 Conditional Random Field Based Approach . . . 20
2.1.3.3 Convolution Neural Network Based Approach . . . 23
2.2 MR Image . . . 30
2.2.1 Minirhizotron Root Image Segmentation . . . 30
2.2.1.1 Unsupervised Root Segmentation . . . 31
2.2.1.2 Supervised Root Segmentation . . . 31
2.2.1.3 Weakly Supervised Root Segmentation . . . 35

3 ALGORITHM . . . 41

3.1 Multiple Instance Learning Neural Network . . . 41
3.1.1 Multiple Instance Learning Neural Network with FCNN Output . . . 42
3.1.2 Multiple Instance Learning Neural Network with MI-ACE . . . 43
3.1.3 Multiple Instance Learning Neural Network with Attention Map . . . 44
3.2 Optimization . . . 46
3.2.1 General Backpropagation in Neural Network . . . 46
3.2.2 MIL Layer Backward Derivation . . . 48

4 EXPERIMENTS AND RESULTS . . . 50

4.1 Data Description . . . 50
4.2 Multiple Instance Neural Network with FCNN Output . . . 51

4.2.1 Random Initialization . . . 51
4.2.2 Transfer Learning . . . 52
4.3 Multiple Instance Neural Network with attention map . . . 54
4.3.1 Attention Map . . . 56
4.3.2 Attention Map with CRF . . . 56
4.4 Multiple Instance Neural Network with MI-ACE . . . 58
4.4.1 MIACE Scoremap as with Top n . . . 58
4.4.2 MI-ACE Scoremap as Initial Pixel Scoremap with Top n . . . 58
4.4.3 MI-ACE Scoremap with Threshold . . . 61
4.5 Compare Unet, MIL-Unet, and MI-ACE . . . 62

5 FUTURE WORK . . . 65

5.1 Experiments on how to indicate where the targets are in the positive image . . . 65
5.1.1 Experiment always using MI-ACE scoremap to indicate where the targets are in the positive image . . . 65
5.1.2 Experiment using MI-ACE scoremap to indicate where the targets are in the positive image at the initial step of training MIL-NN . . . 65
5.1.3 Experiment using the pre-trained model to compute attention map to indicate where the targets are in the positive image . . . 66
5.1.4 Experiment using the learnable attention map to indicate where the targets are in the positive image . . . 66
5.2 Experiments on how to select targets completely and accurately . . . 67
5.2.1 Experiment on picking a fixed number of pixels as targets . . . 67
5.2.2 Experiment on setting threshold to pick targets . . . 67
5.3 Experiments on adding constraints to features of targets and non-targets . . . 67
5.3.1 Experiment on adding constraint to features based on bag label . . . 67
5.3.2 Experiment on adding constraint to features based on spatial and low level features . . . 68

REFERENCES . . . 69

LIST OF TABLES

Table page

2-1 Weights used in Graph Cuts (Boykov and Jolly, 2001) 19

LIST OF FIGURES

Figure page

1-1 Example of a switchgrass minirhizotron image with full annotation. 9

1-2 Example of a switchgrass minirhizotron image with MIL annotation. 10

1-3 The illustration of a minirhizotron imaging system. 11

1-4 Example of NEON dataset. 12

1-5 Example of PASCAL VOC 2012 dataset. 13

2-2 The illustration of CRF 21

2-3 The illustration of neural network structure for CAM 25

2-4 The illustration of trainable attention models 27

2-5 The illustration of trainable attention models 28

3-1 The illustration of multiple instance learning neural network with FCNN 42

3-2 The illustration of multiple instance learning neural network with MI-ACE 44

3-3 The illustration of multiple instance learning neural network with attention map . . 45

3-4 The illustration of neural network 47

3-5 The illustration of MIL layer 48

4-1 Random initialization. pick top 10 52

4-2 Transfer learning. pick top 10 53

4-3 Attention map, pick top 10 55

4-4 Threshold attention map, refine threshold attention map with CRF 57

4-5 Use MIACE as segmentation mask, pick top 10 59

4-6 Initial use MIACE as segmentation mask, pick top 10, then pick top 10 from target class prediction 60

4-7 Use threshold MIACE as segmentation mask 61

4-8 Compare results of Unet, MIL-Unet, and MI-ACE. 62

4-9 initial use MIACE as segmentation mask, pick top 10, then pick top 10 from target class prediction 64

LIST OF SYMBOLS, NOMENCLATURE, OR ABBREVIATIONS

COCO	Common Objects in Context
CNN	convolution neural network
FCNN	fully convolutional neural network
FCN	fully convolutional network
CRF	conditional random field
mIoU	mean Intersection-over-Union
MAP	maximum a posteriori
HOG	histogram of oriented gradients
CAMs	class activation maps
GAP	global average pooling layer
fc	fully connected layer
Deconvnet	deconvolutional network
LET	local entropy thresholding
SLIC	Simple Linear Iterative Clustering
MIL	multiple instance learning
MI-ACE	multiple instance adaptive cosine/coherence estimator
miSVM	multiple instance support vector machine
MIForests	multiple instance random forest
SVM	support vector machine
DA	Deterministic annealing
ACE	Adaptive Cosine Estimator
MIL-NN	multiple instance neural network
MIL-Layer	multiple instance layer
SGD	stochastic gradient descent
NEON	National Ecological Observatory Network

CHAPTER 1
INTRODUCTION

Deep learning has achieved state of art results in image segmentation (Long et al., 2015; Ronneberger et al., 2015; Badrinarayanan et al., 2017). An advantage of these deep learning approaches is the ability to learn features from data automatically for the image segmentation task. However, supervised image segmentation with neural networks requires a large number of training data with fully annotated ground truth. It is often very time consuming and tedious to make fully annotated ground truth. Weakly supervised image segmentation provides an alternative way to train neural network with weak annotation (Bell et al., 2015; Bearman et al., 2016; Li et al., 2018; Boykov and Jolly, 2001; Rother et al., 2004; Li et al., 2004; Batra et al., 2010; Lin et al., 2016; Zhang et al., 2018a; Dai et al., 2015; Papandreou et al., 2015). Instead of giving each pixel a label, spots (Bell et al., 2015; Bearman et al., 2016; Li et al., 2018), scribbles (Boykov and Jolly, 2001; Rother et al., 2004; Li et al., 2004; Batra et al., 2010; Lin et al., 2016; Zhang et al., 2018a), bounding boxes (Dai et al., 2015; Papandreou et al., 2015), and image-level labels (Papandreou et al., 2015) are used as weak or imprecise labels. Although weakly supervised image segmentation approach relieves the tedium of making groundtruth, the current results of weakly supervised image segmentation methods are not as good as the results of supervised image segmentation methods.

Among many ways to assign weak labels, we study image with image-level label following the multiple instance learning framework to do image segmentation. One of our applications is segment minirhizotron images. According to (Lin et al., 2014), it takes $239.7sec/img$ to make full supervision annotation for images in Common Objects in Contest (COCO) dataset. However, when we making labels for our switchgrass dataset, it takes us several hours to label all roots in each image. Hence, it is appealing to reduce the effort in labeling minirhizotron image. Fig.1-1 shows an example of a switchgrass minirhizotron image. The roots are labeled by marking in red color. Besides time, label inaccuracy is also a huge problem. Even a careful annotator could not achieve a high accuracy due to the low contrast of roots to soil, the large

number of tiny roots and so on. Multiple instance learning approach is more suitable than supervised learning approach at dealing with inaccurate label, hence it is necessary to handle image segmentation with weak label as a multiple instance learning problem.

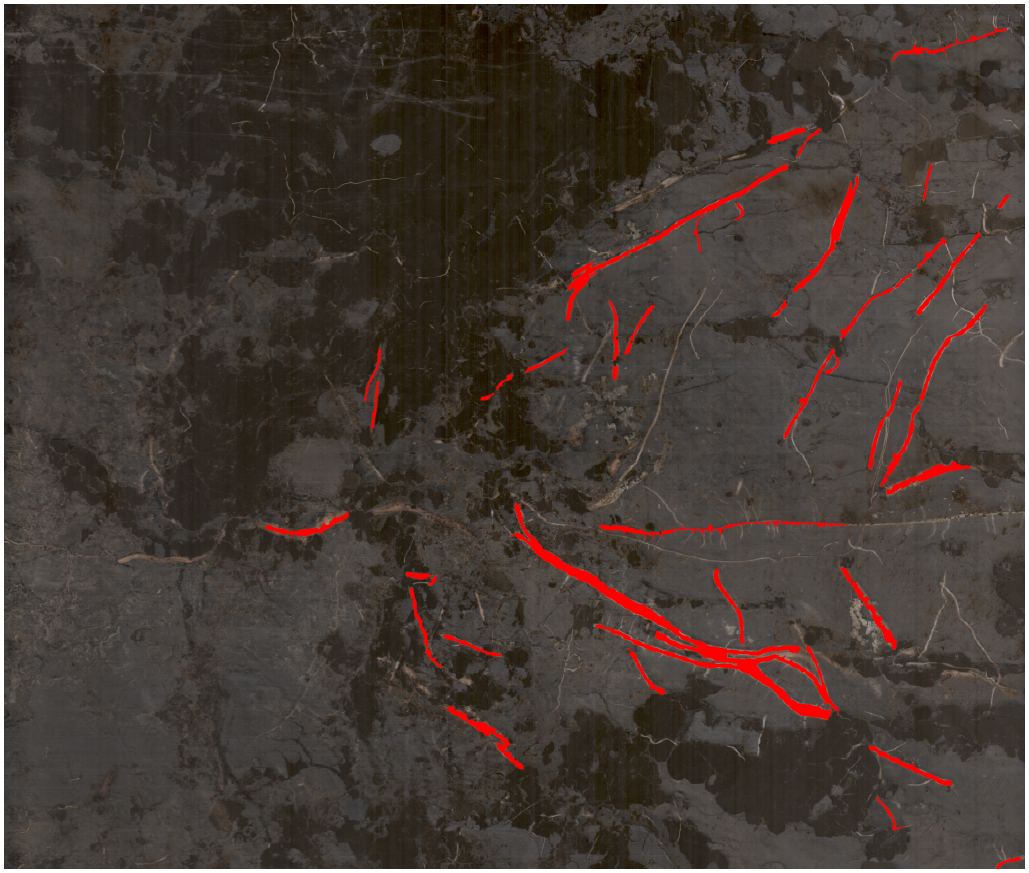
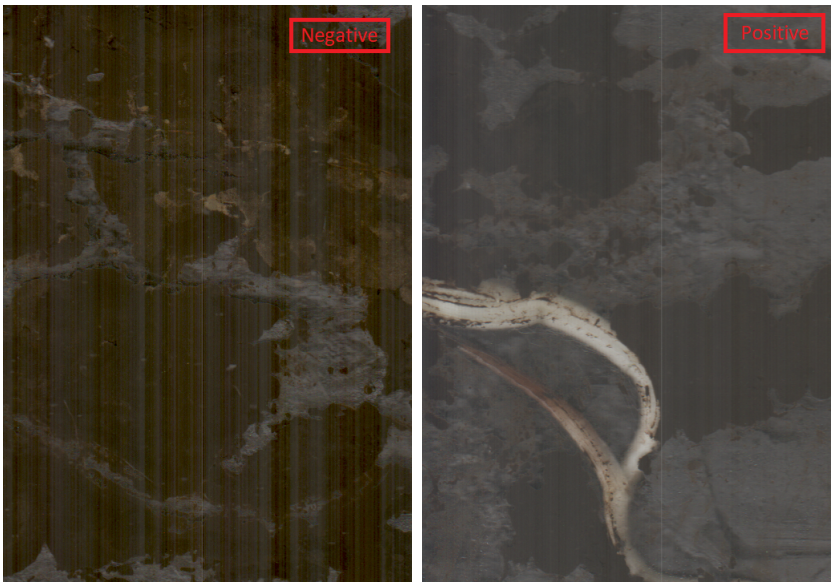


Figure 1-1. Example of a switchgrass minirhizotron image with full annotation.

1.1 Weak Label in Multiple Instance Learning

The training dataset is organized by bags in multiple instance learning. Each bag contains multiple instances. The MIL method deals with training data having bag label instead of instance label. Hence, the MIL method is an algorithm deal with weak label. For a image segmentation problem, each image is a bag. The image-level label is the bag label. Each pixel is an instance in bag. An image having no pixels belong to the object class that we try to find is a negative bag. An image having at least one pixel belongs to the object class that we try to find is a positive bag. For the root segmentation example, an image having roots is a

positive image and roots are targets. An image having no roots is a negative bag as illustrated in Fig.1-2. The MIL label is a stronger weak label than image label (Papandreou et al., 2015). The MIL label provides more supervision. Not only the label of negative bags is known, but also the labels of the instances in the negative bag are known since the instances in negative bags are all from background class.



A negative bag B positive bag

Figure 1-2. Example of a switchgrass minirhizotron image with MIL annotation.

1.2 Minirhizotron Image Dataset

Minirhizotrons provide an efficient and non-destructive way to collect plant roots for studying root system dynamically. Common sampling methods used to examine root system dynamics are destructive and provide limited ability to draw inferences on the plant response to stresses experienced during the growing season. For example, soil coring, excavation, trenches, or ingrowth cores extract a unique dataset in time that destroys the integrity of the root system and thus limits repeat measurements from the same plant without the introduction of confounding effects (Taylor et al., 1991). In contrast to these destructive methods, once inserted into the soil, transparent minirhizotron access tubes allow non-destructive assessment of roots over extended periods of time without repeatedly altering critical soil conditions

or root processes (Johnson et al., 2001; Bates, 1937; Waddington, 1971). Roots that grow adjacent to the tube are imaged by inserting a camera (with a light source) into the tube and acquiring images along the length of the tube, providing information about roots present at different soil depths as illustrated in Fig.1-3. Since minirhizotron tubes allow for the collection of root images over time, this enables monitoring of complex root growth and turnover dynamics (Johnson et al., 2001). The fast and large amount of minirhizotron images in contrary to the time consuming and tedious since standard analysis approaches involve manually outlining and labeling roots using commercially available software such as WinRhizo Tron (Regent Instruments, Canada) or RootSnap (CID BioScience, Camas, WA, USA) advocate the neccessary to develop automatic method to trace roots in minirhizotron images.

A minirhizotron imaging system in field

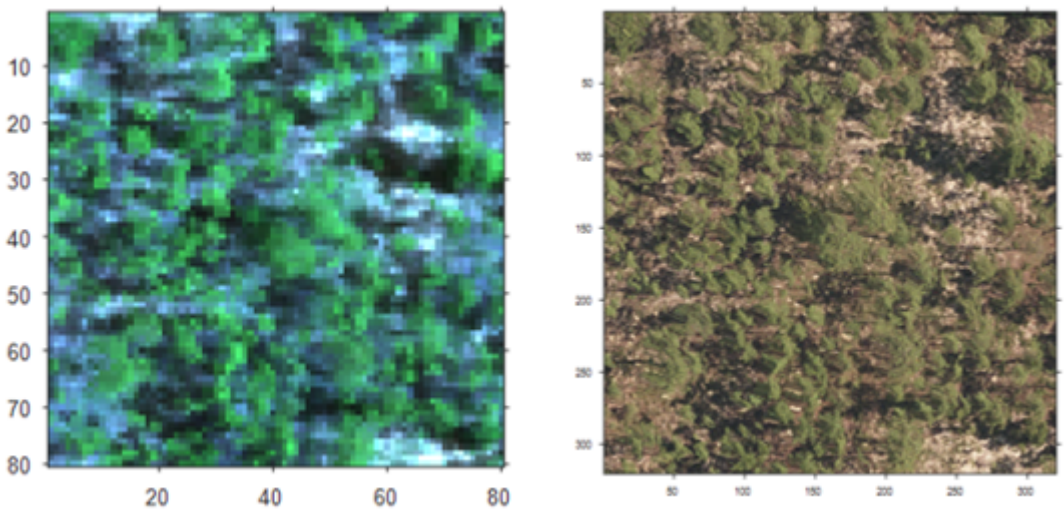
B minirhizotron imaging system illustration

Figure 1-3. The illustration of a minirhizotron imaging system.

1.3 Remote Sensing Dataset

The National Ecological Observatory Network (NEON) field dataset used for the 2017 Data-Science competition is a remote sensing dataset that contains hyperspectral data, light detection and ranging (LiDAR) data, and Red-Green-Blue (RGB) data for a subset of plots at Ordway Swisher for tree crown segmentation (group, 2017). The tree crown segmentation helps understanding forest structure and an important first step in species classification. The resolution of the hyperspectral data is $1.0m \times 1.0m$. The RGB data have a finer resolution

of $0.25m \times 0.25m$ which provide more details in the image than the hyperspectral data. However, the hyperspectral data provides a spectral signatures to differentiate tree species. The LiDAR data provides information on the spatial variation in canopy height. Three bands of the hyperspectral data are taken and illustrated together with the RGB data in Fig.1-4.



A hyperspectral data B RGB data

Figure 1-4. Example of NEON dataset.

1.4 Visual Object Classes Challenge 2012 dataset

The Pattern analysis statistical modelling and computational learning (PASCAL) visual object classes challenge (VOC) 2012 dataset containts RGB images of 20 visual object classes in realistic scenes (Everingham et al.). The PASCAL VOC 2012 dataset has been accepted as the benchmark for image segmentation. Fig.1-5 shows an example of image and the groundtuth of that image from PASCAL VOC 2012 dataset.

1.5 Motivation

The difficulty of getting a large number of images with pixel-level annotation, the performance gap between weakly supervised image segmentation and supervised image segmentation, and the recent success of neural network in image segmentation motivate us to study weakly supervised image segmentation with image-level label under multiple

A Image B GroundTruth

Figure 1-5. Example of PASCAL VOC 2012 dataset.

instance learning constraint. The ability to picking targets from positive image accurately with image-level label is crucial to the segmentation results.

Can we indicate where the targets are and selected them completely in the positive image? In our research, we plan to use multiple ways to select targets from positive bags. We will use the MI-ACE scoremap (Yu et al., 2019), attention map (Zhou et al., 2016), and the output of softmax layer of neural network as guidance to targets in positive bag. The attention map is computed from a network used for image classification. The attention map can indicate the important parts in image to the object class. The MI-ACE scoremap takes the advantage of a pre-trained model using low level features to classified data. The MI-ACE scoremap indicates how similar a test data to the target signature. The output of neural network from softmax layer indicate the possibility of a data belongs to the object class. We will compare attention map, miace scoremap and the output of neural network to find a best way for pixel class scoremap. The multiple instance learning constraint indicates that there are targets in positive image. The more close the targets picked from positive images to the true targets in

images, the more close the performance of weakly supervised image segmentation to supervised image segmentation. The different ways to pick targets in positive image we planed to use include: picking a fixed number of targets from MI-ACE scoremap (attention map, neural network softmax layer output); setting a threshold to MI-ACE scoremap (attention map, neural network softmax layer output); taking the arguments of the maxima of attention map (neural network softmax layer output). Setting a threshold and taking the arguments of the maxima allow us to adaptively change the number of picked targets. Picking a fixed number of targets gives us more control to the number of targets being picked which helps to study the effect of increasing the number of picking targets to the segmentation results.

The difficult to pick all targets from weakly labeled positive image results in only a subset of targets with some amount of non-targets are selected as targets used for training. Can we make the model more robust and trustful by adding constraints to the features learned from the neural network? We will use the bag label, pixels spatial information, low-level features to add constraints to features. Data assigned same label, close in spatial, or similar in low-level features should be more similar in feature space, otherwise, should be more dissimilar in feature space.

CHAPTER 2
LITERATURE REVIEW

In this section, we will give an overview of methods that used for weakly supervised image segmentation in Sec.2.1 and methods used for minirhizotron image segmentation in Sec.2.2

2.1 Semantic Segmentation

Semantic segmentation aims at making a prediction of each pixel in image. Compared with image classification and object detection, semantic segmentation is a quite challenging problem. The recent breakthrough in deep neural network boosts the results on semantic segmentation.

2.1.1 Supervised Semantic Segmentation

The first wildly known work which extended the classification convolution neural network (CNN) for semantic segmentation was conducted by Long *et. al.* (Long et al., 2015). Their fully convolutional network (FCN) removed the fully connected layer from standard CNN, including AlexNet (Krizhevsky et al., 2012), VGG net (Simonyan and Zisserman, 2014), and GoogLeNet (Szegedy et al., 2015), adding a novel skip architecture to upsample the output from a deep convolution layer in CNN. Their adaptation to the CNN achieved a much more accurate spatially dense pixelwise prediction and resulted in a better image segmentation. The skip architecture fused the output of deep layer and shallow. The output of the deep layer had lower spatial resolution but larger feature channel dimension than the shallow layer. Hence a convolution layer was first applied to both the deep layer and the shallow, converting the feature channel dimension to be the same a class number. Then for the output from deep layer, a deconvolution layer that upsample the spatial dimension to be the same as shallow layer are followed. And then the resulting output of deep layer and shallow layer are added together. Each time, two layers were fused togehter. To fuse more layers, the same operations are repeated over layers from deep to shallow.

U-Net (Ronneberger et al., 2015) developed the FCN (Long et al., 2015) to have a large channel dimension on the upsampling part and gradually reduce to the number of class

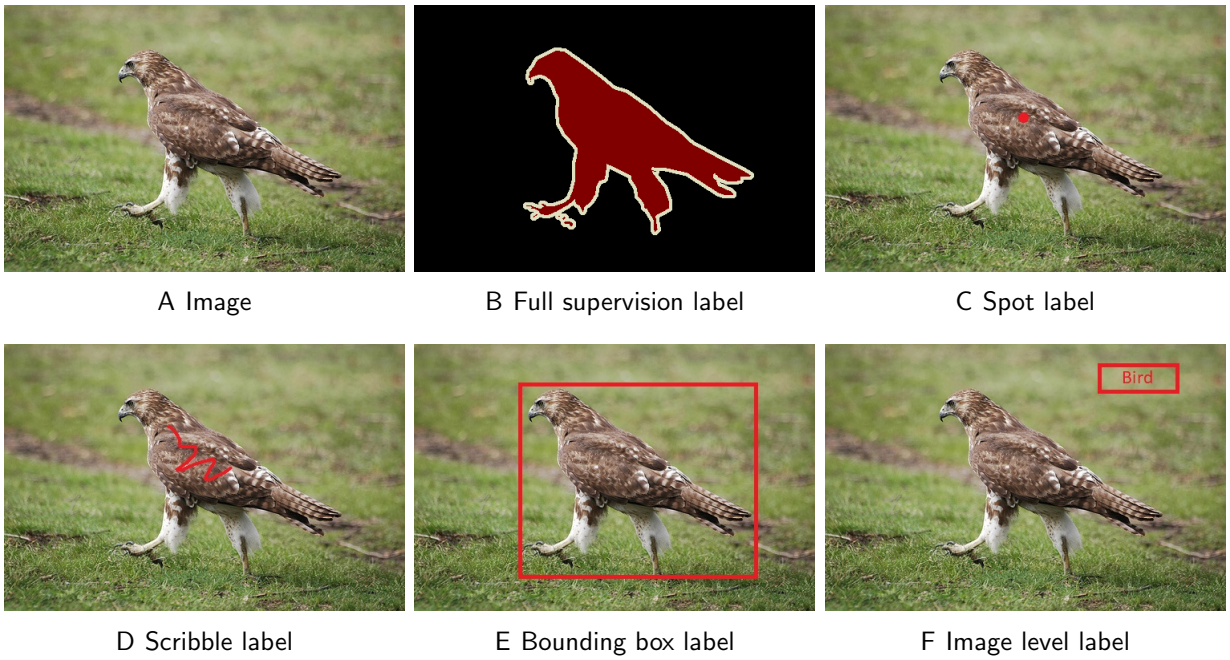
instead of contracting the channel dimension of the upsampling part to the number of class directly in (Long et al., 2015). Gradually reduce the channel dimension make the upsampling part more symmetric to the downsampling part and avoid the channel dimension drop from a large number to a small number too fast. The downsampling path is called contracting path or encoder. The upsampling part is called expanding path or decoder. Keep the channel dimension of the expanding path decreased slowly help keeping the information from the shallow layer. Their experiments showed that U-Net architecture achieves good performance.

2.1.2 Weakly Supervised Image Annotation

Although neural network has outperformed other methods which used hand crafted features in supervised semantic segmentation, it is not so applicable to training a neural network for supervised semantic segmentation from scratch because it requires a large amount of data with pixelwise label. Weakly supervised learning provides an alternative approach to fulfill the semantic segmentation task but only requires coarsely labelled training data. Being able to get rid of labor-intensive and time-consuming pixelwise annotation task makes the weakly supervised semantic segmentation so appealing that extensive valuable researches have been done. Multiple coarse level labels are used for weakly annotation, including scribbles, bounding boxes, spots and image level annotation.

Spot annotation: Spot annotation (Bell et al., 2015; Bearman et al., 2016; Li et al., 2018) is that only one or several points per category in each image are labeled as shown in Fig.2-1C. Making spot annontaion is much easier and faster than making full supervision. According to (Bearman et al., 2016), it takes only $22.1sec/img$ to make point-level label in PASCAL VOC 2012 dataset if only one point is labelled per category. However, it takes $239.7sec/img$ to make full supervision annotation in Fig.2-1B in COCO dataset (Lin et al., 2014).

Scribble annotation: Scribble label (Boykov and Jolly, 2001; Rother et al., 2004; Li et al., 2004; Batra et al., 2010; Lin et al., 2016; Zhang et al., 2018a) is made by drawing scribbles on image and assigning labels to pixels on the scribbles as shown in Fig.2-1D. Scribble



A Image B Full supervision label C Spot label

D Scribble label E Bounding box label F Image level label

Figure 2-1. Weakly supervised image annotation example.

label is easy to made even though scribble label requires all pixels on the same scribble should belong to the same class and at least one scribble per class. Scribble label provide stronger supervison than spot annotation since more pixels are labelled. Graph Cuts (Boykov and Jolly, 2001), the first time a clear cost function was defined for image segmentation problem, used scribble annotated image to interactively segment object from background. Both object and background are labelled by scribble.

Bounding box annotation: Bounding box label (Dai et al., 2015; Papandreou et al., 2015) is another way to provide weakly supervision in Fig.2-1E. For image labeled by boundnig box, a tight box is placed around each individual instance. Unlike spot annotation and scribble annotation, bounding box annotation is not an accurate label for semantic segmentation because not all pixel inside the bounding box are pixels of the instance inside the bounding box. Bounding box label also provide the supervision that pixels in the region outside of the bounding box belong to different class than the instance inside the bounding box.

Image level annotation: The most coarsely label used for semantic segmentation is image level annotation (Papandreou et al., 2015). Annotator label a image as the object that

appeared in the image. For example, in Fig.2-1F, the image label is bird which is the same as the object appeared in the image.

2.1.3 Weakly Supervised Image Segmentation

There are three approaches used for weakly supervised image segmentation: graph cuts based approach; conditional random field based approach, and neural network based approach.

2.1.3.1 Graph Cuts Based Approach

Graph based energy minimization problem via graph cuts are used for weakly supervised image segmentation, especially widely used in interactively dividing a image into foreground and background with scribble labels. A weighted graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$,$\mathcal{V}$ is a set of nodes and $\mathcal{E}$ is a set of edges, is built from image. $\mathcal{V}$ contains two kind of nodes. One is nodes corresponding to pixels in image. The other is nodes corresponding to the set of labels that can be assigned to pixels which are called terminals. A graph $\mathcal{G}$ is made by nodes $\mathcal{V}$ connected with edges $\mathcal{E}$. Each edge has a weight. The edges connected two neighboring pixels are called N-links with weights in the form of $V_{p,q}(L_p, L_q)$. The edges connected a pixels node with a terminal node is called T-links with weights in the form of $D_p(L_p)$. A cut is removing a set of edges $\mathcal{C} \subset \mathcal{E}$ from the graph $\mathcal{G}$ such that each terminal is separated. The graph cuts method find a cut has the minimum cost of cut $\mid C \mid$ 2-1.

$$\mid C \mid = min \sum_{i \in \mathcal{C}} w_i \tag{2-1}$$

and

$$w_i = \begin{cases} D_p(L_p) & \text{if } w_i \text{ is weight of T-links of node p} \\ V_{p,q}(L_p, L_q) & \text{if } w_i \text{ is weight of N-links of node p and node q} \end{cases}$$

Graph Cuts (Boykov and Jolly, 2001; Boykov and Kolmogorov, 2004) was the first work that provided a clear cost function for foreground and background separation. Let $\mathcal{P}$ be the set of nodes p corresponding to pixels, S be the terminal node corresponding to the foreground. T

be the terminal node corresponding to the background. Then

$$\mathcal{V} = \mathcal{P} \cup \{S, T\} \tag{2-2}$$

The nodes of pixels labeled as foreground are in set $\mathcal{O}$, The nodes of pixels labeled as background are in set $\mathcal{B}$. $\mathcal{N}$ is the set of N-links edges(neighborhood links), each N-links $V_{p,q}(L_p, L_q)$ is weighted by $B_{p,q}$. Each link $D_{p,S}(L_p)$ between $\{p, S\}$ is weighted based on it's label, so as each link $D_{p,T}(L_p)$ between $\{p, T\}$. Then

$$\mathcal{E} = \mathcal{N} \cup \{p, T\} \cup \{p, S\} \tag{2-3}$$

For a cut $\mathcal{C}$, the resulting segmentation is defined as $A(C)$. There are two separated set $A(C)$. One set contains S, the other contains T. According to the weight listed in 2-1, the loss of a cut $\mathcal{C}$ is

$$\mid C \mid = \sum_{i \notin \mathcal{O} \cup \mathcal{B}} \lambda \cdot R_p(A_p(C)) + \sum_{\{p,q\} \in \mathcal{N}} B_{p,q} \cdot \delta(A_p(C), A_p(C)) \tag{2-4}$$

Table 2-1. Weights used in Graph Cuts (Boykov and Jolly, 2001)

edge	weight(cost)	for
$V_{p,q}(L_p, L_q)$	$B_{p,q}$	$\{p, q\} \in \mathcal{N}$
$D_{p,S}(L_p)$	$\lambda \cdot R_p(\text{"}bkg\text{"})$	$p \in \mathcal{P}, p \notin \mathcal{O} \cup \mathcal{B}$
	K	$p \in \mathcal{O}$
	0	$p \in \mathcal{B}$
$D_{p,T}(L_p)$	$\lambda \cdot R_p(\text{"}obj\text{"})$	$p \in \mathcal{P}, p \notin \mathcal{O} \cup \mathcal{B}$
	0	$p \in \mathcal{O}$
	K	$p \in \mathcal{B}$

The energy of a graph is defined as 2-5. If the weight defined as weight in Tabel 2-1, then a cut minimize the cost of the cut 2-1 equals to find a cut that minimize the energy of the resulted segmentation 2-2(Bell et al., 2015). Hence, with special defined graph weights, minimize cut equals to minimize graph energy.

$$E = \sum_{p \in \mathcal{P}} D_p(L_p) + \sum_{\{p,q\} \in \mathcal{N}} V_{p,q}(L_p, L_q) \tag{2-5}$$

There are many variations to both the unary term $D_p(L_p)$ and the pairwise term $V_{p,q}(L_p, L_q)$ (Rother et al., 2004; Li et al., 2004; Batra et al., 2010; Lin et al., 2016; Zhang et al., 2018a). In general, the unary term $D_p(L_p)$ indicate individual label-preferences. The pairwise term $V_{p,q}(L_p, L_q)$ encourage spatial coherence by smoothing label change between neighboring pixels.

A recent work Scribblesup (Lin et al., 2016) combines graph model with FCN. The output of FCN contribute to the unary term of cost function of the graph model. The FCN and the graph model is learned iteratively. The graph model propagates the scribble label to the unlabelled pixels. Then the FCN learns from such mask and in return update the cost function of the graph model. And then the graph model update a new mask for FCN. Then FCN and graph model keep on learning iteratively. The FCN provide learnable features that improve the segmentation results. As reported by the paper (Lin et al., 2016), the Scribblesup method achieve a 63.1% mean Intersection-over-Union (mIoU) score which is about 10% higher than GrabCut (Rother et al., 2004) and LazySnapping (Li et al., 2004).

2.1.3.2 Conditional Random Field Based Approach

The prediction made by learning model training with weakly supervision is less accurate than the prediction made by learning model training with full supervision. Conditional Random Field (CRF) (Krähenbühl and Koltun, 2011; Koller and Friedman, 2009; Bishop, 2006) is a common approach that people used to refine the segmentation of image.

CRF is a partially directed graph as shown in Fig.2-2. As a graph, it also has a set of nodes $\mathcal{V}$ and a set of edges $\mathcal{E}$. The shaded nodes $\mathbf{X}$ in blue is the set of observed variables corresponding to each pixels in image. Assuming the size of input image is N. Then $\mathbf{X} = \{X_1, ..., X_N\}$. The unshaded nodes $\mathbf{Y}$ is the set of latent variables corresponding to the label of each pixel. Therefore, $\mathbf{Y} = \{Y_1, ..., Y_N\}$. In CRF model, there is a directed edge from $\mathbf{X}$ to each label node Y_i. Also, Any pair of $\{Y_i, Y_j\}$ should be connected is linked by an undirected edge. In the illustration example Fig.2-2, only neighboring label nodes are connected.

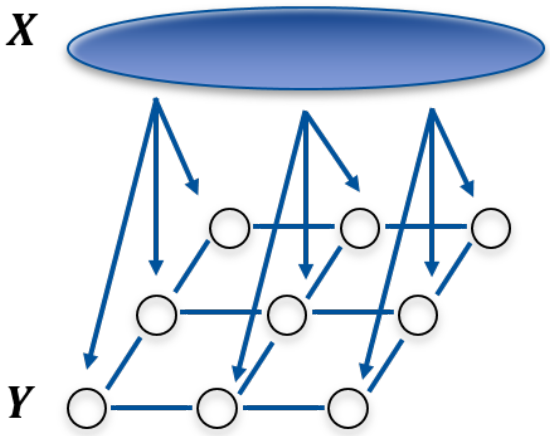


Figure 2-2. The illustration of CRF

The CRF is used to model a conditional probability of $\mathbf{Y}$ over $\mathbf{X}$ in 2-6 which can be factorized in terms of cliques. There are two kind of cliques, one is made by each pair of connected label nodes. The other is made by each label nodes and the observed nodes. Hence a CRF is factorized into Eq.2-7, where $\mathcal{M}$ is the set of connected label nodes, $\phi(Y_i, Y_j)$ is the possibility function of $\{Y_i, Y_j\}$, $\phi(Y_i, \mathbf{X})$ is the possibility function of Y_i given $\mathbf{X}$

$$P(\mathbf{Y}|\mathbf{X}) = \frac{1}{Z(\mathbf{X})} P(\mathbf{Y}, \mathbf{X}) \tag{2-6}$$

and

$$Z(\mathbf{X}) = \sum_{\mathbf{Y}} P(\mathbf{Y}, \mathbf{X})$$

$$P(\mathbf{Y}|\mathbf{X}) = \frac{1}{Z(\mathbf{X})} \prod_{i,j \in \mathcal{M}} \phi(Y_i, Y_j) \prod_{i=1}^{N} \phi(Y_i, \mathbf{X}) \tag{2-7}$$

When $\phi(Y_i, Y_j)$ and $\phi(Y_i, \mathbf{X})$ are in the form of Gibbs distribution, then $\phi(Y_i, Y_j) = exp(-\psi_p(Y_i, Y_j))$, $\phi(Y_i, \mathbf{X}) = exp(-\psi_u(Y_i))$. Substitute them into 2-7, then we got 2-8. E in the form of 2-9 is the Gibbs energy. The inference of pixel label is to find value of label nodes $\mathbf{Y}$ that maximum the value of 2-8 which equals to find value of label nodes $\mathbf{Y}$ that minimize the energy of CRF in 2-9.

$$P(\mathbf{Y}|\mathbf{X}) = \frac{1}{Z(\mathbf{X})} exp(-(\sum_{i=1}^{N} \psi_u(Y_i) + \sum_{i,j \in \mathcal{M}} \psi_p(Y_i, Y_j))) \tag{2-8}$$

$$E = \sum_{i=1}^{N} \psi_u(Y_i) + \sum_{i,j \in \mathcal{M}} \psi_p(Y_i, Y_j) \tag{2-9}$$

The unary potential $\psi_u(Y_i)$ predict the label distribution for a pixel independent of the label of other pixels. The pairwise potential $\psi_p(Y_i, Y_j)$ take into account the labels of other pixels when predict label of a given pixel. The unary term usually computed independently for each pixel by a classifier. The parameters of the unary term and pairwise term are learnt piecewisely. For example, in the paper (Krähenbühl and Koltun, 2011), the unary term is a classifier trained independent of the pairwise term with sample features including textons, colors, histogram of oriented gradients (HOG), and pixel location feature. The unary term is used to estimate the possibility of the label that a pixel was assigned to without considering the labels of other pixels. The pairwise term is defined as 2-10. $w^{(1)} exp(-\frac{|p_i - p_j|^2}{2\theta_\alpha^2} - \frac{|I_i - I_j|^2}{2\theta_\beta^2})$ is the appearance kernel that penalty close pixels have similar color but have different label. $w^{(2)} exp(-\frac{|p_i - p_j|^2}{2\theta_\gamma^2})$ is the smoothness kernel penalty close pixels but have different label to avoid small region. The pairwise term consider the structure of the data. The parameters in the pairwise term is usually learned by grid search.

$$\psi_p(Y_i, Y_j) = \mu(Y_i, Y_j)(w^{(1)} exp(-\frac{|p_i - p_j|^2}{2\theta_\alpha^2} - \frac{|I_i - I_j|^2}{2\theta_\beta^2}) + w^{(2)} exp(-\frac{|p_i - p_j|^2}{2\theta_\gamma^2})) \tag{2-10}$$

and

$$\mu(Y_i, Y_j) = [Y_i \neq Y_j]$$

Efficient Algorithm (Krähenbühl and Koltun, 2011)are developed to inference CRF that made it possible to refine the inaccurate labels of the unary classifier with CRF. Therefore, many weakly supervised image segmentation algorithm use CRF.

2.1.3.3 Convolution Neural Network Based Approach

Papandreou et al. (2015) proposed to use FCN to do weakly supervised image semantic segmentation. The FCN is iteratively trained with refined estimated pixel label. The estimated label is made by assigning the pixel to the class that has the largest score. The score is the output of the FCN fulfilled the weakly supervision. The weakly supervision are add to the score in two ways. First, a constant value is add to the score of the class that is the same as the image label for each pixel, then the pixel is assigned to the class that has the largest score. This method is called "EM-Fixed". The other way is that the constant value is added to a portion of the pixels that have larger score corresponding to the image label, then the pixel is assigned to the class that has the largest score. This method is called "EM-Adapt". Both method yield an undesirable results on weakly supervised image segmentation because the FCN is randomly initialized, the scoremap of FCN cannot estimated a good label for each pixel. Hence the "EM-Fixed" and "EM-Adapt" does not work well. Usually, pixel-level labeled data are needed to change the algorithm to semi-supervised learning. According to the paper (Papandreou et al., 2015), the mean IOU on validation set of semi-supervised method is 62 with 1000 pixel-level labeled semi-supervised training data. The mean IOU on validation set of weakly supervised method is 20.8 for "EM-Fixed" and 38.2 for "EM-Adapt". Instead of assigning object class label to a portion of pixels, Pinheiro and Collobert (2015) assumed that all pixels had the same label as image label. They added a constrain layer using Log-Sum-Exp function to weight the score of the class that is the same as the image class for all pixel during training. This is one way to aggregate score map into a classification scores. The Log-Sum-Exp function is a smooth version and convex approximation of the max function. Global max pooling (Oquab et al., 2015) and global average pooling (Zhou et al., 2016) are other popular ways to aggregate score map into a classification scores.

Attention map provides a better way to estimate pixel label. The attention map becomes the main scheme that people use CNN to do weakly supervised image segmentation. CNNs have been proven to be able to locate to the most discriminative regions when it was trained

to classify images. These discriminative regions are where the CNNs paid high attention to. Generally, there are two main schemes to compute attention. One is post hoc network analysis such as Class saliency map, Guided backpropagation, Class activation maps and Grad-CAM. The other is trainable attention mechanisms.

Class saliency map (CSM)(Simonyan et al., 2013) used for visualizing image classification model can also be used for indicating object location in images. The network is the exact function that map input X to the class score S_c. This method assumes that the class score $S_c(\hat{X})$ of an input $\hat{X}$ can be approximated by a linear function of input $\hat{X}$ Eq.2-11. Then the weight w indicate the importance of a pixel to be changed that would improve the class score. It has been showed that the more importance the pixel is, the higher possibility it located on the an object belong to the class of S_c. The weight can be computed by standard backpropagation. The value of CSM at each pixel is the weight of that pixel has the maximum absolute value across channel 2-12.

$$S_c(\hat{X}) = w^T \hat{X} \tag{2-11}$$

and

$$w = \frac{\partial S_c}{\partial X}|_{\hat{X}}$$

$$M_{ij} = max_c |w_{h(c,i,j)}| \tag{2-12}$$

Guided backpropagation (Springenberg et al., 2014) combine the idea of CSM and the deconvolutional network (Deconvnet)(Zeiler and Fergus, 2014). CSM consider the effect of all pixels having positive activation values that the gradient of ReLU at non-zero output is 1, otherwise is zero on the backward propagation. There are one or some convolution layers before the ReLU layer. A positive output of ReLU means the pixel and its neighborhood in input image contains the feature defined by previous convolution layer. Only consider the gradient of the non-zero output make sure that only the gradient of a region exhibiting the

features are considered. However, some neurons have negative local gradient which means such neural will made the decrease the activation of the higher layer neuron that we aim to visualize. Hence the constrain of Deconvnet are combined with the constrain of CSM that not only suppresses the gradient of the zero output of ReLU, but also suppresses the gradient of neurons having negative local gradient.

Class activation maps (CAMs)(Zhou et al., 2016) is one way to study the attention mechanism of CNNs. The CAMs is computed from a neural network for classification task. The neural network structure of CAM shown in 2-3 contains a stack of convolutional layers, a global average pooling layer (GAP), a fully connected layer (fc), and the output layer. The GAP and fc layers are used to combine features $\mathbf{f_{N\times H\times W}}$ computed by the convolutional layers. The fully connected layer provides a unique set of weight $\mathbf{W_m} = \{W_{m1}, ...W_{mN}\}$ for each object class m. The weight $\mathbf{W_m}$ characterizes the importance of each feature channel n to the object class m. The CAMs for class m is the summation of the $\mathbf{f_{N\times H\times W}}$ weighted by the $\mathbf{W_m}$ over feature channel n in Eq.2-13.

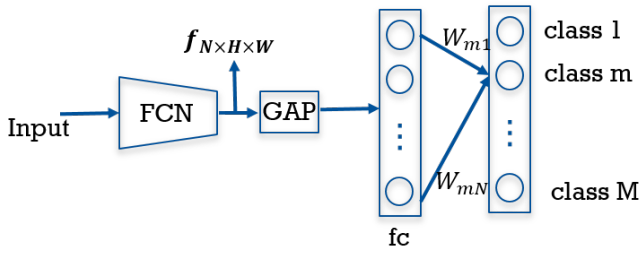


Figure 2-3. The illustration of neural network structure for CAM

$$M_m(x,y) = \sum_{n=1}^{N} W_{mn} f(n,x,y) \tag{2-13}$$

Grad-CAM (Selvaraju et al., 2017) is a generalization of CAM. CAM is able to use the weights of fc layer to weight each feature channel of the output of final convolution layer because the convolution layer is followed by a GAP and fc layer. Grad-CAM extends the work of CAM to neural network having general structure and can visualize attention at any layer. To visualize the attention map of class c having class score S_c at a layer with output $\mathbf{f_{N\times H\times W}}$, the

weight W_{ck} for the k feature channel of class c attention is computed by Eq.2-14. It is easy to prove that the weights of CAM can be computed by Eq.2-14. The attention map of Grad-CAM is computed by Eq.2-15.

$$W_{ck} = \frac{1}{Z}\sum_i \sum_j \frac{\partial S_c}{\partial f(k,i,j)} \tag{2-14}$$

$$M_c(x,y) = ReLu(\sum_k W_{ck} f(k,x,y)) \tag{2-15}$$

Grad-CAM and CAM compute the global weights to weighted feature vector of each pixel in the same way. They are a good way to visualize the class discriminative regions. CSM and guided backpropagation methods computed the importance of each individual pixel to the change of objective function. They are better to show importance of individual pixel. All of them are attention map computed based on the post hoc network analysis scheme. Those attention map are computed based on general purpose neural network structure. Another type of attention map based on trainable attention mechanisms is also extensively studied. Special attention modules are added to the neural network.

Two trainable attention models are proposed in paper (Hong et al., 2016) illustrated in Fig.2-4. One attention α^l of class s is computed by combine the one-hot label vector y_L^l for class l with feature f'_{NHW} in reduced dimension by matrix $W_{d\times NHW}$ and $W_{d\times L}$. Then weight by $W^{att}_{HW\times d}$. It equals that the attention of each pixel is a weighted sum over all values of feature map in Eq.2-18. This is good for pixel-wise attention that the author claimed that this attention map is too sparse that tend to only focus the most discriminative region in image.

$$\alpha^l = W^{att}_{HW\times d}(W_{d\times NHW} f'_{NHW} \odot W^{label}_{d\times L} y^l_L) \tag{2-16}$$

$$= W^{att}_{HW\times d} W^l_{d\times NHW} f'_{NHW} \tag{2-17}$$

$$= W^l_{HW\times NHW} f'_{NHW} \tag{2-18}$$

In order to get a densified attention, attention model 2 was proposed as Eq.2-20. The attention M_l at position $\{i, j\}$ are weighted by z^l. z^l represents the importance of each feature channel by averaging importance over all pixels per feature channel.

$$z^l = f''_{N\times HW}\alpha^l \tag{2-19}$$

$$M_l(i,j) = \sum_n z^l(n) f(n,i,j) \tag{2-20}$$

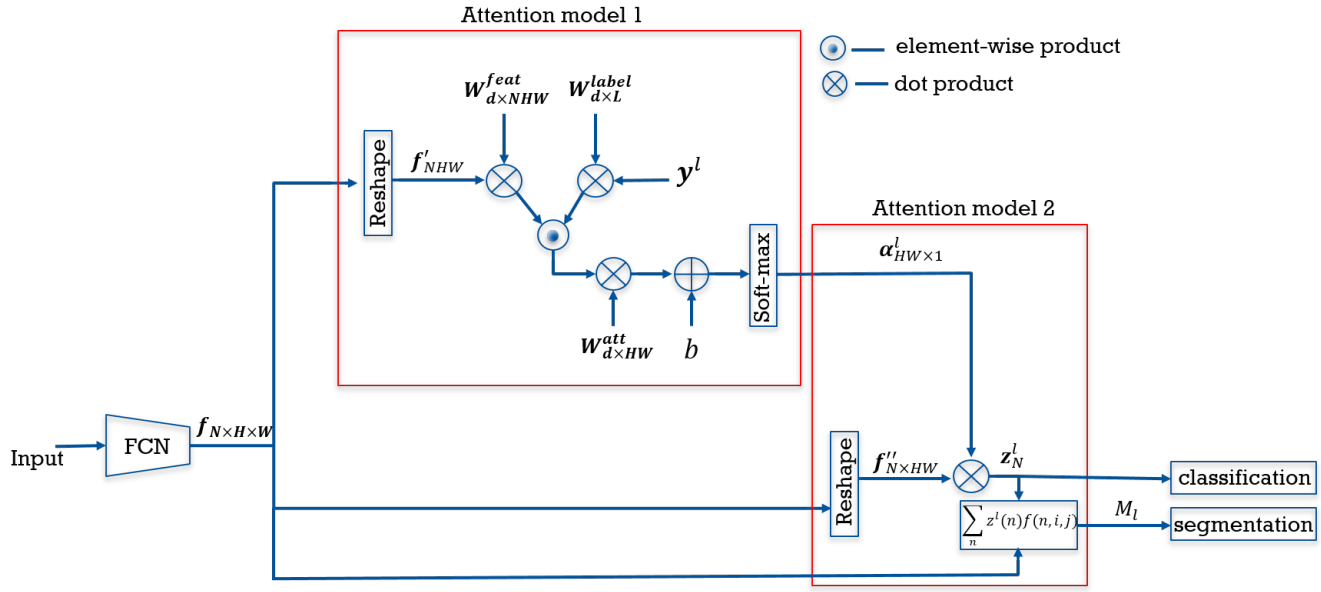


Figure 2-4. The illustration of trainable attention models

Recently, a new trainable attention model (Jetley et al., 2018) is proposed that was able to get attention from any layer illustrated in Fig.2-5. In order to compute the attention from layer L_n using local feature vector $Y^n \in R^{C_n\times H\times W}$ of layer L_n and global feature vector $Y^N \in R^{C_N}$, Y^N is first projected to lower dimension $\hat{Y^N} \in R^{C_n}$ with $\hat{Y^n} = Y^n$ or Y^n could be linear mapping to the dimension of $\hat{Y^n} \in R^{C_N\times H\times W}$ with $\hat{Y^N} = Y^N$. Then the attention of the lth class α^l is computed by Eq.2-21 as the normalized compatibility score by softmax function. Two ways are proposed to compute the compatibility score. One is weighted the

local pixel feature by global pixel feature as Eq.2-22. The other is learning a vector μ to weight the combination of local pixel feature vector and global feature vector as Eq.2-23.

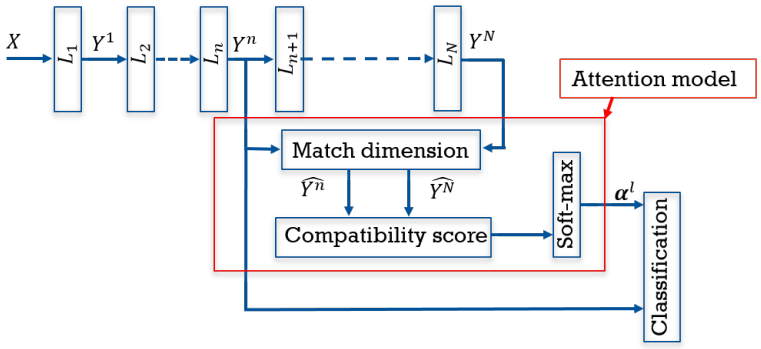


Figure 2-5. The illustration of trainable attention models

$$\alpha^l(i,j) = \frac{exp(c^l_{i,j})}{\sum_i \sum_j exp(c^l_{i,j})} \tag{2-21}$$

$$c^l_{i,j} = \sum_r \hat{Y^n}(r,i,j)\hat{Y^N}(r) \tag{2-22}$$

$$c^l_{i,j} = \sum_r \mu(r)(\hat{Y^n}(r,i,j) + \hat{Y^N}(r)) \tag{2-23}$$

Kolesnikov and Lampert (2016) used CAM (Zhou et al., 2016) to predict possible localization of foreground class and CSM (Simonyan et al., 2013) to predict localization of background class. 20% pixels having largest value in the attention map of CAM are assigned the foreground object label. 10% of the least salient locations in the attention map of CSM are assigned the background label. Such segmentation mask for each image is used to train neural network together with expansion and boundary constrain. The expansion constrain is similar to the work of Pinheiro and Collobert (2015), but a gloal weighted rank pooling is used to aggregate score map into a classification score. The boundary constrain takes the form of enegery of CRF (Krähenbühl and Koltun, 2011). Although this method combine FCN and CRF, all parameters of CRF are pretrained that the FCN and CRF are not training together.

Roy and Todorovic (2017) proposed a new way to compute attention. Rectified Gaussian distribution $P(\mathbf{p})$ is used to model the joint probability of the value of each pixel $\mathbf{p}^T = \{p_1, ..., p_N\}$ in the attention map. The matrix D structured the dependnecies between neighboring neural activations which is similar to the pairwise term of CRF. The vector $\mathbf{b}$ considered the dependencies on the activations from one layer above which is similar to the unary term of CRF. The normalized value of $\mathbf{p}$ which maximize the joint probability $P(\mathbf{p})$ is defined as the Top-Down attention map. Such Top-Down attention map is combined with CAM (Zhou et al., 2016) and refined by CRF to predict the segmentation scores of each pixel. During training, a fc layer is used to aggregate pixel scores to classification scores.

$$P(\mathbf{p}) \propto exp(\frac{1}{2}\mathbf{p}^T D\mathbf{p} + \mathbf{b}^T\mathbf{p}), \mathbf{p} \geq \mathbf{0} \tag{2-24}$$

Hong et al. (2017) use image having image level label together with video having video level label as weakly labelled training data. Their model contains a encoder and a decoder. The encoder part is the same as CAM (Zhou et al., 2016) and is trained with images having image level label. The encoder is used to select frames having object class from weakly labelled video and generate attention map for the selected frame. Then graph cut method (Boykov and Jolly, 2001; Boykov and Kolmogorov, 2004) is used to make the segmentation mask. The attention map is used as the unary term in the energy function. And then, the video frame and corresponding segmentation mask is used to train the decoder to predict label per pixel. This method takes the advantage that video data has more clues to locate the mvoing foreground object than image. It achieves a good segmentation result.

Wei et al. (2018) proposed a method that use attention map for weakly supervised image segmentation with image level label. Two estimated pixel labels per pixel are computed used to compute loss respectively. One used the method the same as "EM-Adapt". The other is generated use attention map. A portion of pixels have larger value on the attention map is assigned target labels, others are background. The attention map is similar to CAM. But the spatial resolution of CAM is relatively small due to the pooling layer of FCNN. Hence,

Wei et al. (2018) reduced one pooling later and extended the last convolution layer to a multiple dilated convolution blocks with different dilated rates. If only use the output of dilated convolution blocks with dilated rates as 1 to compute the attention map, it is the same as CAM. Instead, the attention map is computed as the summation of the average of attention map H_{avg} of dilated convolution blocks with dilated rates other than 1 and the attention map H_1 of dilated convolution blocks with dilated rates as 1. This method using attention map is superior to the results of "EM-Adapt" that the mean IOU on validation set for this method is 60.4 but only 38.2 for "EM-Adapt" (Papandreou et al., 2015).

Tang et al. (2018) added the normalized cut loss $loss_{nc}$ as the regularized term to the loss function L. $loss_1$ is the cross entropy loss over labeled data. The normalized cut loss term tries to avoid partitions in small size and maximize the similarity within class. Lower level features $\mathbf{F}$ including color in RGB space and location in space are used to compute the $loss_{nc}$ based on the predicted label of pixels belong to the kth class $\mathbf{S^k} \in [0, 1]$. This normalized cut loss is motivated by the success of normalized cut in unsupervised segmentation that a good segmentation tends to have small normalized cut loss.

$$L = loss_1 + loss_{nc} = loss_1 + \sum_k \frac{\mathbf{S^k}^T W(\mathbf{1} - \mathbf{S^k})}{\mathbf{d}^T \mathbf{S^k}} \tag{2-25}$$

$W = [W_{ij}]$ is the similarity matrix between the ith node and jth node. Gaussian kernel are used to compute the similarity between nodes. σ_c is a hyperparameter for the cth feature F_c and $\mathbf{d} = W\mathbf{1}$.

$$W_{ij} = e^{-\sum_c \frac{\|F_c(i) - F_c(j)\|_2^2}{\sigma_c}} \tag{2-26}$$

2.2 MR Image

2.2.1 Minirhizotron Root Image Segmentation

Segment roots from minirhizotron image is important to the analysis of the root systems. However, automated root segmentation is a challenge problem because the complex soil background, the variations in illumination, the complicated root shape and size, and so on.

2.2.1.1 Unsupervised Root Segmentation

Shojaedini and Heidari (2013) used second-order cross entropy and level set method to do root segmentation. This method first thresholds the image into foreground and background using second-order cross entropy. The threshold is the value that maximize the second-order cross entropy. Co-occurrence matrix (Zeng et al., 2006) is used to compute the second-order cross entropy. The thresholded image is the image that the value of pixels in the foreground part is kept the same as the original image. The value of pixels in the background part is set to zero. Then the level set method is applied to the thresholded image. The boundary between the foreground and background in the thresholded image is used as the initial boundary for the level set method. The boundary is keep on refining by the level set method until reach a optimal solution to the level set method.

RAHMANZADEH and Shojaedini (2016) further improved the work of Shojaedini and Heidari (2013) by applying the curvelet transform to pre-process the image, then applying the method of Shojaedini and Heidari (2013) to automatically segment root from minirhizotron image. First, the curvelet coefficients is computed. Then, the curvelet coefficients is changed by a mapping function to improve the performance of the detection. After that, inverse curvelet transform on the mapped curvelet components are done to recover the image. The recovered image is the pre-preocessed image for further root segmentation by the method mentioned above.

2.2.1.2 Supervised Root Segmentation

Zeng et al. (2006) proposed a pipeline to automatically extract and measure roots in minirhizotron images. Their method rely on training data with label of root components in the binarized feature map. Their extracting root approach used the green band of RGB minirhizotron images and included five steps. In the pre-processing step, linearly stretching is used to enhance the contrast of the image. Then in the second step, Matched filtering is used to extract features of root objects. The root objects is modeled by match filter as a linear segments of constant width. The value of root objects across the width direction is represent

by Gaussian kernels at different orientations and scales. The orientations are set to be values between 0° to 180°, spaced 15° apart. The scales are set to be 2 and 4. The length of the linear segments is 11 pixels. These match filters generate 24 feature maps per image. Each feature map is processed differently in the next two steps which threshold the feature maps into components and discriminate the root components. The feature maps are first quantized into 256 levels. Then local entropy thresholding (LET) is used to threshold each feature map. The LET method find the threshold that maximized the background and foreground entropy of the co-occurrence matrix of each feature map. The co-occurrence matrix the a 2D matrix each elements at position (i, j) is the number of pixels in training data with value i in feature map and the neighbor of that pixel to the right and below has value j in the feature map. The larger the entropy is, the more compact in feature space the class will be. The author claimed that LET is superior to common thresholding method such as Otsu's. After threshold the feature map, five measures are computed to discriminate root components from others: (1) the percentage of pixels in the region with an intensity value greater than 205; (2) the percentage of intensity edges in the region; (3) the eccentricity of the region; (4) the symmetry of the region around its central curve as measured by the percentage of pixels along the curve whose corresponding boundary pixels are equidistant; (5) the percentage of pixels along the curve whose boundary pixels share parallel tangents. AdaBoost algorithm used such features to classify each components. After then, an extra step was applied to combine components belongs to same root together based on predefined rules. After the root is extracted, they measure the length and diameter as root attributes.

Zeng et al. (2010) also proposed a rapid root segmentation method. This rapid method based on seed points selection. Seed points are points on roots. The author assumed that the candidate seed points are those points which are the local maxima on pre-processed image. This method including four steps: (1) pre-processing step; (2) seeds selecting step; (3)root centerline detection; (4) root region detection. In the pre-processing step, the images are first processed by a lowpass Gaussian filter to remove noise. Then the images are downsampled

by $N \times N$ times. In the selecting seed points step, local maxima in the pre-processed image are first selected as candidate seeds. Then the height and breadth features of each candidate seeds are used in a linear classification model to filter out the seeds that are not on roots. The height is the normalized gray level intensity value and the breadth is the image distance between the two neighboring local minima on either side of the maximum. The author claimed that such a classifier achieved 92% true positive rate on the training set. Once the seeds are selected, the seeds are connected into root centerline in the following root centerline detection step. In this step, seeds are first put into groups and make the centerline, then each group is validated if they are true root centerline. Finally, the compatible centerline are connected. To group root together, a seed is randomly selected. The selected seed and its closest neighbor seed make a line segment. Then the spread and residue features of other seeds to this line segment are used by a linear classifier to group seeds belongs to this line segment together. Such a process to group seeds belong to same line segment is continued until all seeds are put into a group. The spread feature is average distance between neighboring point in the seeds group, with neighbors defined by fist ordering the points according to their projection onto the segment. The residue feature is defined as the mean squared error of the points with respect to their distance from the line segment. The the support and width stability of each group seeds are computed and used by another linear classifier to filter out the false root centerline. The width stability is the percentage of pixels whose width is within some tolerance of the estimated root width. Finally, any two line segments are combined together is the spread and residue features of the combined group is smaller than the summation of this features of each group. Proximity and alignment are also used to combine centerline. After the root centerline is detected, constrained floodfill is applied to the centerlines to extract the root region. The floodfill method is a region grow method that assume all pixel within a distance to the centerline having pixel value larger than the smalles pixel value of the pixels on the centerline are root. After region grow, a further combination is done to combine two segments that are largely overlapped.

Wang et al. (2019) developed a method using a neural network named SegRoot to do root segmentation. The SegRoot is based on the structure of SegNet (Badrinarayanan et al., 2017). SegRoot is trained with soybean root images. The soybean root image dataset contains 65 images paired with groundtruth. 39 images are used for training, 13 images are used for validation, and 13 images are used for testing. Subimages of size 256×256 are cropped from original image for the requirement of the input data size of SegRoot. Data augmentation is also applied to the subimages. The data augmentation is done by randomly apply one of the five image transformations to the subimage including: horizontal flipping, vertical flipping, 90° rotation, 180° rotation and 270° rotation. The author found that there are many fine roots having width just 1 pixel. It is hard to extract features fro such thin objects. They claimed that it is helpful to dilate the groundtruth masks that define a target object made by the root and regions nearby root. The SegRoot aimed at finding pixels of root and its nearby region by maximize the dice score of training data. During test, the segmentation is made by thresholding and eroding the target class scoremap. Several different hyperparameter (width and depth) values of SegRoot are being examined to find the best network structure with respect to speed and accuracy. Due to the small number of training images, transfer learning is used that the encoder part of the network is initialized by the weights pre-trained with ImageNet. Their results show that the SegRoot could not only find roots on the test dataset, it is also helpful to find roots of other type of plant that the network has never seen.

Yasrab et al. (2019) built the RootNav 2.0 for extraction of root system architecture. The RootNav 2.0 has three functions: (1) segment first order root and second order root from minirhizotron image; (2) detect key points of root including seed, first and second-order root tips; (3) construct plant root architectures. The neural network model used in RootNav 2.0 is based on the hourglass networks. The hourglass networks is a stacked of hourglass module (Newell et al., 2016; Pound et al., 2017; Tompson et al., 2014) and residual module (He et al., 2016). The root segmentation and key points detection are jointly learning. The loss corresponding to segmentation and key points are added together. The network is trained

on a dataset of Wheat images from scratch contains 3630 images. 2864 images are used for training, 716 images are used for validation, 50 images are used for test. Data augmentation are also used to increase training data and avoid overfitting. Horizontal flipping, rotation in the range $[-30°, 30°]$ are applied to training data randomly with 50% probability. CRF (Krähenbühl and Koltun, 2011) is used for post-processing to smooth the segmentation results. Experiments on smaller dataset are also be done via transfer learning. These smaller dataset include a dataset of arabidopsis images and a dataset of rapeseed images. The arabidopsis images are divided into three parts. 200 arabidopsis images are used for training, 27 images are used for validation, 50 images are used for testing. For the rapeseed dataset, 91 rapeseed images are used for training, 14 images are used for validation, and 15 are used for testing.

2.2.1.3 Weakly Supervised Root Segmentation

Weakly supervised root segementation has been studied by Yu et al. (2019) that image-level ground truth is used to do root segmentation on switchgrass minirhizotron images. The method include four steps: (1) pre-processing image; (2) feature extraction; (3) root segmentation via multiple instance learning (MIL) approach; (4) post-processing. In the pre-processing step, striping noise caused by the sensors in the scan head are differently calibrated are removed firstly. The column mean of each image column is subtracted from that column in each image and the global image mean is added to each image. Then the pre-preocessed images are oversegmented into superpixels using the Simple Linear Iterative Clustering (SLIC) algorithm (Achanta et al., 2012). The superpixels group nearby pixels which are similar in color together. The feature extraction step extracts features of superpixels. The mean, variance, and entropy of each band in RGB and LAB color spaces ([mean-R, mean-G, mean-B, mean-L, mean-a, mean-b, var-R, var-G, var-B, var-L, var-a, var-b, H-R, H-G, H-B, H-L, H-a, H-b]) are computed to make an 18-dimensional feature vector for each superpixel. Each superpixel feature vector is then scaled by their order of magnitude so that they are normalized for equal weight across the features. After then, MIL classifier are used to do root segmentation. The training data are labeled as positive bags and negative bags for multiple

instance learning. Each image having root objects is labeled as a positive bag. Each image contains pure soil is labeled as negative bag. The superpixels in each images are the instances in the bag of that image. Three MIL algorithms are investigated to classified each superpixel: the multiple instance adaptive cosine/coherence estimator (MI-ACE) (Zare et al., 2017), the multiple instance support vector machine (miSVM) (Andrews et al., 2003), and the multiple instance random forest, (MIForests) (Leistner et al., 2010).

miSVM(Andrews et al., 2003) is developed based on the support vector machine (SVM) method (Cortes and Vapnik, 1995; Chang and Lin, 2011). SVM is a classifier that maximizes the margin between data from different classes which are the closest to the decision boundary. The miSVM applies the SVM classifier on data selected from negative bag and positive bag as Eq.2-27 with constrain Eq.2-28:

$$\arg\max_{\mathbf{w},b}\{\frac{1}{\|\mathbf{w}\|}\min_{s(i)}[t_{s(i)}(\mathbf{w}_T\phi(\mathbf{x}_{s(i)})+b)]\} \tag{2-27}$$

$$s.t.\forall\ s(i) : t_{s(i)}(\mathbf{w}_T\phi(\mathbf{x}_{s(i)})+b) \geq 1 \tag{2-28}$$

$\mathbf{w}$ and b are the parameters of the linear classifier model $y(\mathbf{x}) = \mathbf{w}_T\phi(\mathbf{x}) + b$ in the feature space $\phi(\mathbf{x})$ of data $\mathbf{x}$. $s(i)$ are the selected data from bag B_i. $t_{s(i)}$ is the label of the selected data in bag B_i. If data are given the positive label, $t_{s(i)} = 1$, otherwise, $t_{s(i)} = -1$. The miSVM algorithm defined the rule to select data from positive and negative bag. For data in negative bag, all data are selected and given the negative label. For data in positive bag, at the initial step, all data are selected and given the positive label, but in the following training iteration $iter > 1$, the miSVM classifier trained in iteration $iter - 1$ is used to assign a label to each data in positive bag. If in iteration $iter > 1$, all data in positive bag B_i are classified as negative label, the one having the highest probability belongs to the positive class are given the positive label. The miSVM algorithm is described in Alg.1

MIForests (Leistner et al., 2010) is a random forests classifier F (Hastie et al., 2005) with multiple instance learning constrain. MIForests learns a random forests model which

Algorithm 1: miSVM

Data: $(B,t) = \{(B_1,t_1),...(B_i,t_i),...(B_I,t_I)\}$;
$B_i = \{x_{i1},...x_{in},...x_{iN}\}$ with $x_{s(i)} \in B_i$
Result: $(\mathbf{w},b)$
initialize $(x_{s(i)},t_{s(i)}) = (x_{in},t_i)$;
repeat
  compute SVM solution $(\mathbf{w},b)$ with training data ;
  set $t_{in} = sgn(\langle \mathbf{w},x_{in}\rangle + b)$ for every x_{in} in positive bag ;
  for *every positive bag* B_i **do**
    if $\sum_n \frac{(1+t_{in})}{2} == 0$ **then**
      $m = \arg\max_n(\langle \mathbf{w},x_{in}\rangle + b)$;
      $t_{im} = 1$
    end
  end
until *labels for data in positive bag do not change*;

minimizes the loss of training data L as Eq.2-29 by consider the multiple instance learning constrain Eq.2-30 that at least one data x_{in} in each bag i has to be from the target class t_i. Where $\mathbb{1}$ is the indicator function.

$$\arg\min_F \sum_i \sum_n L(F(x_{in})) \tag{2-29}$$

$$s.t. \forall\ i : \sum_n \mathbb{1}\{t_i = \arg\max F(x_{in})\} \geq 1 \tag{2-30}$$

Deterministic annealing (DA) are used to optimize this problem depicted in Eq.2-29 and 2-30. The DA method add a convex entropy term to the objective function Eq.2-29 that extend the non-convex optimization problem to an easier one as Eq.2-31. where T is called temperature that control the weight between the first term and the second term, k is the number of class in training data, H is the entropy function. $p(k|x)$ is the probability distribution of labels which are aimed to be optimized.

$$\arg\min_{p,F} \sum_i \sum_n \sum_k p(k|x_{in})L(F(x_{in})) + T\sum_i H(p_i) \tag{2-31}$$

The MIForest model is trained in an iterative way. In the initial step, all data in bag are given the bag label. But in the following training iteration $iter > 1$, the $p(k|x)$ learned in iteration $iter - 1$ is used to assign a random label to each data. If in iteration $iter > 1$, all data in positive bag B_i are classified as negative label, the one having the highest probability belongs to the positive class are given the positive label. The training process stops until the current temperature smaller than the minimum temperature. The MIForest algorithm is described in Alg.2.

Algorithm 2: MIForest

Data: $(B,t) = \{(B_1,t_1),...(B_i,t_i),...(B_I,t_I)\}$;
$B_i = \{x_{i1},...x_{in},...x_{iN}\}$
Input: starting temperature T_0
minimum temperature T_{min}
cooling function $T_{m+1} = c(T_m)$
Result: $p(k|x), F$
initialize $(x_{in}, t_{in}) = (x_{in}, t_i)$;
initial iter $m = 0$;
repeat
  train the random forest model F with training data ;
  update temperature $T_{m+1} = c(T_m)$;
  update iter $m = m + 1$;
  compute $p(k|x)$;
  random select label t_{in} according to $p(k|x)$ for every bootstrap sample ;
  set the label for data with highest $p(k|x)$ in each bag equal to bag label ;
until $T_{m+1} \leq T_{min}$;

MI-ACE (Zare et al., 2017) uses Adaptive Cosine Estimator (ACE) as Eq.2-32 to detect targets in positive bags. MI-ACE learns a target signature $\mathbf{s}$ to differenciate targets from background in data $\mathbf{x}$. The background is characterized by a Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_b, \Sigma_b)$.

$$D_{ACE}(\mathbf{x},\mathbf{s}) = \frac{\mathbf{s}^T\Sigma_b^{-1}(\mathbf{x}-\boldsymbol{\mu}_b)}{\sqrt{\mathbf{s}^T\Sigma_b^{-1}\mathbf{s}}\sqrt{(\mathbf{x}-\boldsymbol{\mu}_b)^T\Sigma_b^{-1}(\mathbf{x}-\boldsymbol{\mu}_b)}} \tag{2-32}$$

The signature $\mathbf{s}$ should maximize the similarity between signature to target and minimize the similrity between signature to background in training data as Eq.2-33. N^+ and N^- are the number of positive and negative bags, N_j^- is the number of data in negatvie bag B_j. One data

from each positive bag is selected as target $\mathbf{x}_{s(i)}$ in Eq.2-33. $\mathbf{x}_{s(i)}$ is the most simialr dta in positive bag to signature. All data in each negative bag are treated as background x_j.

$$\arg\max_{\mathbf{s}} \frac{1}{N^+}\sum_{i,t_i=1} D_{ACE}(\mathbf{x}_{s(i)},\mathbf{s}) - \frac{1}{N^-}\sum_{j,t_j=-1}\frac{1}{N_j^-}D_{ACE}(\mathbf{x}_j,\mathbf{s}) \tag{2-33}$$

An efficient algorithm is developed to solve Eq.2-33 by decomposition D_{ACE} as Eq.2-34. The optimization solution of s^* is Eq.2-35 by taking derivative of Eq.2-33 and set it to zero.

$$\begin{aligned} D_{ACE}(\mathbf{x},\mathbf{s}) &= \frac{\mathbf{s}^T\Sigma_b^{-1}(\mathbf{x}-\boldsymbol{\mu}_b)}{\sqrt{\mathbf{s}^T\Sigma_b^{-1}\mathbf{s}}\sqrt{(\mathbf{x}-\boldsymbol{\mu}_b)^T\Sigma_b^{-1}(\mathbf{x}-\boldsymbol{\mu}_b)}} \\ &= \frac{\mathbf{s}^T\mathbf{U}\mathbf{D}^{\frac{-1}{2}}\mathbf{D}^{\frac{-1}{2}}\mathbf{U}^T(\mathbf{x}-\boldsymbol{\mu}_b)}{\sqrt{\mathbf{s}^T\mathbf{U}\mathbf{D}^{\frac{-1}{2}}\mathbf{D}^{\frac{-1}{2}}\mathbf{U}^T\mathbf{s}}\sqrt{(\mathbf{x}-\boldsymbol{\mu}_b)^T\mathbf{U}\mathbf{D}^{\frac{-1}{2}}\mathbf{D}^{\frac{-1}{2}}\mathbf{U}^T(\mathbf{x}-\boldsymbol{\mu}_b)}} \\ &= (\frac{\hat{\mathbf{s}}}{\|\hat{\mathbf{s}}\|})^T(\frac{\hat{\mathbf{x}}}{\|\hat{\mathbf{x}}\|}) \\ &= \hat{\hat{\mathbf{s}}}^T\hat{\hat{\mathbf{x}}} \end{aligned} \tag{2-34}$$

where $\hat{\mathbf{s}} = \mathbf{D}^{\frac{-1}{2}}\mathbf{U}^T\mathbf{s}$, $\hat{\mathbf{x}} = \mathbf{D}^{\frac{-1}{2}}\mathbf{U}^T(\mathbf{x}-\boldsymbol{\mu}_b)$

$$t = \frac{1}{N^+}\sum_{i,t_i=1}\hat{\hat{\mathbf{x}}}_{\mathbf{s(i)}} - \frac{1}{N^-}\sum_{j,t_j=-1}\frac{1}{N_j^-}\hat{\hat{\mathbf{x}}}_{\mathbf{j}} \tag{2-35}$$

The MI-ACE algorithm iteratively update $\mathbf{s}$. At the initial step, the signature s is picked as the data in positive bag that maximize the objective function Eq.2-33. Then, in the following training iteration $iter > 1$ the signature is coumputed by Eq.2-35 where the targets in positive bags are picked using signature learnt at $iter - 1$ using Eq.2-32. The MI-ACE algorithm is described in Alg.3.

During test, after each superpixel is classified by the trained model, a threshold is set to binarize the classification scoremap. Then connected component analysis is applied. The connected components having small eccentricity and size are filtered out to reduce false roots.

Algorithm 3: MI-ACE

Data: $(B,t) = \{(B_1,t_1),...(B_i,t_i),...(B_I,t_I)\}$;
$B_i = \{x_{i1},...x_{in},...x_{iN}\}$

Result: $\mathbf{s}$

initialize $\mathbf{s_0}$ as the instance in a positive bag resulting in largest objective function value;

repeat

update target selected from each positive bag using Eq.2-32 ;

update sigature $\mathbf{s}$ using Eq.2-35 ;

until *meet stop criteria*;

CHAPTER 3
ALGORITHM

In this chapter, we present our proposed method that using multiple instance neural network to do minirhizotron image segmentation. Our multiple instance learning network is developed to deal with image semantic segmentation problem with image level label. In the first section, we describe our multiple instance neural network structure. In the second section we will derive the optimization algorithm used in multiple instance neural network.

3.1 Multiple Instance Learning Neural Network

Multiple instance learning neural network (MIL-NN) is built based on the FCNN with the multiple instance layer. MIL-NN uses training data with image level label. If the ith image has roots, the label of this image $t_i = 1$, corresponding to a positive bag. If the ith image has pure soil, the label of this image $t_i = 0$, corresponding to a negative bag. Each pixel in a image is an instance in the corresponding bag. MIL-NN learns a model defined by parameters $\boldsymbol{\theta}$ that minimize the loss function $\mathcal{L}$ over data $\mathbf{x}_{s(m,n,i)}$ selected from bag B_i with label t_i as Eq.3-1 by satisfying the multiple instance condition Eq.3-2. In Eq.3-2, $\mathbb{1}$ is a indicator function, $f(x_{m,n,i};\boldsymbol{\theta})$ is the pixel score which represents the pixel score of data $x_{m,n,i}$ from bag B_i. The Eq.3-2 is a constrain requires that at least one instance in the bag has the same label as it's bag.

$$\arg\min_{\boldsymbol{\theta}} \mathcal{L}(\theta; \mathbf{x}_{s(m,n,i)}, t_i) \tag{3-1}$$

$$s.t. \forall\ i : \sum_m \sum_n \mathbb{1}\{t_i = \arg\max f(x_{m,n,i};\boldsymbol{\theta})\} \geq 1 \tag{3-2}$$

The MIL-Layer meet the multiple instance learning constrain by selecting data $\mathbf{x}_{s(m,n,i)}$ from bags (images) as Eq.3-3. If it is a negative bag $t_i = -1$, All data are selected for training with label the same as bag label. If it is a positive bag $t_i = 1$, We have proposed multiple ways to select data for training: (1) pick the top n pixels as targets; (2) pick the data above a threshold as targets; (3) pick a portion of data as targets.

$$\mathbf{x}_{s(m,n,i)} = MIL\_Layer(\mathbf{x}, \hat{f}(k)) \tag{3-3}$$

The multiple instance layer (MIL-Layer) is implemented based on the pixel scoremap $\hat{f}(k) = f(k|\mathbf{x};\boldsymbol{\theta})$. We have proposed several ways to compute the pixel scoremap: (1) using FCNN output as pixel scoremap; (2) using MIACE prediction result as pixel scoremap; (3) using attention map as pixel scoremap.

3.1.1 Multiple Instance Learning Neural Network with FCNN Output

The FCNN output of the target class is firstly used as pixel scoremap for the MIL-Layer. FCNN, taking the encoder-decoder structure with softmax layer, has been widely used for supervised semantic segmentation. The output of softmax layer is the probability distribution over classes of a pixel which makes the pixel scoremap. The output of softmax layer corresponding to the target class provides a way to propose the most like target (root pixel) in a positive bag (a image having roots). The MIL-Layer uses the output of FCNN of the target class to select data as targets from positive bags during training by following strategies mentioned above by Eq.3-3.

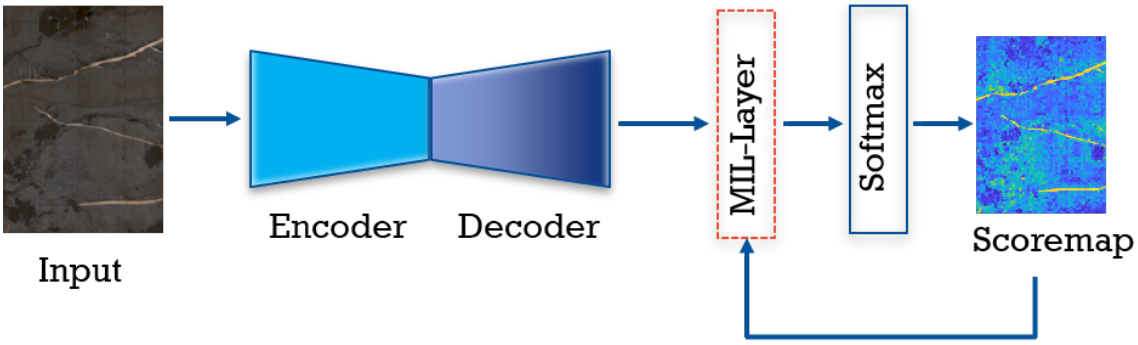


Figure 3-1. The illustration of multiple instance learning neural network with FCNN

The structure of MIL-NN with FCNN output is illustrated in Fig.3-1. FCNN is the neural network without the MIL-Layer in Fig.3-1. An MIL-Layer is inserted between the decoder block and the softmax layer during training. The MIL-NN with FCNN is trained iteratively. In the initial step of training, the neural network parameters $\boldsymbol{\theta}$ are initialized. The output of

FCNN of the target class is computed as the initial pixel scoremap with initial neural network parameters. The training data are selected from bags according to the initial pixel scoremap. Then the parameters of neural network are updated. For iteration $iter > 1$, The pixel scoremap is computed using parameters $\boldsymbol{\theta}$ learned at $iter - 1$ to select training data, and retrain the neural network until it meet the stop criteria. The algorithm of MIL-NN with FCNN output is depicted by Alg.4.

Algorithm 4: MIL-NN with FCNN

Data: $(B, t) = \{(B_1, t_1), ...(B_i, t_i), ...(B_I, t_I)\}$;
$B_i = \{x_{i1}, ...x_{in}, ...x_{iN}\}$
Result: $\boldsymbol{\theta}$
initialize $\boldsymbol{\theta}$;
repeat
  compute the pixel scoremap using FCNN ;
  update target selected from each positive bag based on the scoremap ;
  update network parameters $\boldsymbol{\theta}$;
until *meet stop criteria*;

3.1.2 Multiple Instance Learning Neural Network with MI-ACE

MI-ACE is another method that we used to get the pixel scoremap. MI-ACE learns a target signature from training data. Then the ACE detector use the target signature to decide the probability that a data belongs to the target class. In our case, we pre-train a root signature using image with image level label. Then the pixel scoremap is made by apply ACE detector to every pixel in the image as described by Yu et al. (2019). The MIL-Layer uses the scoremap of ACE detector to select data as targets from positive bags. The MI-ACE scoremap could be used only at the initial step and then replaced by FCNN scoremap in the following iterations. The MI-ACE scoremap could also be used during the whole training process.

The structure of MIL-NN with MI-ACE is illustrated in Fig.3-2. FCNN is the neural network connected by the blue line with arrow in Fig.3-2. An MIL-Layer is inserted between the decoder block and the softmax layer during training with prediction result using a pre-trained MI-ACE detector as one input. The MIL-NN with MIACE is trained iteratively. In the initial step of training, the neural network parameters $\boldsymbol{\theta}$ are initialized. The pretrained MI-ACE

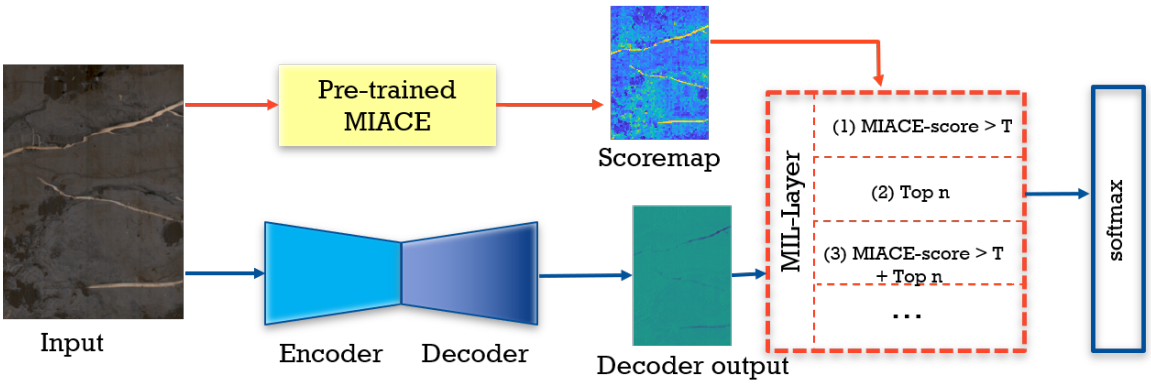


Figure 3-2. The illustration of multiple instance learning neural network with MI-ACE

detector is used to compute the scoremap of each image. The training data are selected from bags according to the initial pixel scoremap. Then the parameters of neural network are updated. For iteration $iter > 1$, either Using the MI-ACE scoremap continuously, or using the pixel scoremap computed by FCNN with parameters $\boldsymbol{\theta}$ learned at $iter - 1$ to select data. And then retrain the neural network until it meet the stop criteria. The algorithm of MIL-NN with MI-ACE is depicted by Alg.5.

Algorithm 5: MIL-NN with MIACE

Data: $(B, t) = \{(B_1, t_1), ...(B_i, t_i), ...(B_I, t_I)\}$;
$B_i = \{x_{i1}, ...x_{in}, ...x_{iN}\}$
Result: $\boldsymbol{\theta}$
randomly initialize $\boldsymbol{\theta}$;
load pre-trained MI-ACE root model;
compute the pixel scoremap using MI-ACE ;
repeat
  (optional) compute the pixel scoremap using FCNN ;
  update target selected from each positive bag based on the scoremap ;
  update network parameters $\boldsymbol{\theta}$;
until *meet stop criteria*;

3.1.3 Multiple Instance Learning Neural Network with Attention Map

We also use attention map as the pixel scoremap. Attention map has been studied to find the important region in a image for the image classification problem. It is a way to understand and visualize neural network. Attention map has also been used for weakly supervised image

semantic segmentation since it is possible to highlight the the object related region. As mentioned in chapter 2, there are multiple ways to compute attention map. In our proposed method, we use the CAMs attention map (Zhou et al., 2016) of root class as target pixel scoremap. A CNN used for image classification is pretrained to classify image into root image or soil image. Then the pretrained CNN is used to compute the CAMs attention map of root class as pixel scoremap. The MIL-Layer uses the scoremap of CAMs to select data as targets from positive bags.

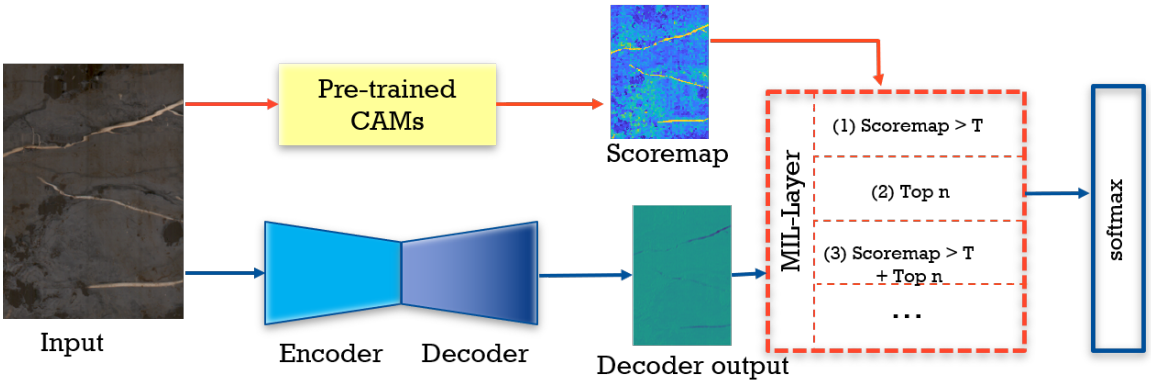


Figure 3-3. The illustration of multiple instance learning neural network with attention map

The structure of MIL-NN with attention map is illustrated in Fig.3-3. FCNN is the neural network connected by the blue line with arrow in Fig.3-3. An MIL-Layer is inserted between the decoder block and the softmax layer during training with CAMs attention map as one input. The MIL-NN with attention map is trained iteratively. In the initial step of training, the neural network parameters $\boldsymbol{\theta}$ are initialized. The pretrained CAMs network is used to compute the scoremap of each image. The training data are selected from bags according to the initial pixel scoremap. Then the parameters of neural network are updated. For iteration $iter > 1$, either using the CAMs scoremap continuously, or using the pixel scoremap computed by FCNN with parameters $\boldsymbol{\theta}$ learned at $iter - 1$ to select data. And then retrain the neural network until it meet the stop criteria. The algorithm of MIL-NN with attention map is depicted by Alg.6.

Algorithm 6: MIL-NN with attention map

Data: $(B,t) = \{(B_1,t_1),...(B_i,t_i),...(B_I,t_I)\}$;
$B_i = \{x_{i1},...x_{in},...x_{iN}\}$
Result: $\boldsymbol{\theta}$
randomly initialize $\boldsymbol{\theta}$;
load pre-trained CAMs model;
compute the pixel scoremap using CAMs ;
repeat
  (optional) compute the pixel scoremap using FCNN ;
  update target selected from each positive bag based on the scoremap ;
  update network parameters $\boldsymbol{\theta}$;
until *meet stop criteria*;

3.2 Optimization

Many different optimization algorithms have been developed for the learning process of a neural network. Among this optimization algorithms, stochastic gradient descent (SGD) (Bottou, 2012) and its variants are the most widely used for learning neural network. However, compute the gradient with respect to the parameters of neural network individually is computationally expensive. Backpropagation (Rumelhart et al., 1988) has been developed to compute the gradients efficiently.

3.2.1 General Backpropagation in Neural Network

A neural network is made by stack layers with different functions together such as convolution layer, fully connected layer, pooling layer, normalization layer, activation layer, loss layer and so on. The structure of neural network make it feasible to compute gradient of a lower layer with gradient from the layer above it efficiently according to backpropagation.

Let's define a network work shown in Fig.3-4 having L layers, the input of the lth layer ($l \in L$) is $\mathbf{x}^l$, the output of the lth layer is $\mathbf{y}^l$, the output of the lth layer is the same as the input of the $(l+1)th$ layer as Eq.3-4.

$$\mathbf{x}_{l+1} = \mathbf{y}_l \tag{3-4}$$

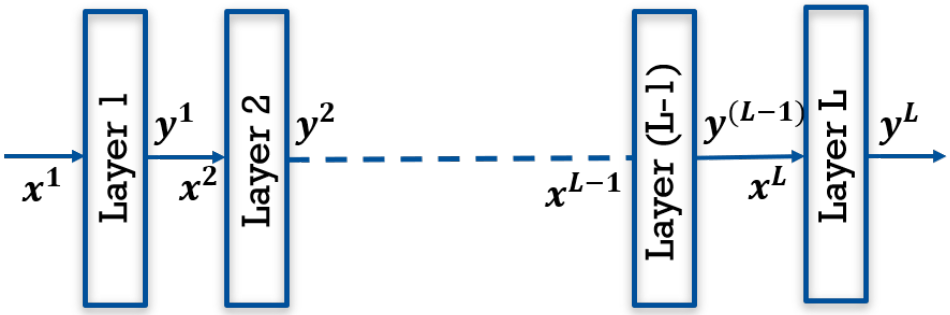


Figure 3-4. The illustration of neural network

The backpropagation computes the gradient using the chain rule. Backpropagation saves gradient computation time by store local gradient of each layer in memory. The local gradient of the lth layer is defined as Eq.3-5

$$\delta(\mathbf{y}^l) = \frac{\partial E}{\partial \mathbf{y}^l} \tag{3-5}$$

By applying the chain rule to Eq.3-5, it is transformed to a gradient related to the local gradient at the $(l+1)th$ layer as Eq.3-6. Having $\delta(\mathbf{y}^{l+1})$ saved in memory, it save the computation cost of $\delta(\mathbf{y}^l)$.

$$\begin{aligned} \delta(\mathbf{y}^l) &= \frac{\partial E}{\partial \mathbf{y}^l} \\ &= \frac{\partial E}{\partial \mathbf{y}^L}\frac{\mathbf{y}^L}{\partial \mathbf{y}^{L-1}}\cdots\frac{\partial \mathbf{y}^{l+1}}{\partial \mathbf{y}^l} \\ &= \delta(\mathbf{y}^{l+1})\frac{\partial \mathbf{y}^{l+1}}{\partial \mathbf{y}^l} \end{aligned} \tag{3-6}$$

The gradient of weights at the lth layer can be compute easily once the local gradient at the lth layer is known by Eq.3-7.

$$\begin{aligned} \delta(\mathbf{w}^l) &= \frac{\partial E}{\partial \mathbf{w}^l} \\ &= \frac{\partial E}{\partial \mathbf{y}^L}\frac{\mathbf{y}^L}{\partial \mathbf{y}^{L-1}}\cdots\frac{\partial \mathbf{y}^{l+1}}{\partial \mathbf{y}^l}\frac{\partial \mathbf{y}^l}{\partial \mathbf{w}^l} \\ &= \delta(\mathbf{y}^l)\frac{\partial \mathbf{y}^l}{\partial \mathbf{w}^l} \end{aligned} \tag{3-7}$$

3.2.2 MIL Layer Backward Derivation

We proposed a custom defined MIL layer to implement the data selection rules to satisfy the MIL condition. Several different selection rules have been used as mentioned in section 3.1. The MIL layer selects data from input of the lth layer $\mathbf{x}^l$ and saved it in a vector as output $\mathbf{y}^l$. Each elements in $\mathbf{x}^l$ and $\mathbf{y}^l$ is a K dimensional vector. The spatial position of the selected data is saved in vector $\mathbf{t}$. For a selected data $\mathbf{x}^l(1:K,m,n)$ saved at the ith position in $\mathbf{y}^l$ and $\mathbf{t}$, then the forward of MIL layer is Eq.3-8 and Eq.3-9.

$$\mathbf{y}^l(1:K,i) = \mathbf{x}^l(1:K,m,n) \tag{3-8}$$

$$\mathbf{t}(i,:) = (m,n) \tag{3-9}$$

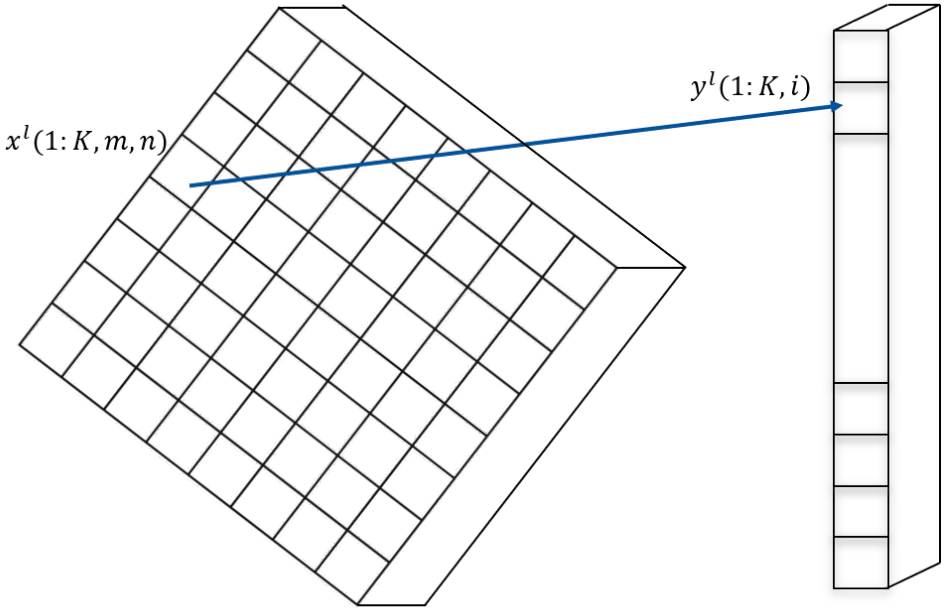


Figure 3-5. The illustration of MIL layer

The input of MIL layer $\mathbf{x}^l$ is the output of the $(l-1)th$ layer. The MIL layer connected the local gradient of the $(l-1)th$ layer $\delta(\mathbf{y}^{l-1})$ with the local gradient of the $(l)th$ layer $\delta(\mathbf{y}^l)$. The gradient of an element in $\delta(\mathbf{y}^l)$ is passed to the gradient of an element in $\delta(\mathbf{y}^{l-1})$ is there

is a line that link these two elements together as shown in Fig.3-5. It has been derived by Eq.3-10 and Eq.3-11.

$$\begin{aligned}\delta(y_{kmn}^{L-1}) &= \frac{\partial E}{\partial y_{kmn}^{L-1}} \\ &= \frac{\partial E}{\partial \mathbf{y}^{L}} \frac{\partial \mathbf{y}^{L}}{\partial y_{kmn}^{L-1}} \\ &= \sum_{i=1}^{n_1} \sum_{k=1}^{K} \frac{\partial E}{\partial y_{ki}^{L}} \frac{\partial y_{ki}^{L}}{\partial y_{kmn}^{L-1}} \\ &= \sum_{i=1}^{n_1} \frac{\partial E}{\partial y_{ki}^{L}} \frac{\partial y_{ki}^{L}}{\partial y_{kmn}^{L-1}}\end{aligned} \tag{3-10}$$

where

$$\frac{\partial y_{ki}^{L}}{\partial y_{kmn}^{L-1}} = \begin{cases} 1 & \text{if } t(i,:) = (m,n) \\ 0 & \text{if others} \end{cases} \tag{3-11}$$

CHAPTER 4
EXPERIMENTS AND RESULTS

In this chapter, we summary our experiments and results using MIL-NN to do root segmentation in minirhizotron images.

4.1 Data Description

Switchgrass (*Panicum virgatum* L.) is recognized as the most promising perennial grass for bioenergy production in the U.S. Nonetheless, despite being the subject of abundant and multidisciplinary research, there is a lack of understanding of the contributions of the switchgrass root system to the adaptability to different environments, biomass productivity, and substantial ecosystem services Schmer et al. (2008). The extensive root system of switchgrass is increasingly recognized to contribute to long-term carbon sequestration, high resource use efficiency, and high biomass productivity Comas et al. (2013); de Graaff et al. (2013). However, despite the essential biological function, information about switchgrass root system dynamics in space and time is limited, at least in part because of the challenges associated with access to the soil-root interface without disturbing the soil environment, which can impair subsequent measurements of the same plant Berhongaray et al. (2013).

Switchgrass root images were collected from minirhizotron access tubes that were installed in a 2-y old switchgrass field at the U.S. Department of Energy National Environmental Research Park at Fermilab in Batavia, IL, USA. Minirhizotron access tubes measuring 1.82 m in length were installed at 60° off the horizontal axis using an angled, guided hydraulic soil core sampler. The tubes were inserted to reach a maximum vertical depth of approximately 1.2 m, and foam caps were installed on the end protruding from the soil to insulate the tubes, block light, and protect them from UV damage. Images were collected by inserting a CI-602 in-situ root imager (CID BioScience, Camas, WA, USA) in to the minirhizotron tubes. This scanner was set to collect 360-degree images at 300 dpi in 28 cm depth increments.

The collected minirhizotron images are divided into subimages of size $710 \times 520 \times 3$ to reduce the burdon of memory requirment for computation. We have a training dataset having

6066 subimages paired with image level label and a test dataset having 316 subimages paired with pixel level label. The training dataset is divided into training data and validation data. The training data have 500 images with image label as root and 500 images with image label as soil. The rest data are validation data. Such training data, validatation data and test data are used in the following experiments.

4.2 Multiple Instance Neural Network with FCNN Output

The experiments in this section use Unet as the backbone structure of FCNN. The output of target class of FCNN is used as the pixel scoremap. Picking top 10 pixels from each positive image is used as the target selection rule. The results are evaluated by receiver operating characteristic (ROC) curve and Precision-Recall curve. The binaried segment results is evaluated by intersection over union (IoU).

4.2.1 Random Initialization

In this experiment, the parameters of MIL-NN is randomly initialized. The learning rate is set to $1e-3$. The MIL-NN network is trained using SGD with momentum. The weight for the momentum term is set to 0.8.

The experiment has been run multiple times, but it fails to segment roots from the image because the using target class output of FCNN with random initialization as pixel class scoremap could not locate at the targets in positive images. This method fails to pick target instances from positive bags to train network. Fig.4-1 shows the pixel class scoremap of this method at the initial step. Top 10 pixels having largest target class score are marked in red in the pixel class scoremap. At the initial step, the network is randomly initialized that the target class output of the network could not ensure the targets have a higher score in the target class output. The uncertainty of the pixel class scoremap results from the random initial. If the initial is bad, the pixel scoremap will get worse and the targets could not been picked from the positive image. It makes the neural network could not learn from targets. This could not be reversed as training going on because in the following iteration, the target pixel scoremap will keep on going bad.

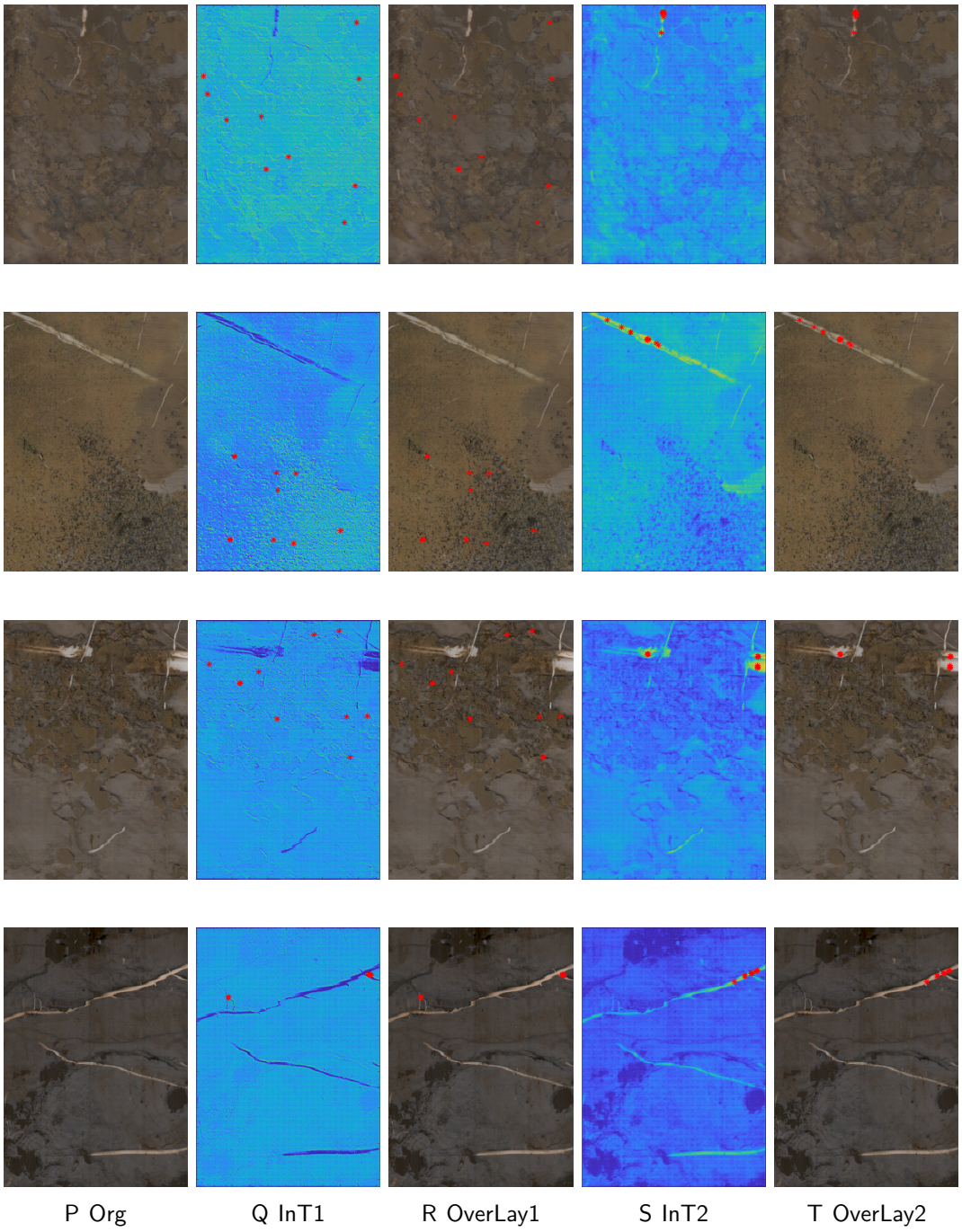


Figure 4-1. Random initialization. pick top 10

4.2.2 Transfer Learning

Transfer learning has been used to learn from small dataset (Papandreou et al., 2015). In this experiment, We use the weights of the pre-trained Unet to initialize our MIL-Unet. The

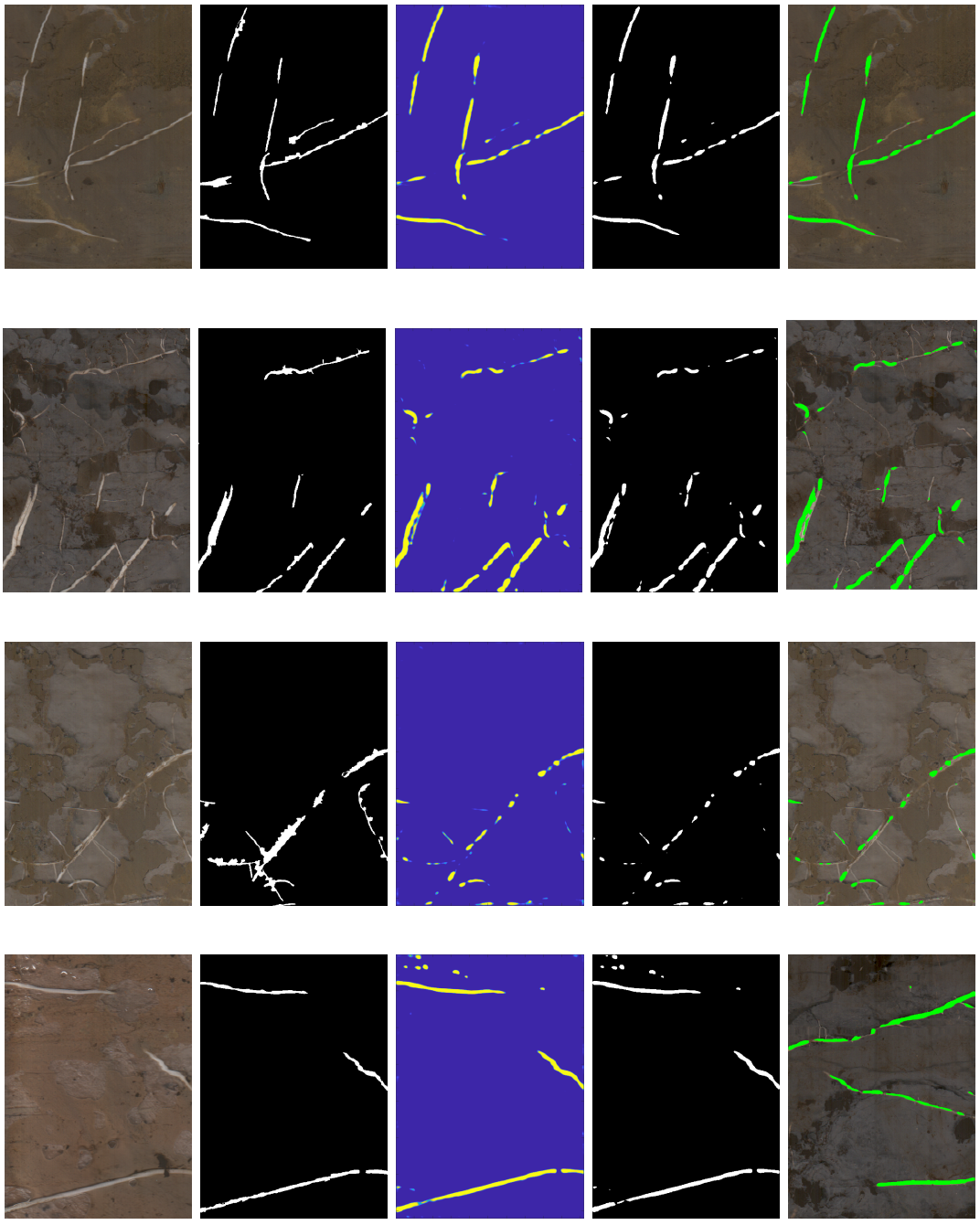


Figure 4-2. Transfer learning. pick top 10

Unet is trained with a fully annotated dataset contains 17550 peanut root images. The peanut minirhizotron dataset was collected in a field trial at the Plant Science Research and Education

Unit during the 2016 growing season. The learning rate is set to $1e-3$. The MIL-NN network is trained using SGD with momentum. The weight for the momentum term is set to 0.8.

Fig.4-2 visualized the results of this method. Compared with the results using random initialization, the MIL-Unet with pre-trained weights is able to extract general features of root objects. The target class output of MIL-Unet with pre-trained weights is able to give a high score to some of the root pixels. The pre-trained model with peanut dataset provides useful information to the new switchgrass dataset. The pixel class scoremap generated with pre-trained model could locate the root pixels in positive images and pick target pixels form positive bag at the initial step. In the following iteration, the results keep on to be refined. The results in Fig.4-2 shows the ability of transfer learning to detection new type of roots using MIL-Unet with pre-trained weights. Transfer learning provides a way to generate pixel class scoremap that could locate at some target pixels.

4.3 Multiple Instance Neural Network with attention map

Attention map is a widely used method to generate pixel class scoremap (Zhang et al., 2018b; Zhou et al., 2018; Wei et al., 2018). The experiments in this section use attention map as pixel class scoremap and Unet as the backbone structure. The model used for this experiments has two branches after the encoder part of Unet. One branch is used to compute the attention map. The last fully convolution layer of encoder is followed by a global pooling layer, a fully connected layer, and a softmax layer. This network we call it "attention network". The other branch followed by the decoder part of the Unet with multiply instance layer. This is called "MIL-Unet". The training process trains the two models in turn. First, the attention network is trained to classify images. The weights of the attention network is initialized randomly. The attention map is computed as the linear combination of the weights of the fully connected layer and the output of the last fully convolution layer of attention network. In the second stage, the MIL-Unet is trained to segment image. During the second stage, the weights of MIL-Unet which are the same as the weights of the attention network are initialized by the pre-trained weights of the attention network in the first step. Other weights of MIL-Unet is

randomly initialized. The results are evaluated by receiver operating characteristic (ROC) curve and Precision-Recall curve. The binaried segment results is evaluated by intersection over union (IoU).

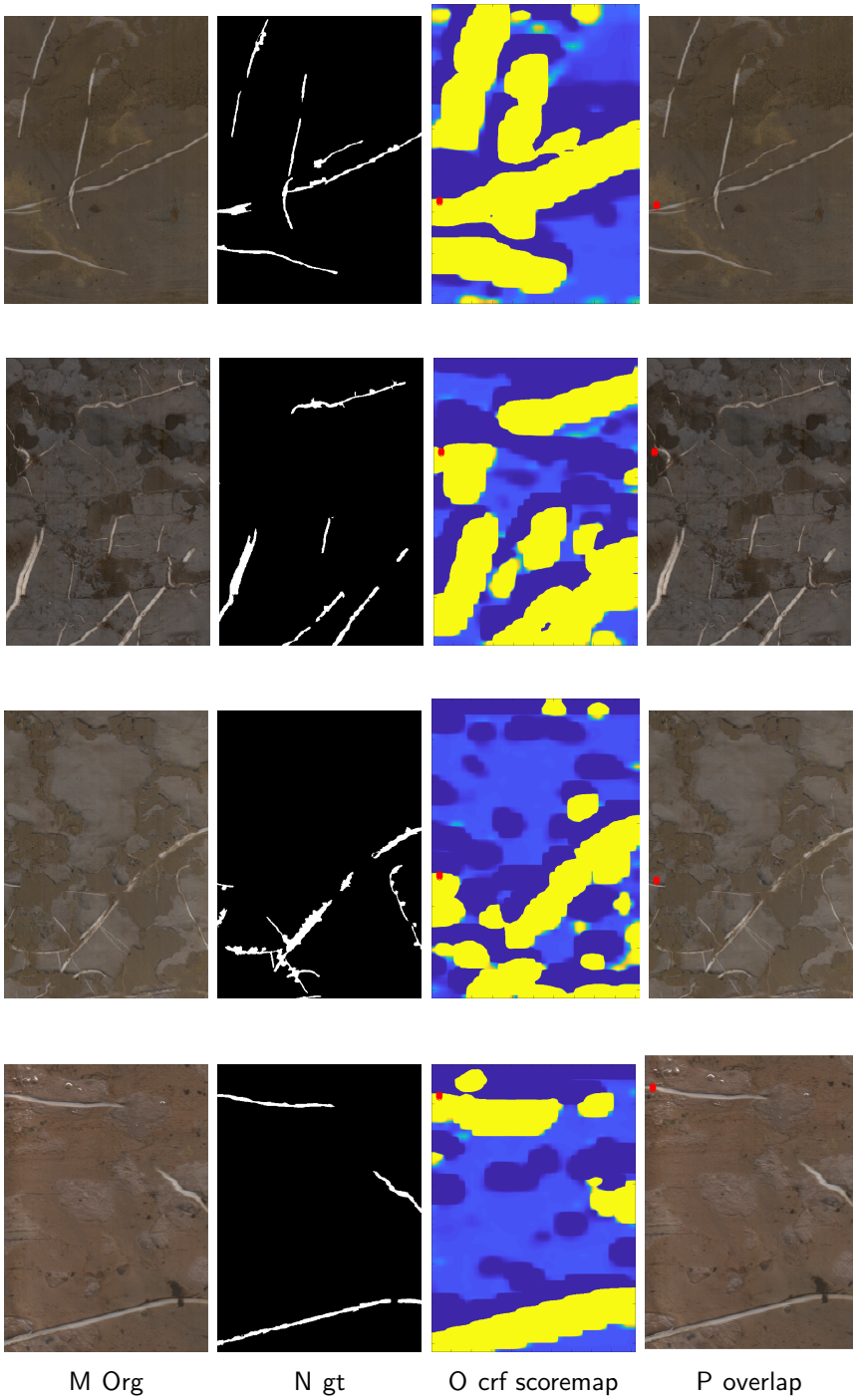


Figure 4-3. Attention map, pick top 10

4.3.1 Attention Map

In this experiment, picking top 10 pixels from attention map of each positive image is used as the target selection rule. This experiment fails to segment roots from the image because using attention map as pixel class scoremap could not locate at the targets in positive images. Fig.4-3 illustrated the attention map at the initial step while training the MIL-Unet in the second stage. At the initial step of the second stage, the top 10 pixels having the largest score value on pixel class scoremap can not locate on the roots in image as the red dots shown in Fig.4-3. These picked pixels are close to the root pixels but not on the root pixels. It is common that the highlight part of attention map shifts a little bit from the objects in images. Because of the shift of the attention map, it is hard to pick target accurately from the attention map. With a bad pixel class scoremap at initial step, the pixel class scoremap will get worse during training since the model is not learning from correct targets. Finally, the model will fail to separate roots from soil.

4.3.2 Attention Map with CRF

CRF is a widely used way to smooth segmentation results. In this experiment, We use CRF to process the segmentation results as the arg max of attention map over all classes in Sec.4.3.1. CRF is a graph based algorithm. An fully connected graph is build for each image that each pixel is a node of the graph, there is a edge between every pair of nodes. The weight of each edge is computed using Eq.2-10. The CRF is optimized by the method proposed by Krähenbühl and Koltun (2011).

The refined segmentation is shown in Fig.4-4. The refined segmentation of the bottom 2 images are good, but the refined segmentation of the top 2 images are bad. The CRF works as a model to smooth the result that if roots take up most part of the image, it tend to classify soil pixels around roots to be root. For example, in the scoremap of the first image of Fig.4-4, the number of pixels with root label overtake the number of pixels with soil label. Therefore, after processed by CRF, the soil pixels are wrongly classified to be roots as shown in the last two images in the first row of Fig.4-4. The inference results of CRF highly depend on the

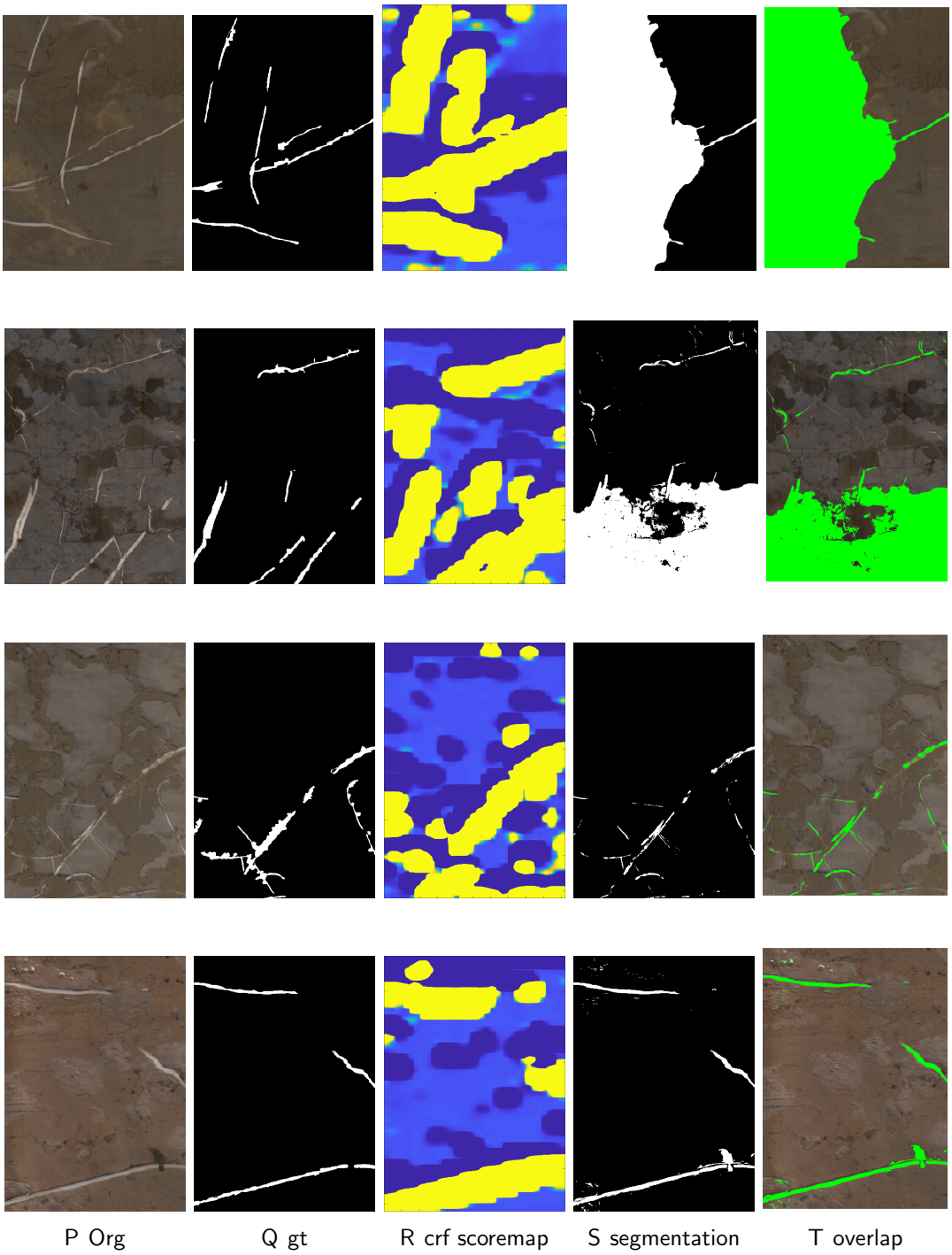


Figure 4-4. Threshold attention map, refine threshold attention map with CRF

initial segmentation. Hence, it is important to get an attention map with accurate location of objects.

4.4 Multiple Instance Neural Network with MI-ACE

In this section, the Unet is used as the backbone structure of FCNN. MI-ACE scoremap is used as the pixel scoremap druing the whole training process and only at the inital step respectively. The MI-ACE is trained in the same way as Yu et al. (2019). The experiments in Sec.4.4.1 and Sec.4.4.2 first compare the effect of adaptively changing targets picking from positive images. The experiments in Sec.4.4.1 and Sec.4.4.3 first compare the effect of increasing targets picking from positive images. The results are evaluated by receiver operating characteristic (ROC) curve and Precision-Recall curve. The binaried segment results is evaluated by intersection over union (IoU).

4.4.1 MIACE Scoremap as with Top n

In this experiment, picking top 10 pixels from MI-ACE scoremap of each positive image is used as the target selection rule during the whole training process. The MI-ACE and the MIL-Unet are trained separately hence the targets picked from the scoremap are not changed during the training process. The learning rate is set to $1e-3$. The MIL-NN network is trained using SGD with momentum. The weight for the momentum term is set to 0.8.

The results of this method is visualized in Fig.4-5. This method is able to separate roots from soil. The results shown in Fig.4-5 find most of the roots in images. Using MI-ACE scoremap as pixel class scoremap is an effective way to picking targets from positive images.

4.4.2 MI-ACE Scoremap as Initial Pixel Scoremap with Top n

In the experiments in Sec.4.4.1, the targets picked from positive images could not be adaptively changed during training which made the training data highly depend on the initialization by the MI-ACE results. In this experiment, only at the initial step that the training data is picked from the MI-ACE scoremap. After the initial step, targets are picking form the target class output of Unet. The targets picked from the pixel class scoremap will changed during training. Picking top 10 pixels from pixel class scoremap of each positive image is used as the target selection rule. The learning rate is set to $1e-3$. The MIL-NN network is trained using SGD with momentum. The weight for the momentum term is set to 0.8.

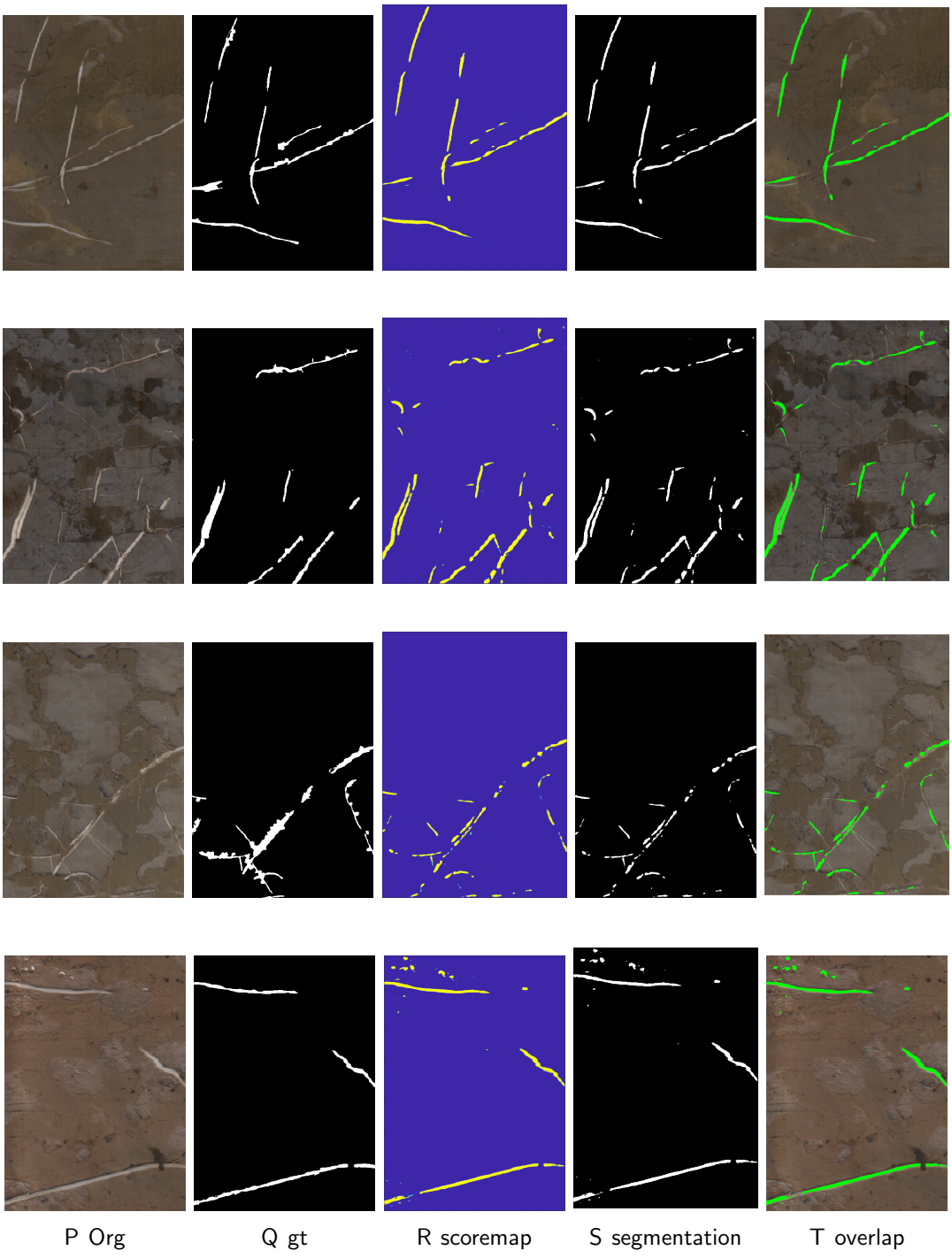


Figure 4-5. Use MIACE as segmentation mask, pick top 10

The results of this method is visualized in Fig.4-6. This method is also able to separate roots from soil. Most of the roots in images are found as shown in Fig.4-6.

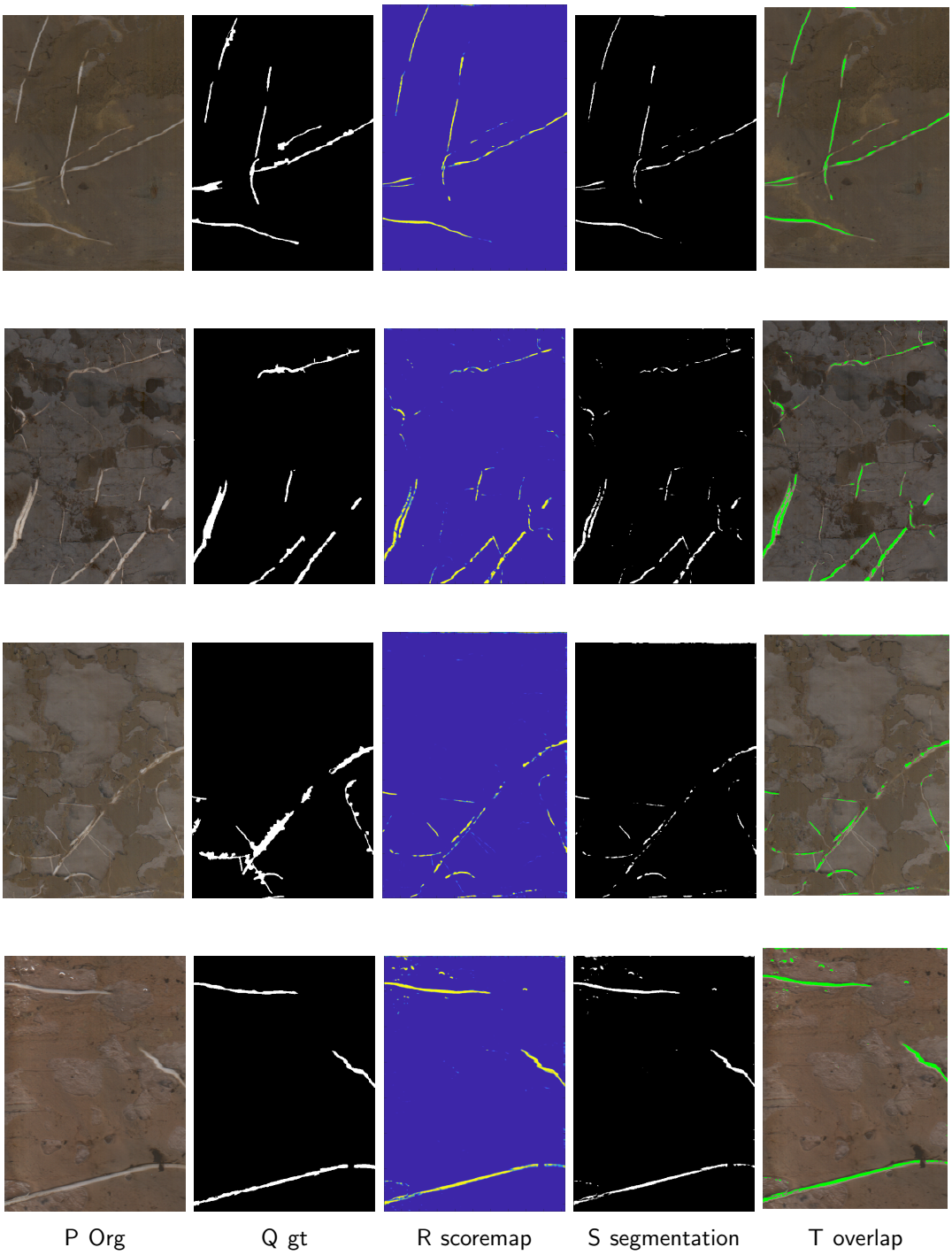


Figure 4-6. Initial use MIACE as segmentation mask, pick top 10, then pick top 10 from target class prediction

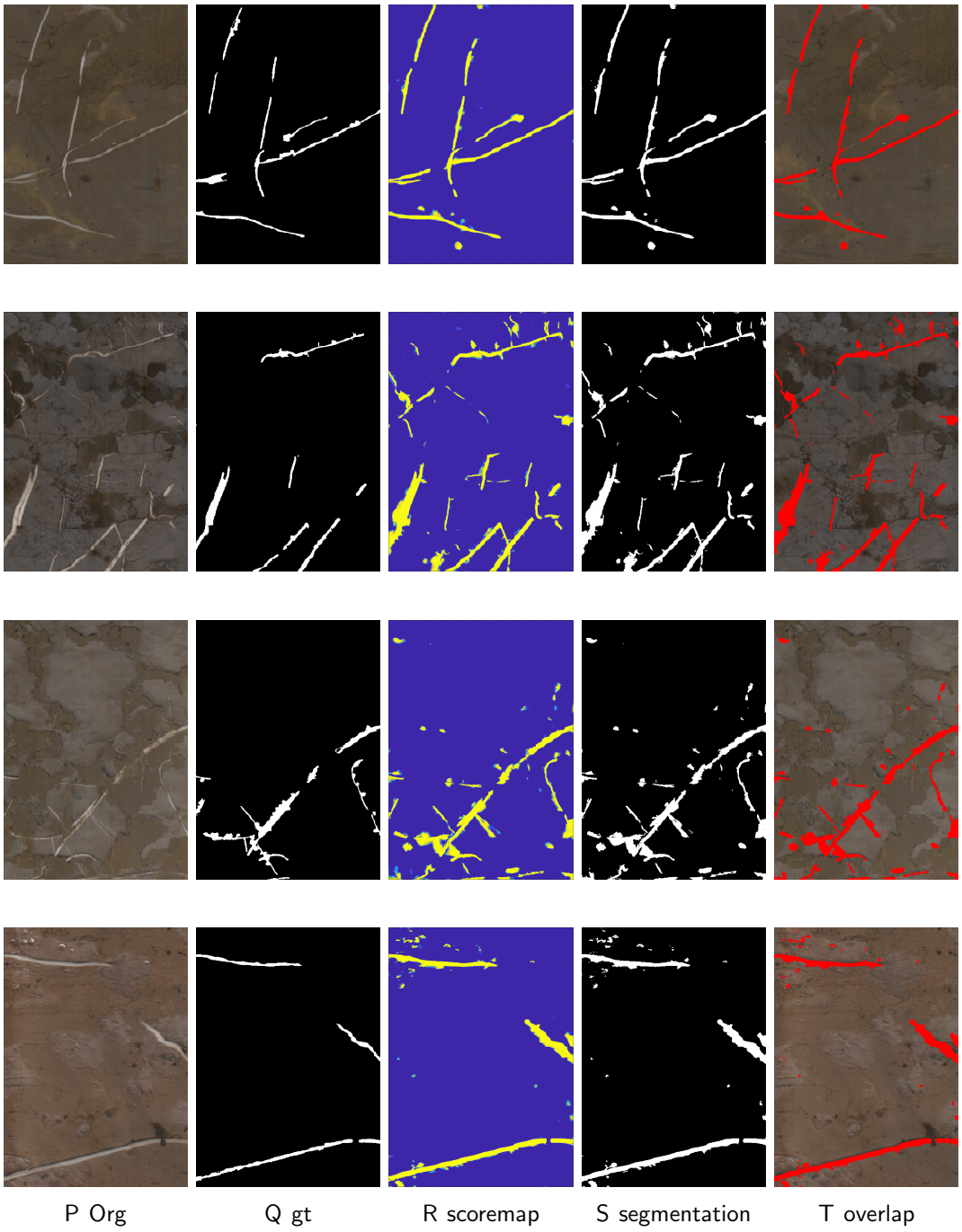


Figure 4-7. Use threshold MIACE as segmentation mask

4.4.3 MI-ACE Scoremap with Threshold

Compared with the experiments in Sec.4.4.1 that only top 10 pixels are picked from positive images as targets, in this experiment, we threshold the MI-ACE scoremap at $FRP =$

3%. The thresholded MI-ACE scoremap is used as the target selection rule during the whole training process. The threshold is large enough that there are more than 10 pixels having target label. The learning rate is set to $1e-3$. The MIL-NN network is trained using SGD with momentum. The weight for the momentum term is set to 0.8.

The results of this method is visualized in Fig.4-7. This method is also able to separate roots from soil. Most of the roots in images are found as shown in Fig.4-7. Compared with results in Fig.4-5, the root segments in Fig.4-6 are bigger. There are less small root segments in Fig.4-6. Hence, picking more targets from pixel class scoremap is a good way to make the target samples in the dataset more complete and diverse.

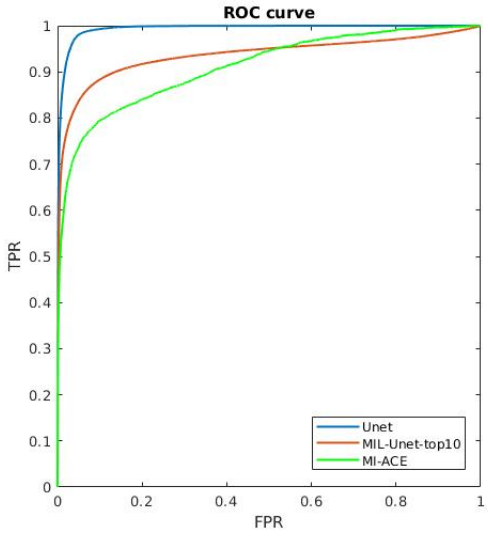


Figure 4-8. Compare results of Unet, MIL-Unet, and MI-ACE.

4.5 Compare Unet, MIL-Unet, and MI-ACE

In this section, we compare the segmentation results of Unet, MIL-Unet, and MI-ACE. These three different methods are trained and tested with same dataset contains 377 training images and 43 test images. The Unet is trained with full annotation. The MIL-Unet and MI-ACE is trained with image level annotation. The Unet is trained in the same way as Xu et al. (2019). The MIACE is trained in the same way as Yu et al. (2019). The MIL-Unet is

trained in the same way as Sec.4.4.2. The results are compared in Fig.4-8 by ROC curves and illustrated in Fig.4-9.

From Fig.4-8, it is clearly that MIL-Unet is better than our previous results MI-ACE, however it still can not compare with Unet. As illustrated in Fig.4-9, the results of MIL-Unet have more small root segments than Unet. The MIL-Unet tends to overfit the roots.

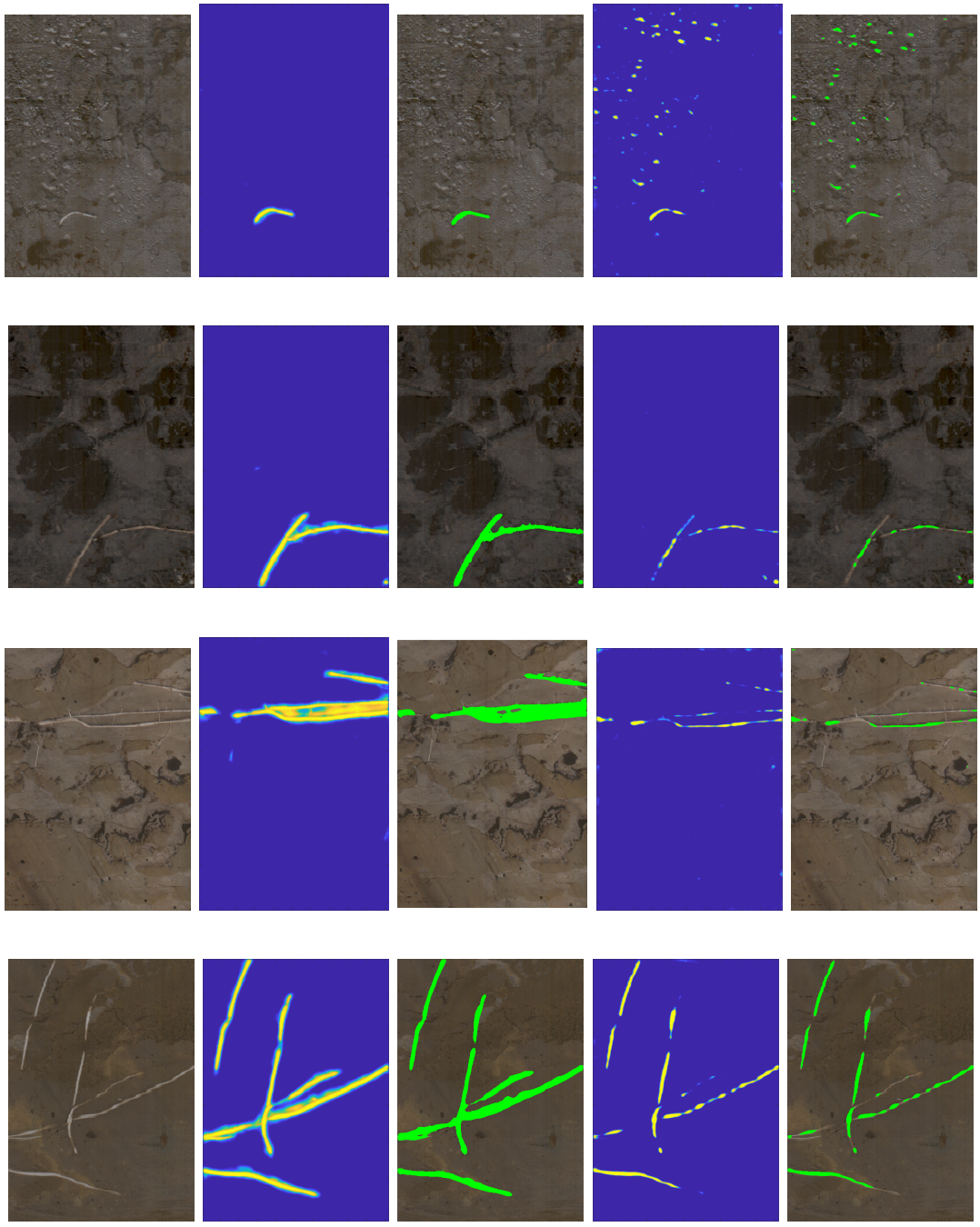


Figure 4-9. initial use MIACE as segmentation mask, pick top 10, then pick top 10 from target class prediction

CHAPTER 5
FUTURE WORK

In this chapter, we will discuss about the experiments that we plan to do for the research.

5.1 Experiments on how to indicate where the targets are in the positive image

First, we will describe about what we plan to do on how to indicate where the targets are in the positive image. We will do experiments to study how well the using of MI-ACE scoremap, attention map, and the output of softmax layer of neural network to find targets respectively.

5.1.1 Experiment always using MI-ACE scoremap to indicate where the targets are in the positive image

In this experiment, we will use setting threshold on MI-ACE scoremap to select targets from positive image for MIL-NN. The experiment has two stages. In the first stage, we will train MI-ACE model to detect targets in data. We will do five-folder cross validation multiple times to train a robust model. In the second stage, we train the MIL-NN using Unet as backbone structure to segment target object from image. We will use ROC curve, PR curve and IoU to evaluate the segmentation result. The experiment will run on the minirhizotron Image dataset, the NEON dataset, and the PASCAL VOC 2012 dataset.

5.1.2 Experiment using MI-ACE scoremap to indicate where the targets are in the positive image at the initial step of training MIL-NN

In this experiment, we will use setting threshold on MI-ACE scoremap to select targets from positive image at the initial step of training MIL-NN. After the initial step, we will take the arg max over the output of the softmax layer of MIL-NN to select targets from positive image for training MIL-NN. The experiment has two stages. In the first stage, we will train MI-ACE model to detect targets in data. We will do five-folder cross validation multiple times to train a robust model. In the second stage, we train the MIL-NN using Unet as backbone structure to segment target object from image. We will use ROC curve, PR curve and IoU to evaluate the segmentation result. The experiment will run on the minirhizotron Image dataset, the NEON dataset, and the PASCAL VOC 2012 dataset.

5.1.3 Experiment using the pre-trained model to compute attention map to indicate where the targets are in the positive image

In this experiment, we will use the pre-trained neural network to compute attention map and take arg max over the attention map to indicate where the targets are in the positive image. If there are no pixels being assigned a target class label in positive image, the one has the largest target class score will be labeled as target. The experiment has two stages. In the first stage, We will train a image classification neural network to compute CAMs (Zhou et al., 2016) as attention map. We will try different image classification neural network including GoogLeNet, VGG16, AlexNet and ResNet. In the second stage, we train the MIL-NN using Unet as backbone structure to segment target object from image. We will use ROC curve, PR curve and IoU to evaluate the segmentation result. The experiment will run on the minirhizotron Image dataset, the NEON dataset, and the PASCAL VOC 2012 dataset.

5.1.4 Experiment using the learnable attention map to indicate where the targets are in the positive image

In this experiment, we will use the learnable attention map and take arg max over the attention map to indicate where the targets are in the positive image. If there are no pixels being assigned a target class label in positive image, the one has the largest target class score will be labeled as target. The MIL-NN will be used in this experiment, including using FCN, SegNet and Unet as backbone structure. The MIL-NN has a encoder part and a decoder part. The encoder part will be used as the backbone structure to compute CAMs. The experiment has two stages. In the first stage, we will train the encoder part to classify image. In the second stage, we will initial the encoder with the pre-trained weights and initial the rest weights randomly. Then train both the encoder part and the decoder part together. We will use ROC curve, PR curve and IoU to evaluate the segmentation result. The experiment will run on the minirhizotron Image dataset, the NEON dataset, and the PASCAL VOC 2012 dataset.

5.2 Experiments on how to select targets completely and accurately

The experiments in Sec.5.1 fix the way to select targets from positive image by either taking arg max or by setting a threshold. The experiments in this part, we will vary ways to select targets from positive image to study how to select targets completely and accurately from positive image under MIL constraint. The experiment in Sec.5.1 will be redone with different ways to select targets.

5.2.1 Experiment on picking a fixed number of pixels as targets

In this experiment, we will do experiment in Sec.5.1 but use picking a fixed number of pixels to select targets. We will increase the number of picked pixels to study how the number and accurate in label of targets effect the segmentation results.

5.2.2 Experiment on setting threshold to pick targets

In this experiment, we will do experiment in Sec.5.1 but use setting threshold to select targets. We will decrease the threshold to study how the number and accurate in label of targets effect the segmentation results.

5.3 Experiments on adding constraints to features of targets and non-targets

The experiments in this part will study the features of targets and non-targets (Fogel et al., 2019). Both the attention map and the segmentation results depend on the features of data. We will add different constraints to the features extracted from neural network. The experiments in Sec.5.1 will be redone with different ways to add constraint to deep-based feature. The neural network learns hierarchical features, we will also study the difference of adding feature constraint to different layers.

5.3.1 Experiment on adding constraint to features based on bag label

In this experiment, we will do experiment in Sec.5.1 but adding constraint to features based on bag label. The features of data in negative bag should be similar to each other in feature space, the selected targets in positive bag should be dissimilar to the features of data in negative bag.

5.3.2 Experiment on adding constraint to features based on spatial and low level features

In this experiment, we will do experiment in Sec.5.1 but adding constraint to features based on spatial and low level features. The features of data close in space and have similar low level feature (color) should be similar, otherwise, should be dissimilar.

REFERENCES

Achanta, Radhakrishna, Shaji, Appu, Smith, Kevin, Lucchi, Aurelien, Fua, Pascal, and Süsstrunk, Sabine. "SLIC superpixels compared to state-of-the-art superpixel methods." *IEEE transactions on pattern analysis and machine intelligence* 34 (2012).11: 2274–2282.

Andrews, Stuart, Tsochantaridis, Ioannis, and Hofmann, Thomas. "Support vector machines for multiple-instance learning." *Advances in neural information processing systems*. 2003, 577–584.

Badrinarayanan, Vijay, Kendall, Alex, and Cipolla, Roberto. "Segnet: A deep convolutional encoder-decoder architecture for image segmentation." *IEEE transactions on pattern analysis and machine intelligence* 39 (2017).12: 2481–2495.

Bates, GH. "A device for the observation of root growth in the soil." *Nature* 139 (1937).3527: 966.

Batra, Dhruv, Kowdle, Adarsh, Parikh, Devi, Luo, Jiebo, and Chen, Tsuhan. "icoseg: Interactive co-segmentation with intelligent scribble guidance." *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 2010, 3169–3176.

Bearman, Amy, Russakovsky, Olga, Ferrari, Vittorio, and Fei-Fei, Li. "What's the point: Semantic segmentation with point supervision." *European conference on computer vision*. Springer, 2016, 549–565.

Bell, Sean, Upchurch, Paul, Snavely, Noah, and Bala, Kavita. "Material recognition in the wild with the materials in context database." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, 3479–3487.

Berhongaray, Gonzalo, Janssens, IA, King, JS, and Ceulemans, R. "Fine root biomass and turnover of two fast-growing poplar genotypes in a short-rotation coppice culture." *Plant and soil* 373 (2013).1-2: 269–283.

Bishop, Christopher M. *Pattern recognition and machine learning*. springer, 2006.

Bottou, Léon. "Stochastic gradient descent tricks." *Neural networks: Tricks of the trade*. Springer, 2012. 421–436.

Boykov, Yuri and Kolmogorov, Vladimir. "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision." *IEEE Transactions on Pattern Analysis & Machine Intelligence* (2004).9: 1124–1137.

Boykov, Yuri Y and Jolly, M-P. "Interactive graph cuts for optimal boundary & region segmentation of objects in ND images." *Proceedings eighth IEEE international conference on computer vision. ICCV 2001*. vol. 1. IEEE, 2001, 105–112.

Chang, Chih-Chung and Lin, Chih-Jen. "LIBSVM: A library for support vector machines." *ACM Transactions on Intelligent Systems and Technology* 2 (2011): 27:1–27:27. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Comas, Louise, Becker, Steven, Cruz, Von Mark V, Byrne, Patrick F, and Dierig, David A. "Root traits contributing to plant productivity under drought." *Frontiers in plant science* 4 (2013): 442.

Cortes, Corinna and Vapnik, Vladimir. "Support-vector networks." *Machine learning* 20 (1995).3: 273–297.

Dai, Jifeng, He, Kaiming, and Sun, Jian. "Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation." *Proceedings of the IEEE International Conference on Computer Vision*. 2015, 1635–1643.

de Graaff, Marie-Anne, Six, Johan, Jastrow, Julie D, Schadt, Christopher W, and Wullschleger, Stan D. "Variation in root architecture among switchgrass cultivars impacts root decomposition rates." *Soil Biology and Biochemistry* 58 (2013): 198–206.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results." http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html, ????

Fogel, Sharon, Averbuch-Elor, Hadar, Cohen-Or, Daniel, and Goldberger, Jacob. "Clustering-driven deep embedding with pairwise constraints." *IEEE computer graphics and applications* 39 (2019).4: 16–27.

group, ECODSE. "ECODSE competition training set." 2017. Funders: "Gordon and Betty Moore Foundation's Data-Driven Discovery Initiative" through Grant GBMF4563 "National Institute of Standards and Technology (NIST). Data Science for Multimodal Plant Identification Task.".

URL https://doi.org/10.5281/zenodo.867646

Hastie, Trevor, Tibshirani, Robert, Friedman, Jerome, and Franklin, James. "The elements of statistical learning: data mining, inference and prediction." *The Mathematical Intelligencer* 27 (2005).2: 83–85.

He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing, and Sun, Jian. "Deep residual learning for image recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, 770–778.

Hong, Seunghoon, Oh, Junhyuk, Lee, Honglak, and Han, Bohyung. "Learning transferrable knowledge for semantic segmentation with deep convolutional neural network." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, 3204–3212.

Hong, Seunghoon, Yeo, Donghun, Kwak, Suha, Lee, Honglak, and Han, Bohyung. "Weakly supervised semantic segmentation using web-crawled videos." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, 7322–7330.

Jetley, Saumya, Lord, Nicholas A, Lee, Namhoon, and Torr, Philip HS. "Learn to pay attention." *arXiv preprint arXiv:1804.02391* (2018).

Johnson, Mark G, Tingey, David T, Phillips, Donald L, and Storm, Marjorie J. "Advancing fine root research with minirhizotrons." *Environmental and Experimental Botany* 45 (2001).3: 263–289.

Kolesnikov, Alexander and Lampert, Christoph H. "Seed, expand and constrain: Three principles for weakly-supervised image segmentation." *European Conference on Computer Vision*. Springer, 2016, 695–711.

Koller, Daphne and Friedman, Nir. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

Krähenbühl, Philipp and Koltun, Vladlen. "Efficient inference in fully connected crfs with gaussian edge potentials." *Advances in neural information processing systems*. 2011, 109–117.

Krizhevsky, Alex, Sutskever, Ilya, and Hinton, Geoffrey E. "Imagenet classification with deep convolutional neural networks." *Advances in neural information processing systems*. 2012, 1097–1105.

Leistner, Christian, Saffari, Amir, and Bischof, Horst. "MIForests: Multiple-instance learning with randomized trees." *European Conference on Computer Vision*. Springer, 2010, 29–42.

Li, Yin, Sun, Jian, Tang, Chi-Keung, and Shum, Heung-Yeung. "Lazy snapping." *ACM Transactions on Graphics (ToG)* 23 (2004).3: 303–308.

Li, Zhuwen, Chen, Qifeng, and Koltun, Vladlen. "Interactive image segmentation with latent diversity." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, 577–585.

Lin, Di, Dai, Jifeng, Jia, Jiaya, He, Kaiming, and Sun, Jian. "Scribblesup: Scribble-supervised convolutional networks for semantic segmentation." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, 3159–3167.

Lin, Tsung-Yi, Maire, Michael, Belongie, Serge, Hays, James, Perona, Pietro, Ramanan, Deva, Dollár, Piotr, and Zitnick, C Lawrence. "Microsoft coco: Common objects in context." *European conference on computer vision*. Springer, 2014, 740–755.

Long, Jonathan, Shelhamer, Evan, and Darrell, Trevor. "Fully convolutional networks for semantic segmentation." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, 3431–3440.

Newell, Alejandro, Yang, Kaiyu, and Deng, Jia. "Stacked hourglass networks for human pose estimation." *European conference on computer vision*. Springer, 2016, 483–499.

Oquab, Maxime, Bottou, Léon, Laptev, Ivan, and Sivic, Josef. "Is object localization for free?-weakly-supervised learning with convolutional neural networks." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, 685–694.

Papandreou, George, Chen, Liang-Chieh, Murphy, Kevin P, and Yuille, Alan L. "Weakly-and semi-supervised learning of a deep convolutional network for semantic image segmentation." *Proceedings of the IEEE international conference on computer vision*. 2015, 1742–1750.

Pinheiro, Pedro O and Collobert, Ronan. "From image-level to pixel-level labeling with convolutional networks." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, 1713–1721.

Pound, Michael P, Atkinson, Jonathan A, Wells, Darren M, Pridmore, Tony P, and French, Andrew P. "Deep learning for multi-task plant phenotyping." *Proceedings of the IEEE International Conference on Computer Vision*. 2017, 2055–2063.

RAHMANZADEH, BAHRAM H and Shojaedini, SV. "Novel Automated Method for Minirhizotron Image Analysis: Root Detection using Curvelet Transform." (2016).

Ronneberger, Olaf, Fischer, Philipp, and Brox, Thomas. "U-net: Convolutional networks for biomedical image segmentation." *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, 234–241.

Rother, Carsten, Kolmogorov, Vladimir, and Blake, Andrew. "Grabcut: Interactive foreground extraction using iterated graph cuts." *ACM transactions on graphics (TOG)*. vol. 23. ACM, 2004, 309–314.

Roy, Anirban and Todorovic, Sinisa. "Combining bottom-up, top-down, and smoothness cues for weakly supervised image segmentation." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, 3529–3538.

Rumelhart, David E, Hinton, Geoffrey E, Williams, Ronald J, et al. "Learning representations by back-propagating errors." *Cognitive modeling* 5 (1988).3: 1.

Schmer, Marty R, Vogel, Kenneth P, Mitchell, Robert B, and Perrin, Richard K. "Net energy of cellulosic ethanol from switchgrass." *Proceedings of the National Academy of Sciences* 105 (2008).2: 464–469.

Selvaraju, Ramprasaath R, Cogswell, Michael, Das, Abhishek, Vedantam, Ramakrishna, Parikh, Devi, and Batra, Dhruv. "Grad-cam: Visual explanations from deep networks via gradient-based localization." *Proceedings of the IEEE International Conference on Computer Vision*. 2017, 618–626.

Shojaedini, S and Heidari, M. "A new method for root detection in minirhizotron images: Hypothesis testing based on entropy-based geometric level set decision." *Ilj* 1 (2013).1: 1.

Simonyan, Karen, Vedaldi, Andrea, and Zisserman, Andrew. "Deep inside convolutional networks: Visualising image classification models and saliency maps." *arXiv preprint arXiv:1312.6034* (2013).

Simonyan, Karen and Zisserman, Andrew. "Very deep convolutional networks for large-scale image recognition." *arXiv preprint arXiv:1409.1556* (2014).

Springenberg, Jost Tobias, Dosovitskiy, Alexey, Brox, Thomas, and Riedmiller, Martin. "Striving for simplicity: The all convolutional net." *arXiv preprint arXiv:1412.6806* (2014).

Szegedy, Christian, Liu, Wei, Jia, Yangqing, Sermanet, Pierre, Reed, Scott, Anguelov, Dragomir, Erhan, Dumitru, Vanhoucke, Vincent, and Rabinovich, Andrew. "Going deeper with convolutions." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, 1–9.

Tang, Meng, Djelouah, Abdelaziz, Perazzi, Federico, Boykov, Yuri, and Schroers, Christopher. "Normalized cut loss for weakly-supervised cnn segmentation." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, 1818–1827.

Taylor, HM, Upchurch, DR, Brown, JM, and Rogers, HH. "Some methods of root investigations." *Developments in Agricultural and Managed Forest Ecology*. vol. 24. Elsevier, 1991. 553–564.

Tompson, Jonathan J, Jain, Arjun, LeCun, Yann, and Bregler, Christoph. "Joint training of a convolutional network and a graphical model for human pose estimation." *Advances in neural information processing systems*. 2014, 1799–1807.

Waddington, J. "Observation of plant roots in situ." *Canadian Journal of Botany* 49 (1971).10: 1850–1852.

Wang, Tao, Rostamza, Mina, Song, Zhihang, Wang, Liangju, McNickle, G, Iyer-Pascuzzi, Anjali S, Qiu, Zhengjun, and Jin, Jian. "SegRoot: A high throughput segmentation method for root image analysis." *Computers and Electronics in Agriculture* 162 (2019): 845–854.

Wei, Yunchao, Xiao, Huaxin, Shi, Honghui, Jie, Zequn, Feng, Jiashi, and Huang, Thomas S. "Revisiting dilated convolution: A simple approach for weakly-and semi-supervised semantic segmentation." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, 7268–7277.

Xu, Weihuang, Yu, Guohao, Zare, Alina, Zurweller, Brendan, Rowland, Diane, Reyes-Cabrera, Joel, Fritschi, Felix B., Matamala, Roser, and Juenger, Thomas E. "Overcoming Small Minirhizotron Datasets Using Transfer Learning." *CoRR* abs/1903.09344 (2019).

URL http://arxiv.org/abs/1903.09344

Yasrab, Robail, Atkinson, Jonathan A, Wells, Darren M, French, Andrew P, Pridmore, Tony P, and Pound, Michael P. "RootNav 2.0: Deep Learning for Automatic Navigation of Complex Plant Root Architectures." *BioRxiv* (2019): 709147.

Yu, Guohao, Zare, Alina, Sheng, Hudanyun, Matamala, Roser, Reyes-Cabrera, Joel, Frischi, Felix B., and Juenger, Thomas E. "Root Identification in Minirhizotron Imagery with Multiple Instance Learning." 2019.

Zare, Alina, Jiao, Changzhe, and Glenn, Taylor. "Discriminative multiple instance hyperspectral target characterization." *IEEE transactions on pattern analysis and machine intelligence* 40 (2017).10: 2342–2354.

Zeiler, Matthew D and Fergus, Rob. "Visualizing and understanding convolutional networks." *European conference on computer vision*. Springer, 2014, 818–833.

Zeng, Guang, Birchfield, Stanley T, and Wells, Christina E. "Detecting and measuring fine roots in minirhizotron images using matched filtering and local entropy thresholding." *Machine Vision and Applications* 17 (2006).4: 265–278.

———. "Rapid automated detection of roots in minirhizotron images." *Machine Vision and Applications* 21 (2010).3: 309–317.

Zhang, Jianfeng, Zhang, Liezhuo, Teng, Yuankai, Zhang, Xiaoping, Wang, Song, and Ju, Lili. "Interactive Binary Image Segmentation with Edge Preservation." *arXiv preprint arXiv:1809.03334* (2018a).

Zhang, Xiaolin, Wei, Yunchao, Kang, Guoliang, Yang, Yi, and Huang, Thomas. "Self-produced guidance for weakly-supervised object localization." *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018b, 597–613.

Zhou, Bolei, Khosla, Aditya, Lapedriza, Agata, Oliva, Aude, and Torralba, Antonio. "Learning deep features for discriminative localization." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, 2921–2929.

Zhou, Yanzhao, Zhu, Yi, Ye, Qixiang, Qiu, Qiang, and Jiao, Jianbin. "Weakly supervised instance segmentation using class peak response." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, 3791–3800.