Fall 2022

CNN-Based Dendrite Core Detection From Microscopic Images of Directionally Solidified NI-Base Alloys

Xiaoguang Li

CNN-based Dendrite Core Detection from Microscopic Images of Directionally Solidified Ni-base Alloys

by

Xiaoguang Li

Bachelor of Science
Zhengzhou University 2013

Submitted in Partial Fulfillment of the Requirements

for the Degree of Master of Science in

Computer Science and Engineering

College of Engineering and Computing

University of South Carolina

2022

Accepted by:

Song Wang, Director of Thesis

Yan Tong, Reader

Ramtin Mohammadizand, Reader

Jeff Simmons, Reader

Tracey L. Weldon, Vice Provost and Dean of the Graduate School

Abstract

Dendrite core is the center point of the dendrite. The information of dendrite core is very helpful for material scientists to analyze the properties of materials. Therefore, detecting the dendrite core is a very important task in the material science field. Meanwhile, because of some special properties of the dendrites, this task is also very challenging. Different from the typical detection problems in the computer vision field, detecting the dendrite core aims to detect a single point location instead of the bounding-box. As a result, the existing regressing bounding-box based detection methods can not work well on this task because the calculated center point location based on the upper-left and lower-right corners of the bounding-box is usually not precise. In this work, we formulate the dendrite core detection problem as a segmentation task and proposed a novel detection method to detect the dendrite core directly. Our whole pipeline contains three steps: Easy Sample Detection (ESD), Hard Sample Detection(HSD), and Hard Sample Refinement (HSR). Specifically, ESD and HSD focus on the easy samples and hard samples of dendrite cores respectively. Both of them employ the same Central Point Detection Network (CPDN) but do not share parameters. To make HSD only focus on the feature of hard samples of dendrite cores, we destroy the structure of the easy samples of dendrites which are detected by ESD and force HSD to learn the feature of hard samples. HSR is a binary classifier which is used to filter out the false positive prediction of HSD. We evaluate our method on the dendrite dataset. Our method outperforms the state-of-the-art baselines on three metrics, i.e., Recall, Precision, and F-score.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Dendrites are tree-like structures of crystals which are formed during some materials' solidification [34]. In the procedure of solidification, the morphology of dendrite can has a huge change which leads to a significant effect on the properties of materials [30] [6] [3] [17]. For example, [33] claims that learning the growth mechanism of metal dendrites in the electrochemical procedure can extend battery life significantly and [22] mentions the study of dendrites structure is helpful to predict solidification defects. The dendrite core is the center point of the dendrite. Several dendrite samples are shown in Fig. 1.1 where red points denote the dendrite cores. The information of dendrite core is very helpful for the material scientists to analyze the properties of materials. Therefore, detecting the dendrite core is a very important task in the material science field. Meanwhile, because of some special properties of the dendrites, this task is also very challenging. First, the microscopic images of the dendrites can be very blurred as shown in Fig. 1.2 (a)(b), and it can be very difficult to locate the dendrite cores from the microscopic image. Second, besides the blurry, there are many different noises in the microscopic images such as the dark and bright points in Fig. 1.2 (c) and the black spots in Fig. 1.2 (d). These noises will increase the difficulty of locating the dendrites. Third, some of the dendrites in the microscopic image are incomplete such as the dendrites in the yellow boxes in Fig. 1.2 (e)(f). The appearance of these incomplete dendrites is totally different from that of the complete ones and only the materials scientists with expertise can recognize them. As a result, locating the dendrites and annotating the dendrite cores are very time-consuming

and expensive.

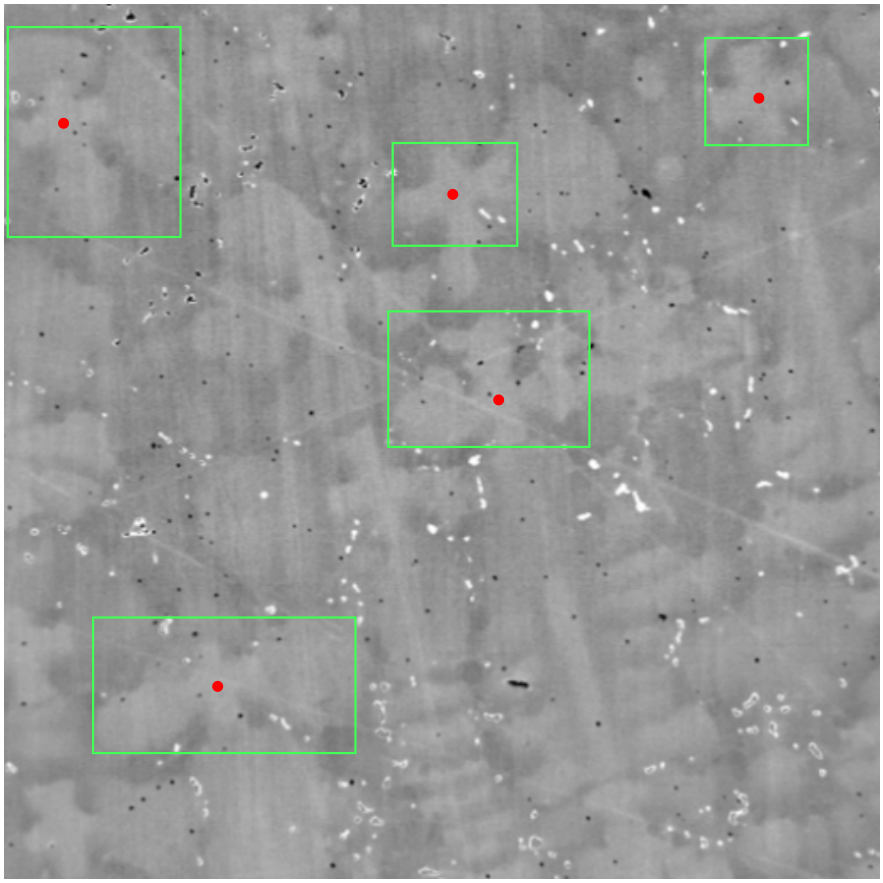

Figure 1.1 An illustration of dendrite samples. The red points denote the dendrite cores.

As Deep Neural Network (DNN) has achieved significant success in the natural image process, material scientists are exploiting DNN to solve the problems in their field. For example, [41] proposes to use DNN to reconstruct 3D objects by detecting 2D objects on each cross-section, and [12] proposes to use DNN to perform X-ray CT segmentation task. However, there are few kinds of research focusing on dendrite core detection. Different from the typical detection problems in the computer vision field, detecting the dendrite core aims to detect a single point location instead of the bounding-box. Therefore, the existing regressing bounding-box based detection methods such as [18], [26], [20], and [27] can not work well on this task. From Fig. 1.3

we can find that the regressing bounding-box based methods try to tightly cover the dendrite instead of focusing on the dendrite core. Besides, the appearance of the dendrites varies a lot, therefore the calculated center point location based on the upper-left and lower-right corners of the bounding box is usually inaccurate when used as the estimate of the dendrite core. As for the key points detection algorithms, they also can not work well on this task because of the complex properties of the dendrites mentioned above. For example, the key point detection method in [23], which is designed for detecting human key points, can not be used to detect the dendrite cores, even after increasing network complexity, due to the blurry or serious incompleteness of the dendrites.

Inspired by [23], In this work, we formulate the dendrite core detection problem as a segmentation task and propose a novel detection method to detect the dendrite core directly. Our whole pipeline contains three steps: Easy Sample Detection (ESD), Hard Sample Detection(HSD), and Hard Sample Refinement (HSR). Specifically, ESD and HSD focus on the easy samples and hard samples of dendrite cores respectively. Both of them employ the same Central Point Detection Network (CPDN) but not sharing parameters. To make HSD only focus on the feature of hard samples of dendrite cores, we destroy the structure of the easy samples of dendrites which are detected by ESD and force HSD to learn the feature of hard samples. HSR is a binary classifier which is used to filter out the false positive prediction of HSD.

Our main contribution is twofold.

❶ We propose a novel detection method to detect the dendrite cores directly.

❷ We conduct a series of experiments for exploiting the optimal crop size and crop intensity to destroy the structure of the easy samples of dendrites.

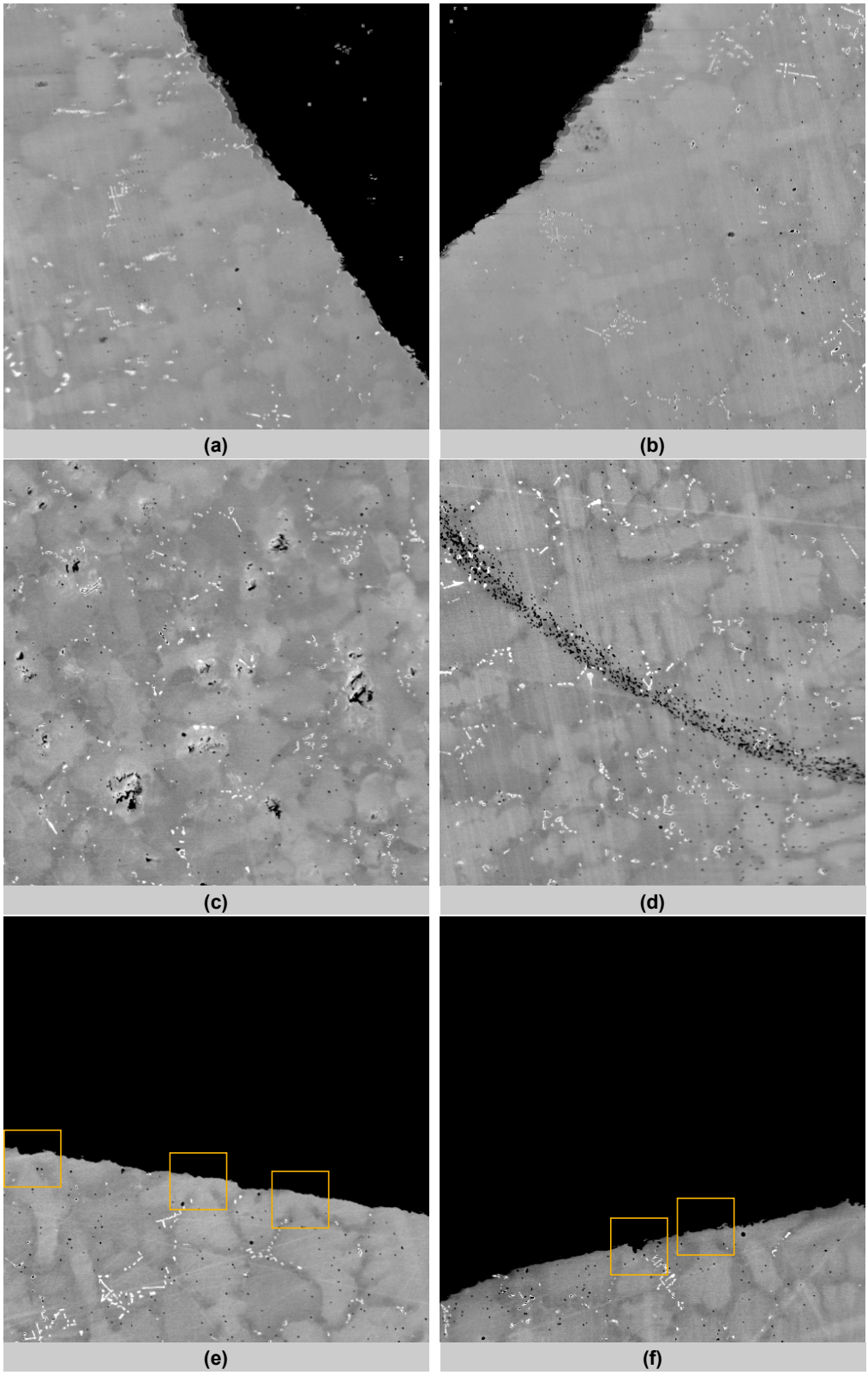

Figure 1.2 Examples of microscopic images with blurry and noise in (a)(b) and (c)(d) respectively and the incomplete dendrites in (e)(f).

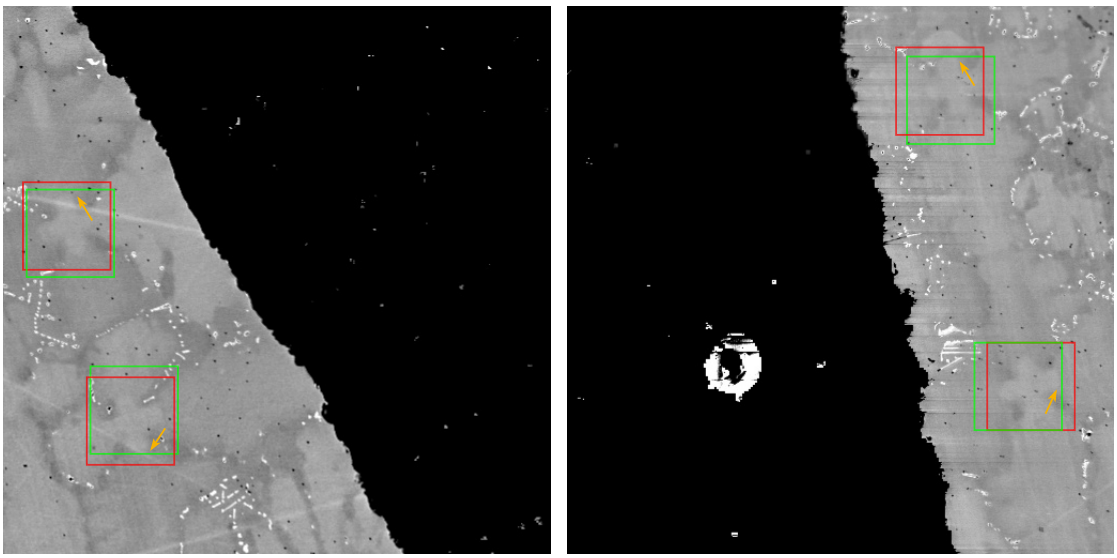

Figure 1.3 Visualization results of the regressing bounding-box based method. The red boxes denote the ground truth and the green boxes denote the predictions. The yellow arrows denote the edges that the predicted boxes try to fit.

Chapter 2

Background

2.1 Convolutional Neural Network (CNN)

The Convolutional Neural Network (CNN) is similar to the fully connected neural network. Both CNN and the fully connected neural network are comprised of neurons and can optimize their parameters through the learning process. However, compared with the fully connected neural network, CNN has several salient advantages. First, CNN is much more computationally efficient because of parameter sharing. Second, the sparsity of connection in CNN makes each output of the convolution layer only depend on a small number of inputs which efficiently prevent the overfitting. Third, CNN can keep the position information of the 2D images. Since the AlexNet [13] is proposed in 2012, CNN has almost become the most popular deep learning architecture in the computer vision field. Usually, the whole CNN contains several components such as convolution operation, pooling, activation function, stride, and padding. We will elaborate on each of these components in the following paragraphs.

2.1.1 Convolution

The convolution operation is the most important component in CNN. Specifically, the convolution operation is a mathematical operation between the input vector and the corresponding filter or kernel. As shown in Fig. 2.1, it performs the convolution operation at the red circle location of the input vector. It makes element-wise multiplication and sums the result. In this case, the kernel size is 3*3, therefore, it will take a 3*3 area at each location of the input vector. In practice, in each convolu-

tion layer, it chooses many different kernels to do the convolution operation because each of these kernels can extract different feature information from the input vector. Based on the previous study, the shallow convolution layers of CNN can extract low-level features from the input vector such as edges, and the deeper layers can extract possible objects such as faces or even more complex features.

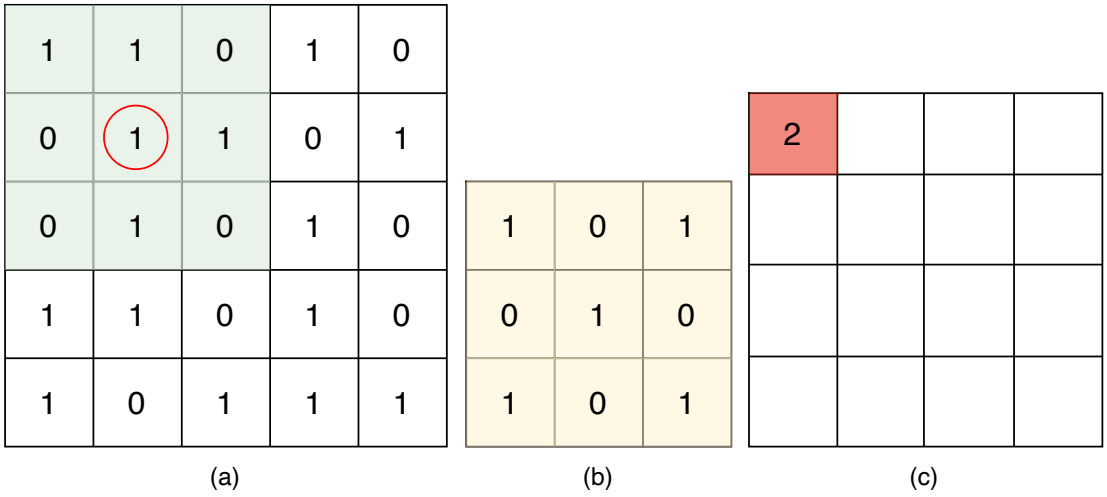

Figure 2.1 (a) denotes the input vector, (b) denotes the filter or kernel, and (c) denotes the result of convolution operation at the red circle location of the input vector.

2.1.2 Activation function, stride, and padding

The convolution operation is linear. In order to make CNN more powerful, it passes the result of the convolution operation through a non-linear function such as ReLU in Fig. 2.2. The ReLU function drops the values smaller than zero. The stride specifies the distance it moves the filter or kernel at each step. We can choose a big stride if we want to have less overlap between adjacent convolution operations. In order to keep the output feature having the same dimension as the input feature, we also pad zeros around the input vector.

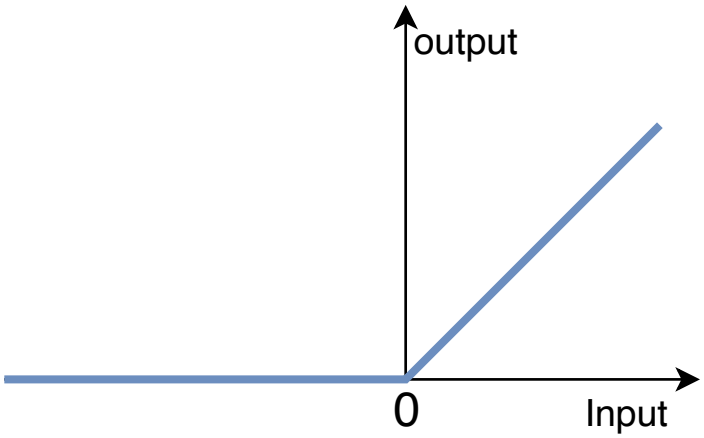

Figure 2.2 ReLU function.

2.1.3 Pooling

The pooling operation reduces the feature's dimension which can save the training time and prevent overfitting. There are no trainable parameters in the pooling operation. For the max-pooling in Fig. 2.3, it keeps the maximum value around the 2*2 window. In this case, the max-pooling downsamples the feature dimension to 2*2 from 4*4.

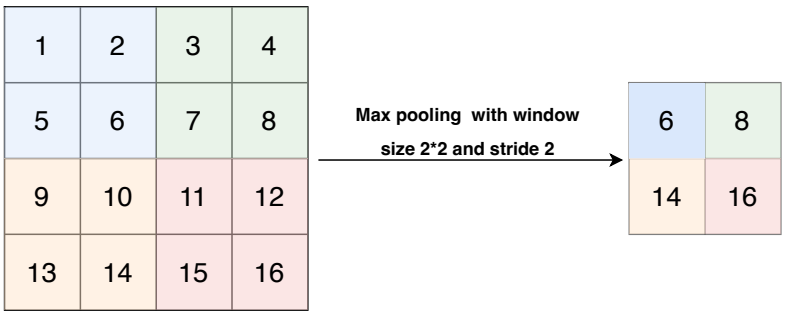

Figure 2.3 Max pooling.

2.2 Residual learning framework

It is difficult to train a deep neural network when it contains a lot of layers. As the depth of the deep neural network increases, the accuracy degrades quickly. In order to solve the degradation problem, [9] propose the "deep residual learning framework". $\mathbf{x}$ and $\mathbf{F}(\mathbf{x})$ in Fig. 2.4 denote the input feature and the residual respectively. $\mathbf{H}(\mathbf{x})$ in Eq. (2.1)

$$\mathbf{F}(\mathbf{x}) = \mathbf{H}(\mathbf{x}) - \mathbf{x} \tag{2.1}$$

denotes the desired underlying mapping. Instead of making the layers fit $\mathbf{H}(\mathbf{x})$ directly, the residual learning framework makes the layers fit $\mathbf{H}(\mathbf{x}) - \mathbf{x}$. In this case, if the input feature $\mathbf{x}$ contains all the useful information the residual learning framework will push $\mathbf{F}(\mathbf{x})$ to zero. By this mechanism, the deep residual network can be optimized easily and trained end-to-end.

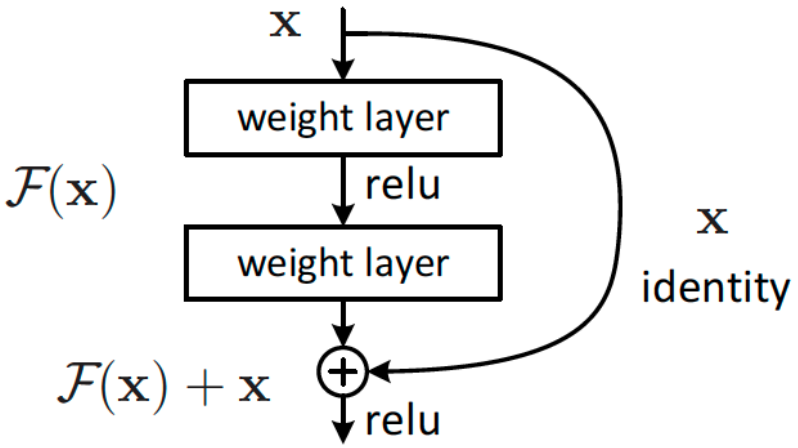

Figure 2.4 Residual learning block. The weight layer denotes the convolution layer. The figure comes from [9].

2.3 Introduction of Stacked-Hourglass Network

The Stacked-Hourglass Network[23] is designed for predicting human key points. The single Hourglass module in Fig. 2.5 (a) is composed of many residual modules. The

whole network in Fig. 2.5 (b) consists of multiple Hourglass modules. These stacked Hourglass modules repeatedly downsample and upsample the features and catch the human pose information across various scales. During the training step, it predicts multiple heatmaps and applies the L2 loss on each of them. For the inference, it only keeps the heatmap which is produced by the last Hourglass module.

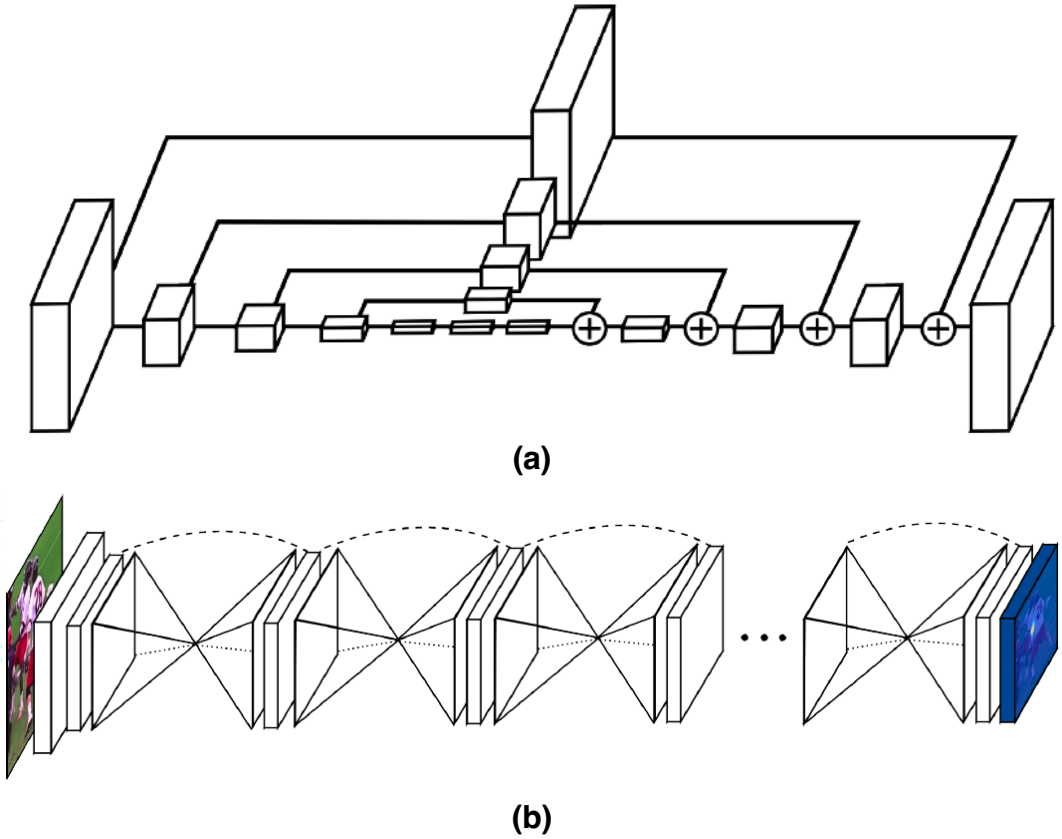

Figure 2.5 The figures come from [23]. (a) is the single Hourglass module and (b) is the whole Stacked-Hourglass Network.

Chapter 3

Related Work

3.1 CNN based detection on natural images

In recent years, deep convolutional neural network(CNN) based methods have been working with promising performance on natural images processing such as image classification [14][16][36][15], instance segmentation [39][11][5][37], and object detection [42][31][10][26][32] tasks. For the existing CNN based object detection methods, most of them can be classified into two groups: The regressing bounding-box based methods [18][27][35][25] that regress the upper-left and lower-right corners of the bounding-boxes corresponding to the objects and the segmentation based methods [23][40] that classify the pixels in the input image into different groups.

Regressing bounding-box based methods. The regressing bounding-box based methods detect the objects by regressing the upper-left and lower-right coordinates of the bounding-boxes corresponding to the objects. For instance, as discussed in [8], it first generates the potential bounding-boxes based on the region proposals and then makes the classification and refines the bounding-boxes. To enhance the speed of detection, [7] and [27] make an improvement on the proposal stage by computing the proposals with a deep convolutional neural network. In order to further increase the detection speed, [26] and [18] propose the one-stage detection which spatially separates bounding-boxes and associates with class probabilities. Different from [26], [18] applied multiple aspect ratios and scales over different feature maps. Unlike [26][18], [35][25] regress the location of the bounding-boxes in the image level

directly instead of the feature level. Regarding the flexibility of detected bounding-boxes, [20] suggests generating inclined proposals with angle information. Then it adjusts the angle of predicted bounding-boxes to make the detected result more accurate. However, these methods mentioned above focus more on the edges of the objects' bounding boxes instead of the center point locations. As a result, the calculated center point locations based on the upper-left and lower-right corners of the bounding-boxes are not precise.

Segmentation based methods. The segmentation based methods classify the pixels in the input image into different groups and each group corresponds to one category. Different from the regressing bounding-box based methods, the segmentation based methods can detect the center point locations of the objects directly. For example, for the human key points detection task, the pixels in the input image are classified into background, mouth, nose, and so on. In [23], it concatenates multiple UNet[28] structures together that repeatedly downsample and upsample the features and catch the human key points information at different scales. To reduce the model parameter and speed up the detection, [40] pay attention to the context information and propose a method called Cascaded Context Mixer which can integrate the spatial level and channel level information together and refine them step by step.

3.2 CNN based detection and segmentation on the material science images

Detection. In order to improve the detection speed and accuracy, in recent years, many works are trying to solve the detection problem in the material science field by using the deep learning based method. For example, [38] proposed to use Faster R-CNN based method to detect the internal defects from the CT scanning image of the metal three-dimensional lattice structure. To prevent overfitting to a small training dataset, it reduces the number of convolution layers and pooling layers. In

[21], a region based approach is implemented by CNN to detect and localize the anomalies in the scanning electron microscope images of nanofibrous materials. It also uses CNN-extracted features to evaluate the degree of difference between the anomaly samples and anomaly-free samples. In [29], CNN and a Deep Feed Forward Network are combined to detect the anomaly of Carbon Fiber Reinforced Polymer thermograms. It can detect the anomaly in thermograms in real-time without any manual intervention. To speed up the detection and reduce the model parameters, [2] proposed a method called WDD-Net which applies depthwise separable convolution and global pooling to detect the wafer structural defects. Besides the detection of anomalies and defects, in [41], CNN is applied to detect the fiber cross-section in the 2D microscopic images, and the detected 2D fiber cross-sections are then used to reconstruct the 3D fiber structure.

Segmentation. Besides detection, CNN is also widely used for the segmentation of various material science images. For example, [19] proposed to train a CNN model based on DeepLab to perform the segmentation of AL-LA alloy microscopic images. It takes advantage of the local symmetric information and applies symmetric rectification to enhance the segmentation accuracy. In [4], FibeR-CNN is proposed to analyze the fiber images based on the segmentation result, which employs R-CNN as the backbone and inserts additional convolution layers to help the prediction. To avoid labeling a large number of ground truth samples manually, [1] designed a semi-supervised learning method that applies UNet as the backbone to solve the segmentation problem of aluminum alloy metallographic image. This semi-supervised method only requires labeling a small number of images and can get better segmentation results than the traditional segmentation methods. To further improve the segmentation performance, [24] proposed the use of the Generative Adversarial Network (GAN) based method to perform the segmentation on carbon steel microstructure images. To reduce the number of training images, it also proposed a

CNN based framework to generate the training data.

Detecting the dendrite core is a very challenging and important task that can help material scientists, there are very few prior researchs that are exactly focused on this problem. Most of the above CNN based detection methods for the materials science images treat all the detection targets equally without distinguishing the easy and hard samples. Besides, without considering the unique and complex properties of dendrites, general CNN based detection methods for the natural images may not work well for this specific task. In this work, we formulate the dendrite core detection problem as a segmentation task and propose a novel detection method to detect the easy samples and hard samples of dendrite cores separately.

Chapter 4

Methodology

In this work, we formulate the dendrite core detection problem as a segmentation task. The whole pipeline contains three steps: Easy Sample Detection (ESD), Hard Sample Detection(HSD), and Hard Sample Refinement (HSR). Specifically, ESD and HSD focus on the easy samples and hard samples of dendrite cores respectively. Both of them employ the same Central Point Detection Network (CPDN), which is shown in Fig. 4.1 (b), but not sharing parameters. To make it clear, we denote the CPDN $\mathbf{D}_1$ in ESD and $\mathbf{D}_2$ in HSD. HSR is used to improve the Precision of HSD. The whole pipeline is shown in Fig. 4.1 (a). We will elaborate on ESD in Chapter 4.1, HSD in Chapter 4.2, CPDN in Chapter 4.3, and HSR in Chapter 4.4 respectively.

4.1 Easy Sample Detection (ESD)

In this step, it only focuses on detecting the easy samples of dendrite cores from the input image. We denote the input image $\mathbf{I}_1$ and ground truth heatmap $\mathbf{H}_1$ in ESD where $\mathbf{I}_1 \in \mathbb{R}^{H \times W \times 3}$ and $\mathbf{H}_1 \in \mathbb{R}^{H \times W \times 1}$. The ground truth heatmap $\mathbf{H}_1$ is binarized that one denotes the dendrite core and zero denotes the background. $\mathbf{D}_1$ will produce a heatmap $\hat{\mathbf{H}}_1$ to denote the detected location of each dendrite core based on the input image $\mathbf{I}_1$

$$\hat{\mathbf{H}}_1 = \mathbf{D}_1(\mathbf{I}_1) \tag{4.1}$$

where $\hat{\mathbf{H}}_1$ has the same dimension as $\mathbf{H}_1$. We aim to make ESD detect the easy samples of dendrite cores with a very high confidence to improve the Precision. The

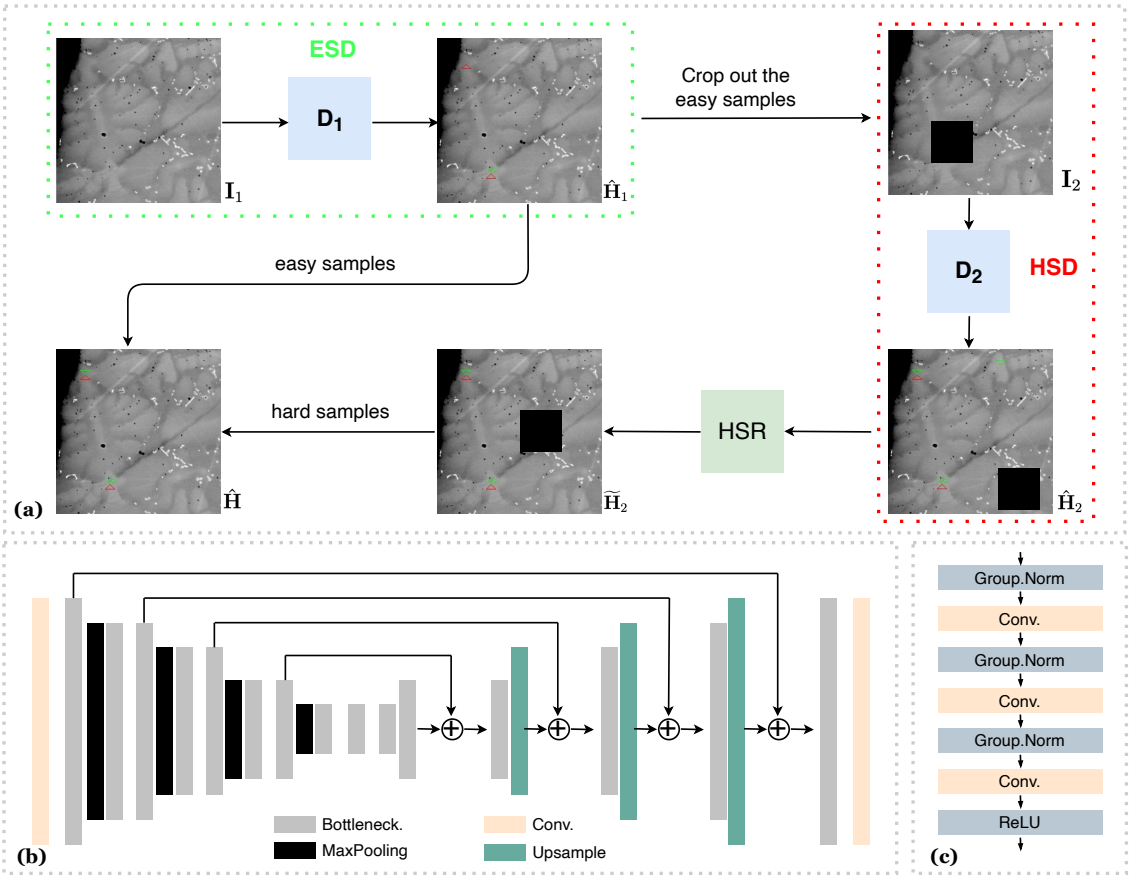

Figure 4.1 (a) denotes our whole pipeline, (b) denotes the Central Point Detection Network (CPDN), and (c) denotes the Bottleneck Block. Both $\mathbf{D}_1$ and $\mathbf{D}_2$ employ CPDN but not sharing parameters.

dendrite cores that can not be detected will be processed in HSD. Therefore, we set up a relatively higher confidence threshold α to binarize the predicted heatmap $\hat{\mathbf{H}}_1$. If the value of the location (i, j) in $\hat{\mathbf{H}}_1$ greater than α, we set $\hat{\mathbf{H}}_1(i, j)$ to one, otherwise set the value to zero, i.e.,

$$\hat{\mathbf{H}}_1(i,j) = \begin{cases} 1, & \text{if } \hat{\mathbf{H}}_1(i,j) > \alpha, \\ 0, & \text{otherwise.} \end{cases} \tag{4.2}$$

4.2 Hard Samples Detection (HSD)

In this step, we only focus on the hard samples of dendrite cores which can not be detected in ESD such as the blurred or incomplete samples of dendrites. We denote

the input image $\mathbf{I}_2$ and ground truth heatmap $\mathbf{H}_2$ in HSD where $\mathbf{I}_2$ and $\mathbf{H}_2$ have same dimension as $\mathbf{I}_1$ and $\mathbf{H}_1$. To make $\mathbf{D}_2$ only focus on the feature of hard samples of dendrite cores, $\mathbf{I}_2$ shall only contain the hard samples of dendrites. We can construct $\mathbf{I}_2$ by cropping the structure of easy samples of dendrites which are detected by $\mathbf{D}_1$ from $\mathbf{I}_1$, i.e.,

$$\mathbf{I}_1(i-s:i+s, j-s:j+s) = 0, \qquad \text{if } \hat{\mathbf{H}}_1(i,j) = 1 \tag{4.3}$$

where s decide the crop size which will be discussed in Chapter 5.4. The ground truth $\mathbf{H}_2$ is generated by:

$$\mathbf{H}_2(i,j) = \begin{cases} 0, & \text{if } \hat{\mathbf{H}}_1(i,j)=\mathbf{1}, \\ \mathbf{H}_1(i,j), & \text{otherwise.} \end{cases} \tag{4.4}$$

$\mathbf{D}_2$ will produce another heatmap $\hat{\mathbf{H}}_2$ which denotes the detected locations of hard samples of dendrite cores based on $\mathbf{I}_2$, i.e.,

$$\hat{\mathbf{H}}_2 = \mathbf{D}_2(\mathbf{I}_2) \tag{4.5}$$

In HSD, it usually detects the hard samples of dendrite cores with a relatively lower confidence score. In order to improve the Recall, we setup a relative lower threshold β to binarize the predicted heatmap $\hat{\mathbf{H}}_2$

$$\hat{\mathbf{H}}_2(i,j) = \begin{cases} 1, & \text{if } \hat{\mathbf{H}}_2(i,j)>\beta, \\ 0, & \text{otherwise.} \end{cases} \tag{4.6}$$

4.3 Central Point detection network (CPDN)

Both $\mathbf{D}_1$ and $\mathbf{D}_2$ employ the same CPDN but not sharing parameters. CPDN is implemented based on the architecture of U-Net. The encoder of CPDN consists of a convolution layer followed by several bottleneck layers and max-pooling layers. The bottleneck block consists of three group normalization layers and three convolution layers with kernel size 1*1, 3*3, and 1*1 respectively, followed by a ReLU operation

as shown in Fig. 4.1 (c). The decoder of CPDN consists of several upsample layers and bottleneck layers, followed by a convolution layer with kernel size 1*1. $\mathbf{D}_1$ and $\mathbf{D}_2$ are trained separately and optimized with L_2 loss.

4.4 Hard Samples Refinement (HSR)

In order to improve the Recall in HSD, we set up a relative lower confidence threshold β which leads to predict a lot of false positive samples of dendrite cores. To solve this problem, we add HSR after HSD. The HSR is a binary classifier which is gotten by fine tuning the ResNet-50 pretrained model on the dendrite dataset. We denote the input of HSR $\mathbf{I}_3$ which is a small patch with size 80*80 and can be gotten by

$$\mathbf{I}_3^{(i,j)} = \mathbf{I}_2(i-40:i+40, j-40:j+40), \qquad \text{if } \hat{\mathbf{H}}_2(i,j)=1 \tag{4.7}$$

HSR will output one for the input $\mathbf{I}_3^{(i,j)}$ if $\hat{\mathbf{H}}_2(i,j)$ is a true positive prediction, otherwise output zero. We denote $\widetilde{\mathbf{H}}_2$ to be the refined result of $\hat{\mathbf{H}}_2$ and can get $\widetilde{\mathbf{H}_2}$ by

$$\widetilde{\mathbf{H}}_2(i,j) = \begin{cases} 1, & \text{if } \mathbf{HSR}(\mathbf{I}_3^{(i,j)})=1, \\ 0, & \text{otherwise.} \end{cases} \tag{4.8}$$

The effectiveness of HSR will be discussed later in Chapter 5.3. Finally, we get the predicted heatmap $\hat{\mathbf{H}}$ by

$$\hat{\mathbf{H}}(i,j) = \begin{cases} 1, & \text{if } \hat{\mathbf{H}}_1(i,j)=1 \text{ or } \widetilde{\mathbf{H}}_2(i,j)=1, \\ 0, & \text{otherwise.} \end{cases} \tag{4.9}$$

Building the training dataset for HSR. We randomly crop the 80*80 patches from the original images to build the training dataset for HSD. Specifically, around each dendrite core in the original images, we crop five positive samples and ten negative samples of dendrites. For the positive samples, first we draw a circle **A** as shown in Fig. 4.2 around each of dendrite cores with the radius 4 pixels. Then we

randomly choose five locations inside circle **A** and crop a patch with the size 80*80 around each of the five locations. It is the same for the negative samples, we draw another two circles **B**, **C** as shown in Fig. 4.2 around each of dendrite cores with the radius 15 and 40 pixels respectively. Then we randomly choose five locations inside the green area of circle **B**, blue area of circle **C** respectively and crop a patch with the size 80*80 around each of the locations. We show a positive sample in Fig. 4.2 (b) and two negative samples in Fig. 4.2 (c), (d).

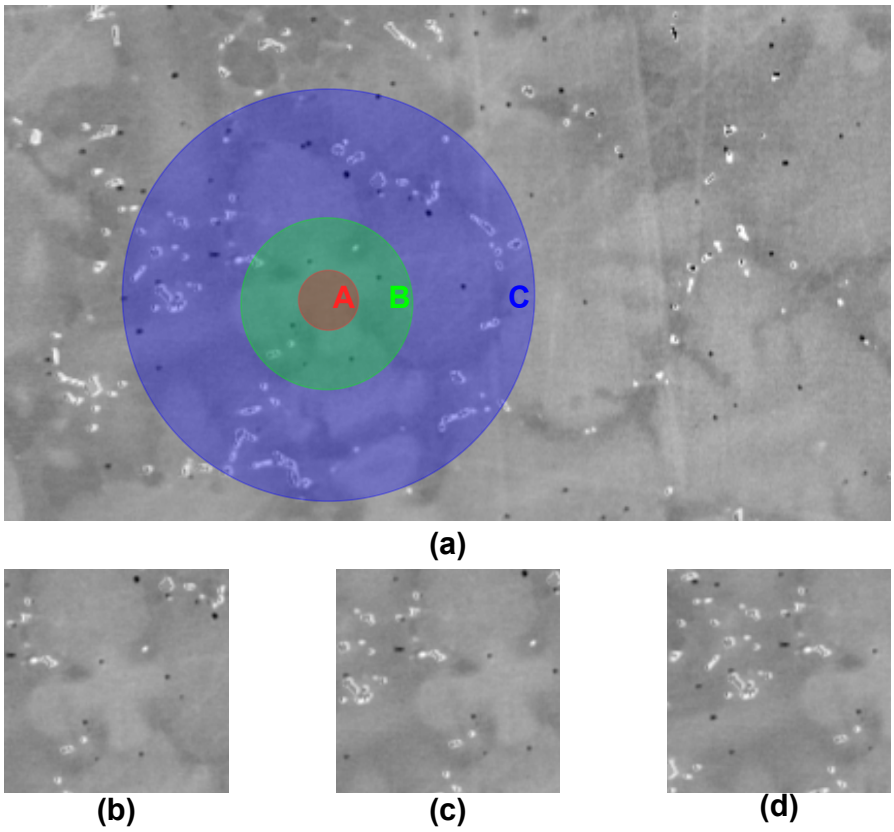

Figure 4.2 Building the training set for HSR. The positive samples are cropped around the randomly selected points inside circle **A**, the negative samples are cropped around the randomly selected points inside the green area of circle **B** and the blue area of circle **C**. The circles **A**, **B**, and **C** are around the dendrite core with radius 4, 15, 40 pixels respectively. (b) is an example of the positive sample and (c), (d) are the examples of negative samples.

4.5 Loss Function

We use the L_2 loss as objective function to train $\mathbf{D}_1$ and $\mathbf{D}_2$ separately which is shown as:

$$\mathcal{L}_1 = \lambda_1 * ||\hat{\mathbf{H}}_1 - \mathbf{H}_1||_2 \tag{4.10}$$

$$\mathcal{L}_2 = \lambda_2 * ||\hat{\mathbf{H}}_2 - \mathbf{H}_2||_2 \tag{4.11}$$

4.6 Implementation details

We train $\mathbf{D}_1$ in ESD with a learning rate of 0.0004. Then, we train $\mathbf{D}_2$ in HSD with a learning rate of 0.0001 by fixing the parameter of $\mathbf{D}_1$. We train both $\mathbf{D}_1$ and $\mathbf{D}_2$ 100 epochs. Then, we train HSR 50 epochs with a learning rate of 0.0001. We set up the threshold α to 0.4, β to 0.1, and λ_1 and λ_2 in the loss function to 0.5. The experiments are conducted on the same platform with two NVIDIA Tesla V100 GPUs.

Chapter 5

Experiments

5.1 Setups

Dataset. The dendrites dataset includes 1,461 microscopic images with resolution 512*512. We divide the dataset into a training set (60%), a validation set (20%), and a test set (20%).

Metrics. We follow the previous object detection methods and use Recall, Precision, and F-score to evaluate our method.

Baselines. We compare our method with three state-of-the-art object detection algorithms: Faster R-CNN, SSD, and Stacked-Hourglass.

5.2 Comparison Results

5.2.1 Quantitative Results

We compare our method with three state-of-the-art object detection methods and show the comparison results in Table 5.1 and Table 5.2. From Table 5.1 and Table 5.2, we can observe that with the deviation distance(the distance between the predicted location and ground truth) of 10 pixels, our method can reach the best result on Recall, Precision, and F-score. Specifically, comparing to the regressing bounding-box based detection [27], our method can achieve 5.77% relative higher Recall, 7.29% relative higher Precision, and 6.53% relative higher F-score and comparing to the segmentation based detection [23], our method can achieve 0.77% relative higher Recall, 1.58% relative higher Precision, and 1.18% relative higher F-score.

Table 5.1 Comparison results of [18], [27], [23], and our method on Recall, Precision, and F-Score.

Methods	Deviation	Recall↑	Precision↑	F-Score↑
SSD	10	0.9585	0.9415	0.9499
Faster-RCNN	10	0.9112	0.9050	0.9081
Stacked-Hourglass	10	0.9564	0.9559	0.9561
Our	**10**	**0.9638**	**0.9710**	**0.9674**

Table 5.2 Comparison results of [18], [27], [23], and our method based on the number of dendrite cores detected.

Methods	Deviation	Total	T-Positive↑	F-Positive↓
SSD	10	1882	1804	112
Faster-RCNN	10	1882	1715	180
Stacked-Hourglass	10	1882	1800	83
Ours	10	1882	**1814**	**54**

5.2.2 Qualitative Results

We display the visualization results on three samples taken from the dendrite dataset in Fig. 5.1. From the visualization result, we can find that our method has the ability to detect more hard samples of dendrite cores such as the blurred or incomplete dendrites inside the yellow boxes, while the other methods can not.

5.3 Ablation Study

5.3.1 Quantitative Results

To validate the effectiveness of our method, we consider three variants: ESD (Chapter 4.1), HSD (Chapter 4.2), and HSR (Chapter 4.4). From Table 5.3 and Table 5.4,

Table 5.3 Ablation study on ESD, HSD, and HSR. We show the result for ESD in the first row, ESD + HSD in the second row and ESD + HSD + HSR in the third row. In the ablation study, we use the deviation distance 10 pixels.

Methods	Deviation	Recall↑	Precision↑	F-Score↑
ESD	10	0.9341	0.9745	0.9538
+HSD	10	0.9670	0.9563	0.9616
+HSR	**10**	**0.9638**	**0.9710**	**0.9674**

we can observe that: ❶ Only using ESD, it gets a lower Recall which leads to a worse F-score. ❷ The HSD is effective to increase the Recall. Specifically, it can increase 3.52% of Recall compared to ESD. ❸ HSR is effective to filter out the false positive prediction of HSD and increase Precision. Specifically, it can increase 1.54% of Precision compared to the HSD. As a result, by taking advantages of both HSD and HSR, our method achieves the best F-score.

5.3.2 Qualitative Results

We show the visualization result in Fig. 5.2 and Fig. 5.3 to prove the effectiveness of each component. From the visualization result, we can find that ❶ HSD has the ability to detect more hard samples of dendrite cores such as the dendrite cores inside the yellow boxes in Fig. 5.2. ❷ HSR can filter out the false-positive detection such as the dendrite cores inside the yellow boxes in Fig. 5.3. As a result, ESD, HSD, and HSR can work together to increase the F-score.

5.4 Exploiting the crop size

During the procedure of destroying the structure of easy samples of dendrites detected by ESD, the crop size is an important factor that can affect the final detected result. In order to exploit the reasonable crop size, we have conducted a series of

Table 5.4 Ablation study on ESD, HSD, and HSR based on the number of dendrite cores detected. We show the result for ESD at the first row, ESD + HSD at the second row and ESD + HSD + HSR at the third row. In the ablation study, we use the deviation distance 10 pixels.

Methods	Deviation	Total	T-Positive↑	F-Positive↓
ESD	10	1882	1800	83
+HSD	10	1882	**1820**	83
+HSR	10	1882	1814	**54**

experiments. We take different crop sizes to destroy the structure of easy samples of dendrites. Specifically, we use the crop size 40*40, 60*60, 80*80, 100*100, and no crop, respectively. The visualization results of the cropped images are displayed in Fig. 5.4.

We evaluate the Recall, Precision, and F-score on the dendrite dataset which is discussed in Chapter 5.1 with different crop sizes and show the results in Table 5.5 and Fig. 5.5. From Fig. 5.5, we can see that ❶ Without destroying the structure of easy samples of dendrites detected by ESD, the final F-score is very low. ❷ It reaches the best F-score when the crop size is 80*80. When continuing to increase the crop size to 100*100, the F-score decreases. The reason for this result is that the smaller crop size or without cropping can not destroy the structure of dendrites detected by ESD and the larger crop size may destroy the potential dendrites which should be detected by HSD. Based on these observations, we finally choose the crop size 80*80.

5.5 Exploiting the intensity

5.5.1 Filling the cropped area with fixed intensity

In addition to the crop size, the crop intensity also has a big influence on the prediction result of HSD. In this work, we conduct a series of experiments to exploit the optimal crop intensity on dendrite dataset. Specifically, we try the fixed intensities of 0, 128,

Table 5.5 Exploiting the optimal crop size after ESD when the deviation distance is 10 pixels. In this work, we try the crop size: 40*40, 60*60, 80*80, 100*100, and no crop, respectively.

Crop size	Recall↑	Precision↑	F-Score↑
no crop	0.9633	0.8229	0.8876
40*40	0.9628	0.9546	0.9587
60*60	0.9675	0.9544	0.9609
80*80	**0.9670**	**0.9563**	**0.9616**
100*100	0.9654	0.9523	0.9588

Table 5.6 Exploiting the optimal intensity in the cropped areas when deviation distance is 10 pixels and cropped size is 80*80. 'Gaussian' means destroy the structure of easy samples by only using Gaussian smoothing. '0 + Gaussian' means first filling the cropped areas with intensity 0 and then processing the cropped boundary by Gaussian smoothing.

Intensity	Recall↑	Precision↑	F-Score↑
0	**0.9670**	**0.9563**	**0.9616**
128	0.9659	0.9478	0.9568
255	0.9675	0.9549	0.9612
Gaussian	0.9617	0.9597	0.9607
0 + Gaussian	0.9617	0.9541	0.9579

and 255, respectively. Based on the discussion in Chapter 5.4, using crop size 80*80 can reach the best F-score. Therefore, we only consider the crop size 80*80 here. The visualization results of different crop intensity are shown in Fig. 5.6. We also evaluate the Recall, Precision, and F-score with the different intensities. From Fig. 5.7 we find that with the intensity 0, it can get the best F-score. Finally, we choose the intensity 0 in this work.

5.5.2 Processing the cropped area with Gaussian smoothing

In addition to fill the cropped area with the fixed intensity, we also exploit using the Gaussian smoothing to process the cropped area. The standard deviation used to generate the kernel of Gaussian smoothing is in the range [1, 20] and the kernel size is 11*11. Based on the discussion in Chapter 5.5.1, using the intensity 0, the HSD can get the best F-score. Therefore, we use the Gaussian smoothing to process the boundaries of the cropped areas with intensity 0. The visualization results are shown in Fig. 5.8 (a). Besides, we also try to destroy the structure of easy samples by only using Gaussian smoothing. The visualization results are shown in Fig. 5.8 (b). From Table 5.6 we can see that with the fixed intensity 0, it can get the best F-score.

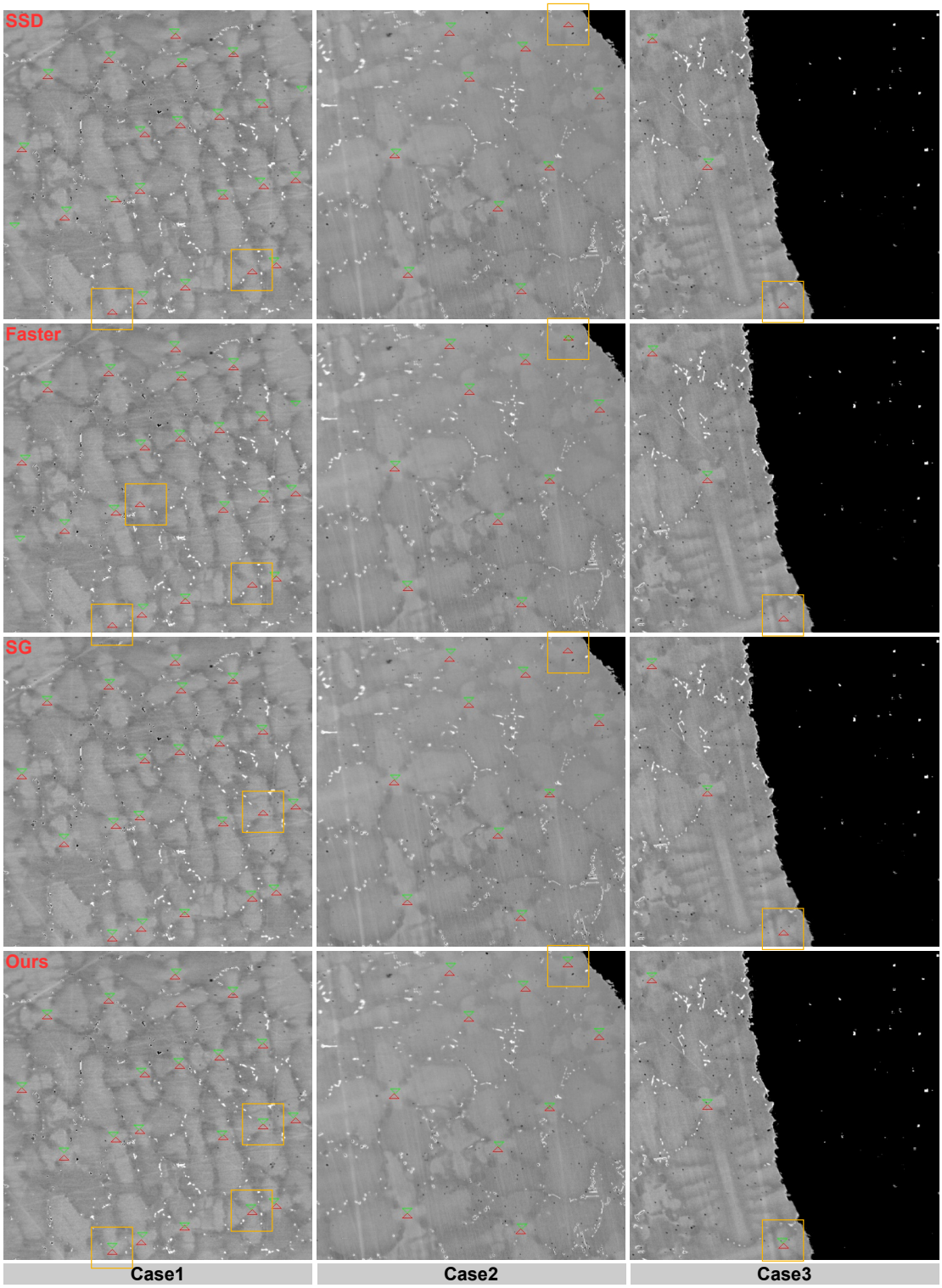

Figure 5.1 Three visualization results of [18], [27], [23], and our method. The upper points of the red triangles and the lower points of green triangles denote the ground truth and predicted dendrite cores respectively. We highlight the hard samples of dendrite cores detected by our method by the yellow boxes.

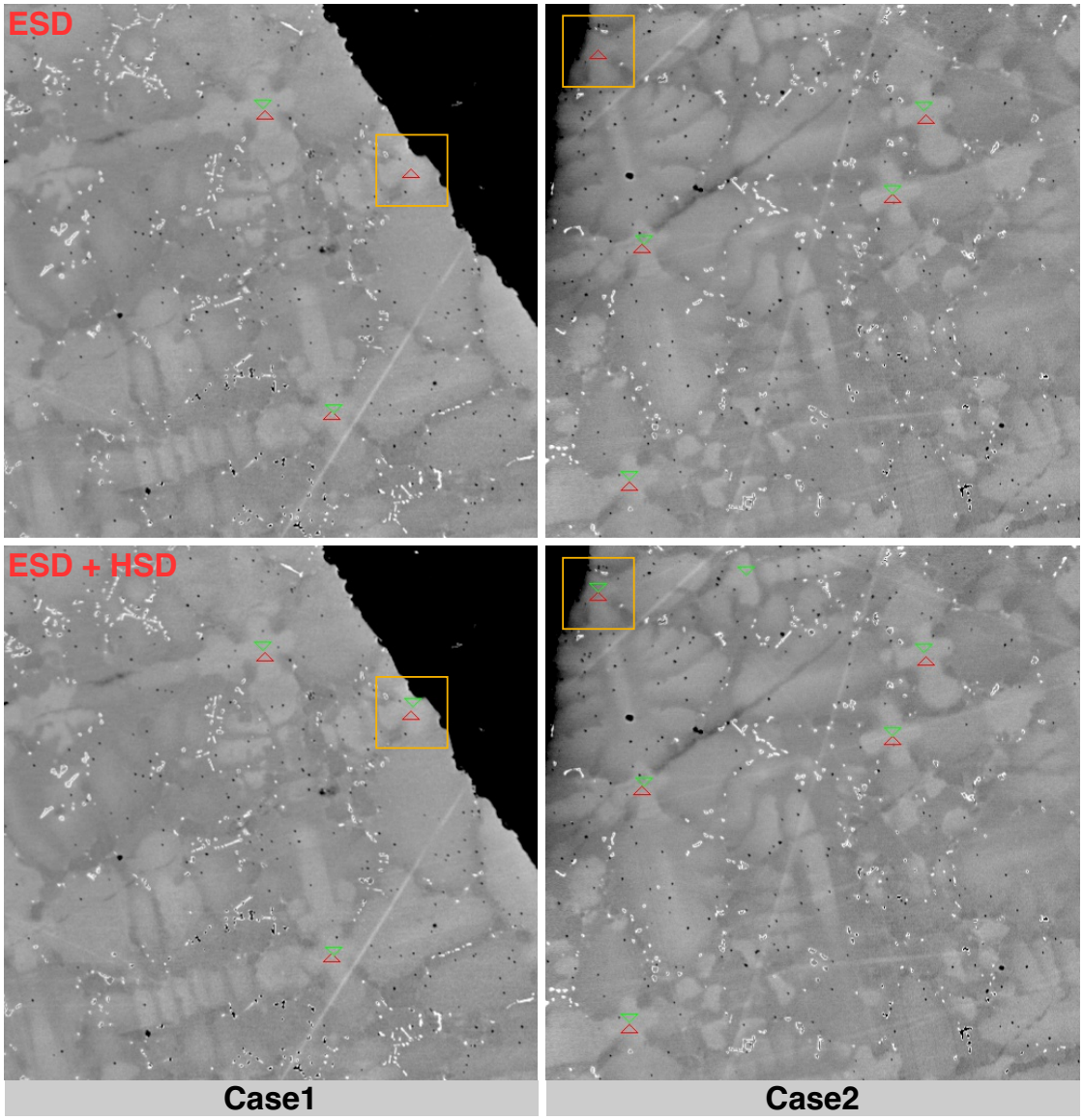

Figure 5.2 Two visualization results of the ESD and ESD + HSD. The upper points of the red triangles and the lower points of green triangles denote the ground truth and the predicted dendrite cores respectively. The yellow boxes denote the hard samples of dendrite cores detected by the HSD.

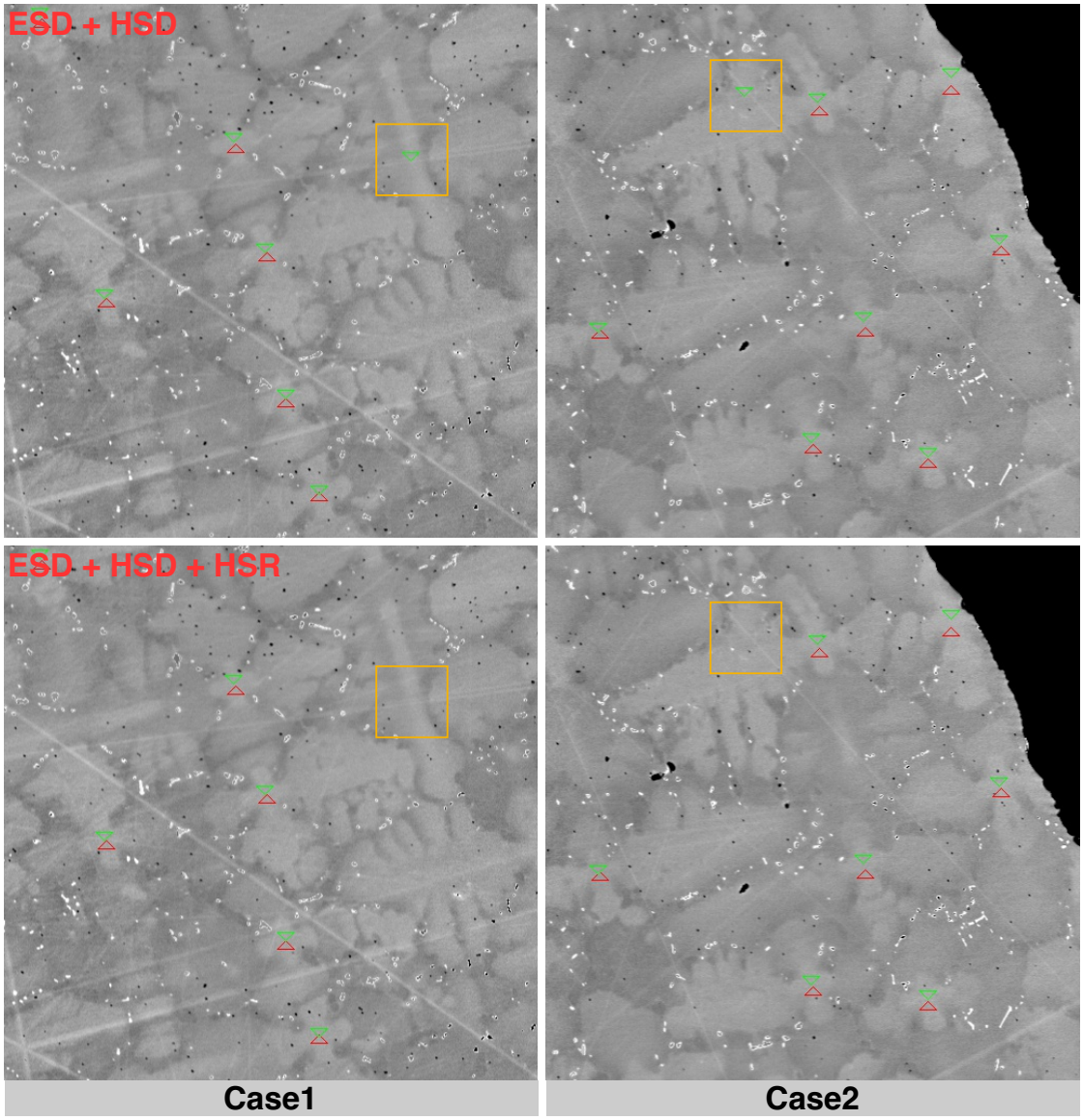

Figure 5.3 Two visualization results of the ESD + HSD and ESD + HSD + HSR. The upper points of the red triangles and the lower points of green triangles denote the ground truth and the predicted dendrite cores respectively. The yellow boxes denote the false positive predictions filtered out by the HSR.

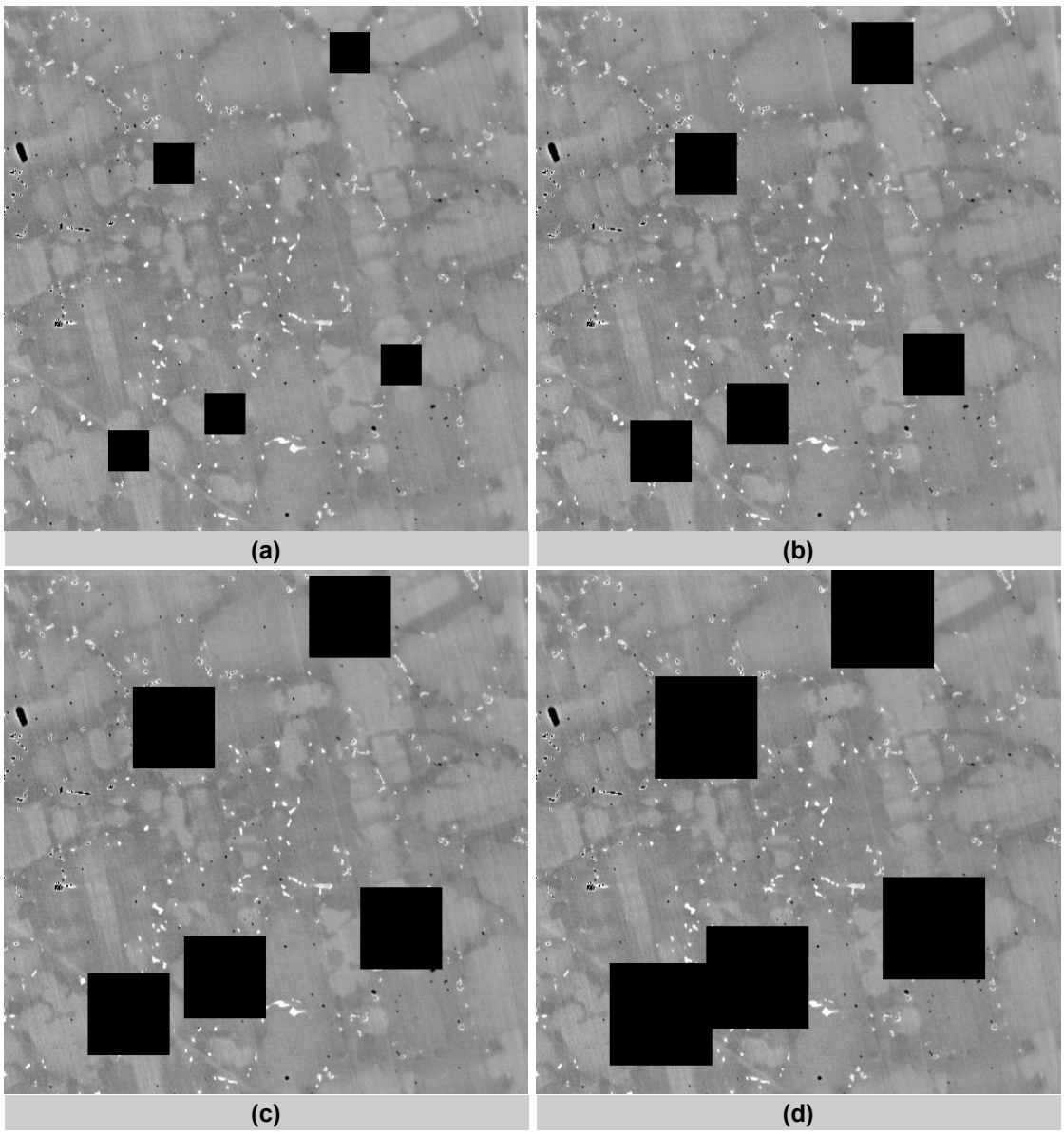

Figure 5.4 The visualization of different crop sizes. (a) denotes the crop size 40*40, (b) denotes the crop size 60*60, (c) denotes the crop size 80*80, and (d) denotes the crop size 100*100.

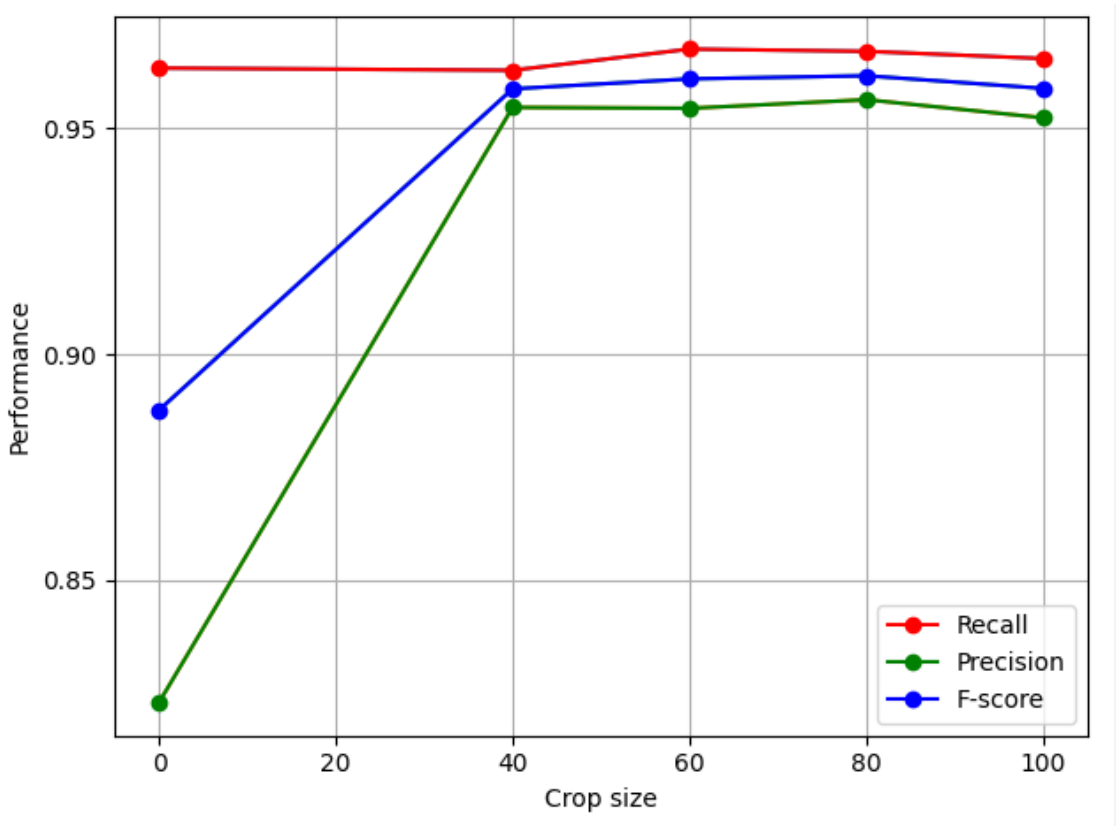

Figure 5.5 Exploiting the optimal crop size after ESD. Red points denote the Recall, green points denote the Precision, and blue points denote the F-score. When the crop size is equal to 80*80, it gets the best F-score.

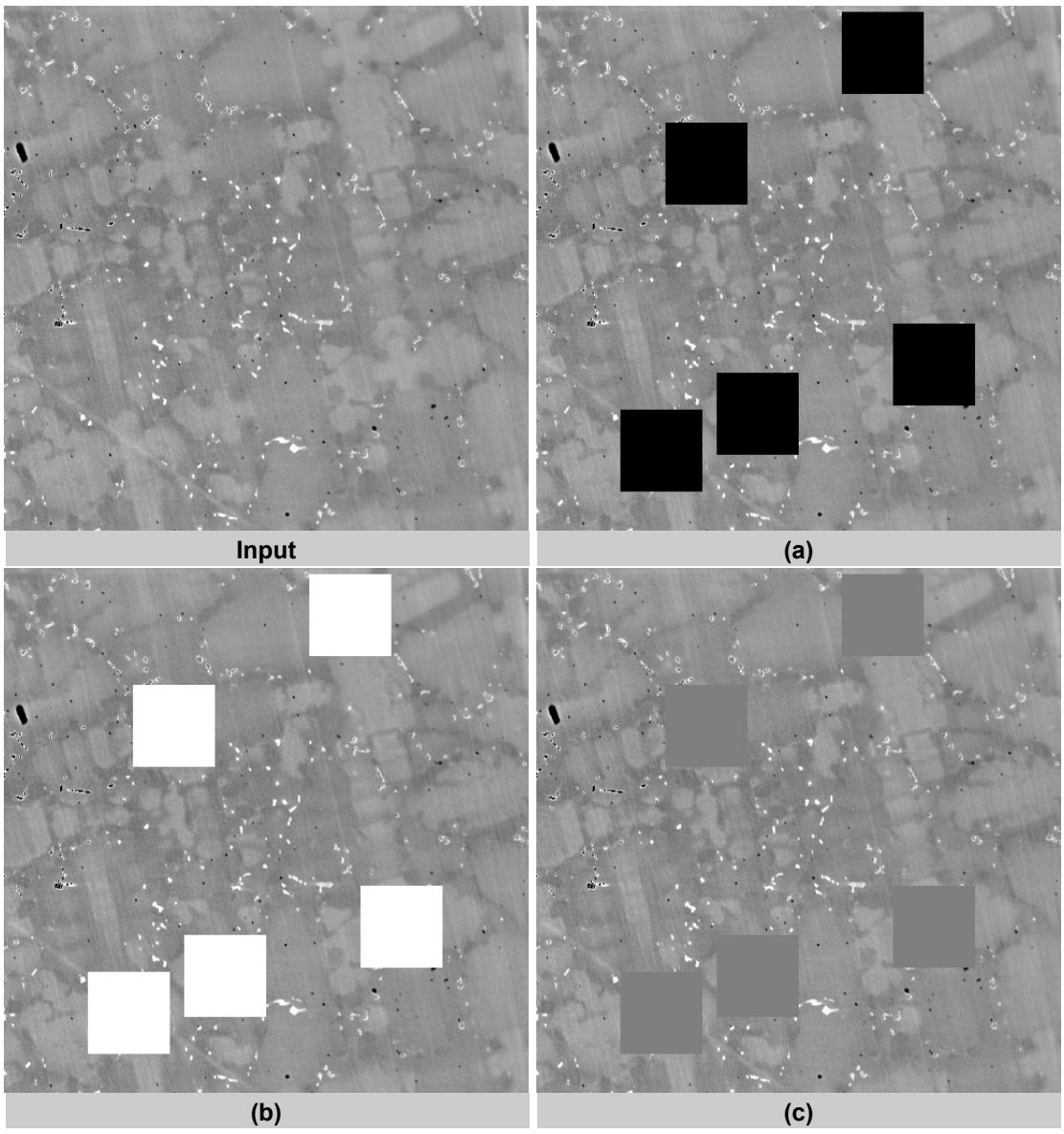

Figure 5.6 Exploiting the optimal intensity to fill the cropped area. The intensity in the cropped areas is 0 for (a), 255 for (b), and 128 for (c).

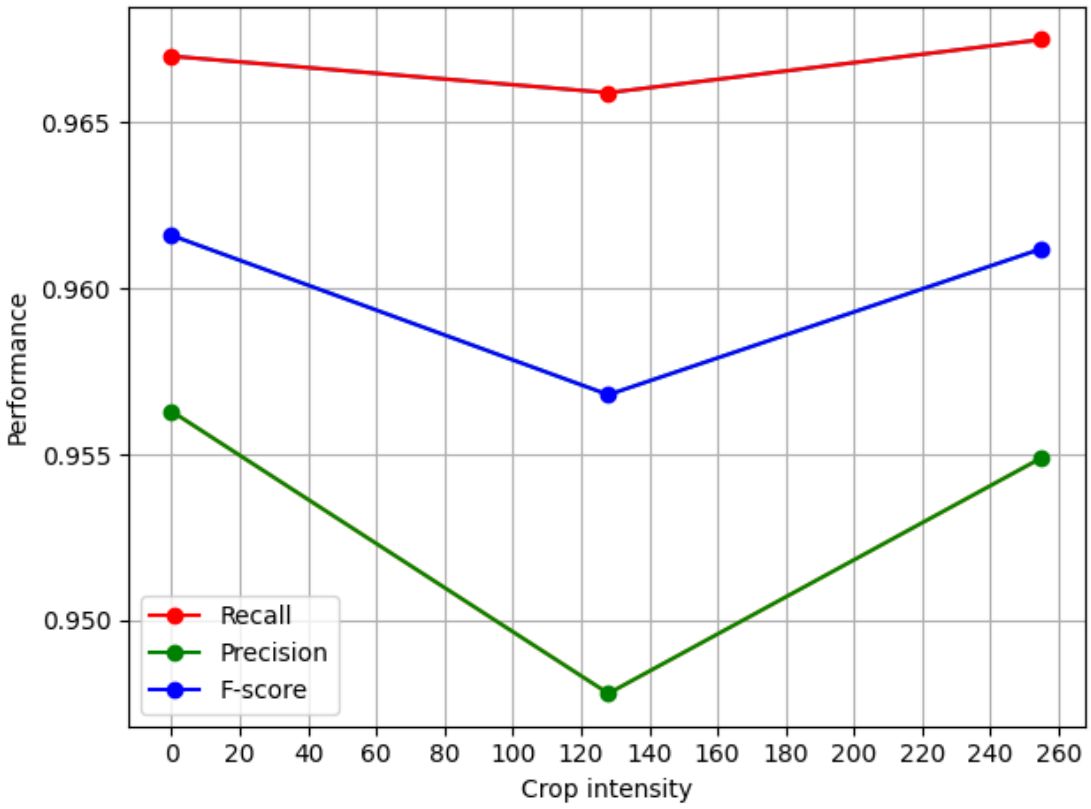

Figure 5.7 Exploiting the optimal intensity to fill the cropped area. In this work, we try three intensities 0, 128, and 255, respectively. Red points denote the Recall, green points denote the Precision, and blue points denote the F-score. When the intensity is equal to 0, it gets the best F-score.

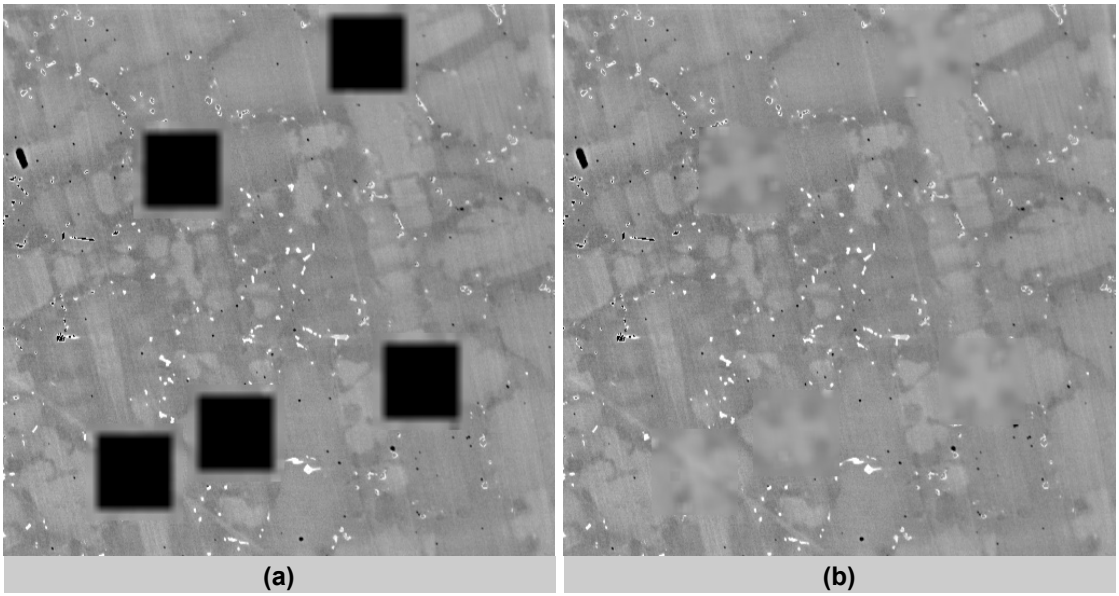

Figure 5.8 Using Gaussian smoothing to process cropped areas. (a) denotes first to fill the cropped area with intensity 0 and then use Gaussian smoothing to process the edges. (b) Only using Gaussian smoothing to destroy the structure of easy samples of dendrites.

Chapter 6

Conclusions

In this work, we formulate the object detection problem as a segmentation task and proposed a novel detection method to detect the dendrite cores directly. The whole pipeline contains Easy Sample Detection, Hard Sample Detection, and Hard Sample Refinement. The Easy Sample Detection and Hard Sample Detection focus on the easy samples and hard samples of dendrite cores respectively. Both of them employ the same Central Point Detection Network but not sharing parameters. The Hard Sample Refinement is a binary classifier which is used to filter out the false positive prediction of Hard Sample Detection. We also conducted a series of experiments for exploiting a way to destroy the structure of easy samples of dendrites detected by Easy Sample Detection. As a result, our proposed detection method outperforms the state-of-the-art baselines on three metrics, i.e., Recall, Precision, and F-score.

One potential limitation of our method is that we use different components to focus on improving the Recall and Precision separately which increases the difficulty to train the whole pipeline. E.g., we use Hard Sample Detection to increase the Recall and use Hard Sample Refinement to increase the Precision. In future work, we will exploit new approaches to combine all the components into a single network.

Bibliography

[1] Dali Chen et al. "Semi-supervised learning framework for aluminum alloy metallographic image segmentation". In: *IEEE Access* 9 (2021), pp. 30858–30867.

[2] Xiaoyan Chen et al. "A light-weighted CNN model for wafer structural defect detection". In: *IEEE Access* 8 (2020), pp. 24006–24018.

[3] R Daudin et al. "Particle-induced morphological modification of Al alloy equiaxed dendrites revealed by sub-second in situ microtomography". In: *Acta Materialia* 125 (2017), pp. 303–310.

[4] Max Frei and Frank Einar Kruis. "FibeR-CNN: Expanding Mask R-CNN to improve image-based fiber analysis". In: *Powder Technology* 377 (2021), pp. 974–991.

[5] Golnaz Ghiasi et al. "Simple copy-paste is a strong data augmentation method for instance segmentation". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2021, pp. 2918–2928.

[6] John W. Gibbs et al. "The three-dimensional morphology of growing dendrites". In: *Scientific Reports* 5.1 (2015), pp. 1–9.

[7] Ross Girshick. "Fast r-cnn". In: *IEEE International Conference on Computer Vision*. 2015, pp. 1440–1448.

[8] Ross Girshick et al. "Rich feature hierarchies for accurate object detection and semantic segmentation". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 580–587.

[9] Kaiming He et al. “Deep residual learning for image recognition”. In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 770–778.

[10] Kaiming He et al. “Spatial pyramid pooling in deep convolutional networks for visual recognition”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.9 (2015), pp. 1904–1916.

[11] Lei Ke, Yu-Wing Tai, and Chi-Keung Tang. “Deep occlusion-aware instance segmentation with overlapping bilayers”. In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2021, pp. 4019–4028.

[12] Tomasz Kazimierz Konopczynski. “Deep Learning Segmentation Algorithms for X-ray CT data”. PhD thesis. 2021.

[13] Alex Krizhevsky. “One weird trick for parallelizing convolutional neural networks”. In: *ArXiv Preprint ArXiv:1404.5997* (2014).

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in Neural Information Processing Systems* 25 (2012), pp. 1097–1105.

[15] Jack Lanchantin et al. “General multi-label image classification with transformers”. In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2021, pp. 16478–16488.

[16] Yann LeCun et al. “Backpropagation applied to handwritten zip code recognition”. In: *Neural Computation* 1.4 (1989), pp. 541–551.

[17] B Li, HD Brody, and A Kazimirov. “Real-time observation of dendrite coarsening in Sn-13% Bi alloy by synchrotron microradiography”. In: *Physical Review E* 70.6 (2004), p. 062602.

[18] Wei Liu et al. “Ssd: Single shot multibox detector”. In: *European Conference on Computer Vision*. Springer. 2016, pp. 21–37.

[19] Boyuan Ma et al. “Deep learning-based image segmentation for al-la alloy microscopic images”. In: *Symmetry* 10.4 (2018), p. 107.

[20] Jianqi Ma et al. “Arbitrary-oriented scene text detection via rotation proposals”. In: *IEEE Transactions on Multimedia* 20.11 (2018), pp. 3111–3122.

[21] Paolo Napoletano, Flavio Piccoli, and Raimondo Schettini. “Anomaly detection in nanofibrous materials by CNN-based self-similarity”. In: *Sensors* 18.1 (2018), p. 209.

[22] H Neumann-Heyme, K Eckert, and C Beckermann. “General evolution equation for the specific interface area of dendrites during alloy solidification”. In: *Acta Materialia* 140 (2017), pp. 87–96.

[23] Alejandro Newell, Kaiyu Yang, and Jia Deng. “Stacked hourglass networks for human pose estimation”. In: *European Conference on Computer Vision*. Springer. 2016, pp. 483–499.

[24] Aditi Panda, Ruchira Naskar, and Snehanshu Pal. “Deep learning approach for segmentation of plain carbon steel microstructure images”. In: *IET Image Processing* 13.9 (2019), pp. 1516–1524.

[25] Lu Qi et al. “Multi-Scale Aligned Distillation for Low-Resolution Detection”. In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2021, pp. 14443–14453.

[26] Joseph Redmon et al. “You only look once: Unified, real-time object detection”. In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 779–788.

[27] Shaoqing Ren et al. “Faster r-cnn: Towards real-time object detection with region proposal networks”. In: *Advances in Neural Information Processing Systems* 28 (2015), pp. 91–99.

[28] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation". In: *International Conference on Medical Image Computing and Computer-assisted Intervention*. Springer. 2015, pp. 234–241.

[29] Numan Saeed et al. "Automatic defects detection in CFRP thermograms, using convolutional neural networks and transfer learning". In: *Infrared Physics Technology* 102 (2019), p. 103048.

[30] Shinji Sakane et al. "Three-dimensional morphologies of inclined equiaxed dendrites growing under forced convection by phase-field-lattice Boltzmann method". In: *Journal of Crystal Growth* 483 (2018), pp. 147–155.

[31] Pierre Sermanet et al. "Overfeat: Integrated recognition, localization and detection using convolutional networks". In: *ArXiv Preprint ArXiv:1312.6229* (2013).

[32] Mohammad Javad Shafiee et al. "Fast YOLO: A fast you only look once system for real-time embedded object detection in video". In: *ArXiv Preprint ArXiv:1709.05943* (2017).

[33] Minghua Sun et al. "Structural and morphological evolution of lead dendrites during electrochemical migration". In: *Scientific Reports* 3.1 (2013), pp. 1–6.

[34] Tomohiro Takaki et al. "Unexpected selection of growing dendrites by very-large-scale phase-field simulation". In: *Journal of Crystal Growth* 382 (2013), pp. 21–25.

[35] Zhi Tian et al. "Fcos: Fully convolutional one-stage object detection". In: *IEEE International Conference on Computer Vision*. 2019, pp. 9627–9636.

[36] Peng Wang et al. "Contrastive learning based hybrid networks for long-tailed image classification". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2021, pp. 943–952.

[37] Xinyi Wu et al. “Dannet: A one-stage domain adaptation network for unsupervised nighttime semantic segmentation”. In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2021, pp. 15769–15778.

[38] ZHANG Yuyan et al. “Internal Defect Detection of Metal Three-dimensional Multi-layer Lattice Structure Based on Faster R-CNN”. In: *Acta Armamentarii* 40.11 (2019), p. 2329.

[39] Gang Zhang et al. “Refinemask: Towards high-quality instance segmentation with fine-grained features”. In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2021, pp. 6861–6869.

[40] Jing Zhang, Zhe Chen, and Dacheng Tao. “Towards high performance human keypoint detection”. In: *International Journal of Computer Vision* 129.9 (2021), pp. 2639–2662.

[41] Youjie Zhou et al. “Large-scale fiber tracking through sparsely sampled image sequences of composite materials”. In: *IEEE Transactions on Image Processing* 25.10 (2016), pp. 4931–4942.

[42] Yukun Zhu et al. “segdeepm: Exploiting segmentation and context in deep neural networks for object detection”. In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 4703–4711.