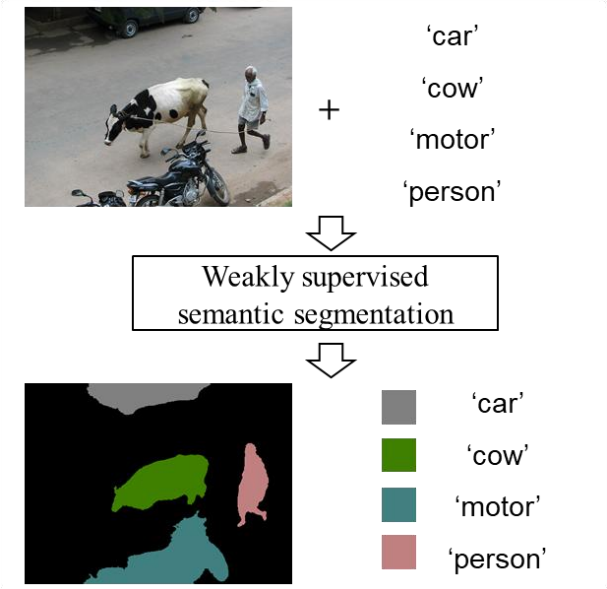
Thèse de doctorat de l'UTT

Zhengyang LYU

Improving the Pseudo-masks for Weakly Supervised Semantic Segmentation



Champ disciplinaire :
Sciences pour l'Ingénieur

2024TROY0035

Année 2024

utt
UNIVERSITÉ DE TECHNOLOGIE
TROYES

THESE

pour l'obtention du grade de

DOCTEUR
de l'UNIVERSITE DE TECHNOLOGIE DE TROYES

en SCIENCES POUR L'INGENIEUR

Spécialité : OPTIMISATION ET SURETE DES SYSTEMES

présentée et soutenue par

Zhengyang LYU

le 22 octobre 2024

Improving the Pseudo-masks for Weakly Supervised Semantic Segmentation

JURY

M. Frédéric MORAIN-NICOLIER	PROFESSEUR DES UNIVERSITES	Président
Mme Christine FERNANDEZ-MALOIGNE	PROFESSEURE DES UNIVERSITES	Rapporteure
Mme Caroline PETITJEAN	PROFESSEURE DES UNIVERSITES	Rapporteure
M. Nicos PARAGIOS	PROFESSEUR	Examinateur
M. Andreas WEINMANN	PROF. DR	Examinateur
M. Alexandre BAUSSARD	PROFESSEUR DES UNIVERSITES	Directeur de thèse
M. Pierre BEAUSEROY	PROFESSEUR DES UNIVERSITES	Directeur de thèse

to my parents,

who have shared with me the most precious knowledge in the world: love.

Acknowledgements

First of all, I would like to extend my deepest thanks to my supervisors, Mr. Pierre BEAUSEROY and Mr. Alexandre BAUSSARD, for their dedicated guidance and support over the past four years. Each discussion with them has given me a more accurate understanding and clearer direction to the research questions, which finally helps me to finish this thesis. I am very grateful for their patience and professional guidance. Working with them has been very interesting and inspiring.

Next, I want to thank the members of my defense committee. My sincere thanks to Mrs. Christine FERNANDEZ-MALOIGNE and Mrs. Caroline PETITJEAN for spending their time to read my manuscript and provide valuable reports. I also want to thank Mr. Frédéric MORAIN-NICOLIER, Mr. Nikos PARAGIOS and Mr. Andreas WEINMANN for agreeing to join in my defense committee and offering insightful comments.

I am also grateful to all the wonderful mentors in the Université de Technologie de Troyes. Special acknowledgment goes to the secretaries in the doctoral school. They are Mrs. Isabelle LECLERCQ, Mrs. Pascale DENIS and Mrs. Thérèse KAZARIAN. I also want to thank all secretaries in LIST3N. They are Mrs. Bernadette ANDRE, Mrs. Veronique BANSE, Mrs. Fabienne BORNOT, and Mrs. Annabelle CHEPY. I am really grateful for their indispensable support in my daily work.

I also want to express my appreciation to my friends who supported me throughout my PhD journey. We shared many wonderful and memorable moments together. They are Jingyang LI, Tingcheng LI, Zhihao GAO, Shengqiang SHI, Yizhan FENG, Yusen LI, Zengxu GUO, Yanli CHEN, Yufeng SHUAI, Ronghua HU, Kandy BENIE, and Hong Nguyen NGUYEN. I will miss the days when we engage in sports and travel together.

A heartfelt thanks to my girlfriend Hanxiao SUN and my cat Xiaowu LI, who have taught me that simply being present is a direct expression of love.

Lastly, I am deeply grateful to my family, especially my parents for their unwavering support, which empowered me to pursue my own path in life.

List of Figures

1.1 Examples of images and corresponding pixel-level labels ground truth. From left to right: images and ground truth from dataset for natural scene image [18], medical image [4], road view [19] and remote sensing image [9]. 2

1.2 Basic training pipeline for weakly supervised semantic segmentation with image-level labels. 3

2.1 Diagram of a neuron. 9

2.2 Structure of a Multi-Layer Perceptron with 2 hidden layers. Arrows are used to represent the connections that are associated to learnable parameters. Gray arrows represent the bias and the black arrows represent the weights associated to the input. In the figure, the bold arrows are related to the corresponding annotations. 11

2.3 (a) Sigmoid activation function and (b) its gradient. 12

2.4 (a) Tanh activation function and (b) its gradient. 13

2.5 (a) ReLU activation function and (b) its gradient. 14

2.6 Illustrations for the converges of learning process with different value of learning rate. Convergence (a) with appropriate learning rate, (b) and (c) for too large or too low learning rate, respectively. The yellow circle represents a local minimum value and the green circle represents the global minimum value. 16

2.7 Illustrations of (a) under-fitting, (b) appropriate fitting and (c) over-fitting. 23

2.8 Architecture of LeNet-5 [47]. 24

2.9 Illustration of obtaining result $\boldsymbol{F}$ by convolution operations using 2D input $\boldsymbol{I}$ and kernel $\boldsymbol{K}$. (a) and (b) use stride that is equal to 1 and 2, respectively, without zero padding. (c) uses stride that is equal to 2 with zero padding to enlarge the input size. 25

2.10 Illustration of obtaining result $\boldsymbol{F}$ by applying an atrous convolution operation on the input feature map $\boldsymbol{I}$, with a stride of 1 and no zero padding. The dilation rate of atrous convolution kernel $\boldsymbol{K}$ is equal to 2. 26

2.11 Visualization of feature maps from different convolutional layers in different CNN architectures by [63], including AlexNet [64] (top row), VGG-19 [65] (middle row) and ResNet-50 [66] (bottom row). . 27

2.12 Illustration of (a) max pooling and (b) average pooling operations. . 27

2.13 Architecture of AlexNet [64]. Boxes indicate the output feature related to certain operations: Light yellow for convolution operations, darker yellow for a ReLU function, orange for max-pooling, light purple for the fully connected layers, and dark purple for the softmax operation. 29

2.14 Illustration of VGG-16 [65]. The boxes with colors represent different process, and the numbers above represent the output feature's size after corresponding processes. The input for the VGG-16 is assumed as a RGB image with the spatial size of 224×224. 30

2.15 Illustration of two types of skip connections used in the ResNets with (a) two and (b) three convolutional layers in the main path [66]. The notation Input, D-dim represents an input feature map with the D channels. The 3×3, D-dim box represents a convolutional layer with convolutional kernels' size of 3×3 and the number of kernels in the layer is D. For 1×1, D-dim, and 1×1, D/4-dim boxes in the figure, the meanings are similar. 30

2.16 Illustration of different segmentation tasks adapted from [82]. 32

2.17 The architecture and skip connections in FCN [10]. Light blue boxes are the resized output features while the purple box indicates the final output processed by the softmax operation. The $\bigoplus$ symbol represents the feature summation process. 33

2.18 The architecture of U-Net [11]. 35

2.19 The building block proposed by [12] to use multi-feature from deep to shallow. 35

2.20 The pipeline of DeepLabV1 [13]. 36

2.21 An illustration of Atrous Spatial Pyramid Pooling module in DeepLabV2 [14]. In the figure, rate is the dilation rate to determine the number of zeros added in the kernel (see Section 2.4.1). 37

3.1 Illustration of (a) under-clustering and (b) over-clustering given by [85]. 41

3.2 A comparison of annotation times given by [30]. 42

3.3 Overview for the two-stage training pipeline of the weakly supervised semantic segmentation with only image-level labels. 43

3.4 Illustration of the classification training pipeline used in method [53]. The GAP box is the global average pooling process. The fc layer & sigmoid box represents a fully connected layer with sigmoid activation function. 45

3.5 Illustration of the classification training pipeline used in method [106]. The GAP & sigmoid box is the global average pooling process with activation by sigmoid function. The 1×1 conv box represents the convolutional layer with convolutional kernels' size of 1×1. . . 46

3.6 Examples of CAM for images with one class. From top to bottom: image and CAM of the image. These CAMs are obtained from a classification network which backbone is ResNet-50. 47

3.7 Examples of CAM for images with two classes. From top to bottom: image, CAM for the 'person' class, and CAM for the other class of the image. These CAMs are obtained from a classification network which backbone is ResNet-50. 47

3.8 Examples of seed mask for images with single class. From top to bottom: image and seed mask. 48

3.9 Examples of seed mask for images with two classes. From top to bottom: image and seed mask. 49

3.10 An illustration of (a) the confusion matrix and (b) four basic cardinalities of the confusion matrix illustrated by [117]. GT represents ground truth. *TP*, *TN*, *FP* and *FN* stand for True Positive, True Negative, False Positive and False Negative, respectively. 49

3.11 Illustration of the relationship between the accuracy of pseudo-masks with the segmentation performance: mIoU (%) of the pseudo-masks on train set (blue dot) and mIoU (%) of the prediction masks on test set (red dot) of PASCAL VOC 2012 [18], given from DeepLabV2 segmentation model [14] trained by pseudo-masks from several WSSS methods [28, 29, 36, 107, 111, 120, 121, 122]. 51

3.12 Visual comparison between the output of the DeepLabV2 segmentation model [14] trained using ground truth or pseudo-masks generated by method in [107]. From top to bottom: image, ground truth, output of the DeepLabV2 network trained with ground truth, pseudo-masks, output of the DeepLabV2 network trained with pseudo-masks. 51

3.13 Visual comparison between the original CAM and enhanced CAM from [36]. From left to right: image, foreground and background estimated in the original CAM, foreground and background estimated in the enhanced CAM obtained by [36]. The estimated background response is deduced as complementing the class-wise CAM. 53

3.14 Illustration of the suppressor used in [39]. The input and output are the original and suppressed feature maps. 54

3.15 Illustration of the extraction of hierarchical feature in [107]. 55

3.16 Illustration of the pipelines to generate CAM from (a) original image, (b) multi-scaling image in [21, 119], and (c) cropped image in [37, 108]. 56

3.17 A toy example to obtain enhanced CAM by manually crop the image from the discriminative region. From left to right: the original CAM with the image, the cropped images with corresponding CAM. The parts of image under the gray masks are cropped and only the rest parts of the image are used as inputs when obtaining the CAM. . . 56

3.18 Illustration of the seeded region growing in method [41]. From left to right: the original image with its image-level label, the seed mask at the beginning, after 6 iterations and after 12 iterations. The pink and purple regions in the seed masks represent the person and horse objects, black regions for background, and the white regions are temporary unlabeled regions. 57

3.19 Illustration of semantic affinity labels in [28]. From left to right: the original image, CAM for the object respect to the image, labels of location pairs with the explanation for the labels, and the refined pseudo-mask on the image. 58

3.20 Examples of auxiliary cues used in weakly supervised semantic segmentation. From top to bottom: image, ground truth, saliency maps and superpixel segments. Saliency maps are generated by method in [135]. Superpixel segments are generated by method in [136]. 60

4.1 A pipeline for classification network training with CAM generation, following [106]. The 1×1 conv box represents the convolutional layer with convolutional kernels' size of 1×1. The GAP & sigmoid box is the global average pooling process with activation by sigmoid function. 64

4.2 A pipeline for classification network training with the proposed $L_{overlap}$ loss to obtain improved CAM without overlapping regions. . 66

4.3 A pipeline for classification network training with the proposed L_{rs} and L_{cls_v} loss to activate less-discriminative regions in the improved CAM. 68

4.4 Illustration of the adaptive tanh activation function: (a) the original image, (b) the feature map for the class cat, the response map obtained by (c) applying the original tanh function and (d) the adaptive tanh function on the feature map. 70

4.5 Illustration of gradient-sensitive regions of the tanh function for positive input. The red and green curves indicate the corresponding output of tanh and its gradients for regions with high gradient sensitivity. 70

4.6 Illustration of (a) a weighted map and (b) the element-wise product of the weighted map with corresponding class-wise feature for the example image shown in Figure 4.4. 72

4.7 A comparison between the baseline CAM and EmeCAM for images from the PASCAL VOC 2012 train set with two classes. From top to bottom: original image, ground truth mask, the baseline CAM for the two classes, and the EmeCAM for the two classes with respect to the image. 74

4.8 A comparison between the seed from the baseline CAM and EmeCAM by applying a fixed threshold for the images from the PASCAL VOC 2012 train set with two classes. The images are the same examples in Figure 4.7. 75

4.9 A comparison between the baseline CAM and EmeCAM for the images from the PASCAL VOC 2012 train set with single class. . . 76

4.10 A comparison between the seed from the baseline CAM and EmeCAM by applying a fixed threshold for the images from the PASCAL VOC 2012 train set with single class. The images are the same examples in Figure 4.9. 76

4.11 A comparison between the baseline CAM and EmeCAM for the images from the PASCAL VOC 2012 train set with two classes. From top to bottom: original image, ground truth mask, the baseline CAM, and the EmeCAM with respect to the image. 77

4.12 A comparison between the seed from the baseline CAM and EmeCAM by applying a fixed threshold for the images from the PASCAL VOC 2012 train set with single class. The images are the same examples in Figure 4.11. 78

4.13 A comparison between the CAM from baseline and the proposed method for the images from the PASCAL VOC 2012 train set. From top to bottom: original image, ground truth mask, the baseline CAM, the CAM obtained where training with the first branch, and the CAM obtained where training with both two branches. 79

4.14 A comparison between the obtained seed from the baseline and the proposed method by subjecting the CAM to a fixed threshold for the images from the PASCAL VOC 2012 train set. From top to bottom: original image, ground truth mask, the seed from the baseline CAM, the obtained seed where training with the first branch, and the obtained seed where training with both two branches. 80

4.15 A comparison between the seed obtained from baseline and the proposed method when using saliency as the background cues for the images from the PASCAL VOC 2012 train set. From top to bottom: original image, ground truth mask, the seed from the baseline CAM, the obtained seed where training with the first branch, and the obtained seed where training with both two branches. 82

5.1 Detailed pipelines of the overall framework of the proposed method. 87

5.2 An example to illustrate the 2-step selection process with (a) the activation map for class horse for the input image shown in Figure 5.1, the visualized (b) location set $\mathcal{K}$ for the training set $\mathcal{T}_1$ and (c) location set $\mathcal{M}$ for the training set $\mathcal{T}_2$. 91

5.3 Detailed pipelines of the framework to evaluate the upper bound of the proposed method using ground truth masks. 92

5.4 SVM classification masks on the PASCAL VOC 2012 train set. . . 94

5.5 Qualitative results of ablation study without and with train set sample filtering process on the PASCAL VOC 2012 train set. 99

5.6 Qualitative results of baseline and the proposed expansion method with dCRF process on images from the PASCAL VOC 2012 train set. 100

5.7 Qualitative results of baseline and the proposed expansion method with IRNet process on images from the PASCAL VOC 2012 train set. 101

5.8 Qualitative segmentation results on the PASCAL VOC 2012 validation set (a) with different sizes of objects and (b) when the ground truth is not total accurate. The white regions labeled in ground truth are used to label the boundaries and to mask out ambiguous, heavily occluded or too small objects. 104

A.1 Illustration des types d'annotations en segmentation faiblement supervisée, en comparaison avec la segmentation supervisée. 112

A.2 Illustration du pipeline d'entraînement en deux étapes pour la segmentation sémantique faiblement supervisée avec des étiquettes au niveau de l'image. 113

A.3 Exemple d'images contenant deux classes (première ligne). La deuxième ligne correspond à la vérité terrain. Les troisième et quatrième lignes montrent les cartes du CAM pour chacune des classes présentent dans l'image, illustrant le fait que les objets ne sont pas forcément entièrement mis en évidence par ces cartes. La dernière ligne correspond aux pseudos masques obtenus à partir des CAM (à comparer avec la deuxième ligne). Le modèle de classification utilisé est ResNet-50 [66]. 115

A.4 Processus d'entrainement du classifieur permettant de générer le CAM. 117

A.5 Procédure d'entrainement du classifieur permettant d'obtenir des CAM excluant les zones de recouvrement. 118

A.6 Schéma de l'entraînement du modèle de classification avec des termes visant à pondérer les cartes d'activation. 119

A.7 Comparaison entre le CAM de base et CAM amélioré pour des images du jeu de données d'entraînement PASCAL VOC 2012 avec deux classes. De haut en bas : image originale, masque de vérité terrain, CAM de base pour les deux classes, et CAM amélioré pour les deux classes respectives. 121

A.8 Comparison entre les pseudo masques obtenus avec la méthode de base de génération des pseudo masque (ligne 3) et notre méthode (ligne 4). Les images sont les même que dans la Figure A.7. 122

A.9 Comparaison du CAM classique et de la méthode proposée pour des images du jeu de données d'entraînement PASCAL VOC 2012. De haut en bas : image originale, masque de vérité terrain, CAM de base, CAM obtenu avec la méthode d'entraînement utilisant L_{rs}, et CAM obtenu en utilisant l'entraînement avec L_{rs} et L_{cls_v}. 123

A.10 Comparison entre les pseudo masques obtenus à partir du CAM classique ou de la deuxième méthode. La première ligne correspond aux image originales, la deuxième ligne est la vérité terrain, la troisième ligne correspond aux pseudo masques pour le CAM classique, la quatrième ligne correspond aux pseudo masque en ne considérant que la première branche de la deuxième approche (i.e. L_{rs}), et la dernière ligne correspond aux pseudo masques obtenus avec la deuxième méthode (incluant L_{rs} et L_{cls_v}). 125

A.11 Génération de pseudo masques de bonnes qualité à partir des CAM via l'utilisation de deux étapes de post-traiements. 125

A.12 Schéma détaillant les étapes de la méthode proposée. 126

A.13 Schéma du processus permettant d'évaluer la borne supérieure de la méthode proposée, en utilisant la vérité terrain. 128

A.14 Résultats de segmentation sur l'ensemble de validation PASCAL VOC 2012 (a) avec des objets de différentes tailles et (b) lorsque la vérité terrain n'est pas totalement exacte. 132

List of Tables

2.1 The ResNets achitectures [66]. FLOPs are the floating point operations. 31

3.1 The segmentation results for the WSSS methods with (noted by *) or without saliency in validation set and test set of PASCAL VOC 2012. The evaluation metric is mIoU (%). 60

4.1 mIoU (%) of seed obtained from the baseline method and EmeCAM on the PASCAL VOC 2012 train and augmented train sets. 77

4.2 The classification accuracy (%) of the model trained using the baseline method and the proposed method to decrease the overlapping regions (refereed to as EmeCAM in the table) on the PASCAL VOC 2012 train and augmented train sets. 78

4.3 mIoU (%) of the seed obtained by model trained using the baseline method and considering ablation with the proposed method to enlarge activation regions on the PASCAL VOC 2012 train and augmented train sets. 81

4.4 Classification accuracy (%) of the model trained using the baseline method and considering ablation in the proposed method to enlarge activation regions on the PASCAL VOC 2012 train and augmented train sets. 83

5.1 Architecture of the backbone network adopted from ResNet-50. Building blocks are shown in brackets, with the numbers of blocks stacked. Down-sampling is performed by each layer with a stride of 2. Input image is 448×448. 93

5.2 Comparison of segmentation mIoU (%) scores using different features on the PASCAL VOC 2012 train set. The highest values are highlighted by bold. bkg represents the background. 95

5.3 mIoU (%) of the SVM classification mask using F_1, F_2 or F_3 as shallow feature and different number of iterations in the self-enhanced SVM process. The best results for each feature are highlighted in bold. 97

5.4 Precision (%) of sample selection for categories without and with train set Sample Filtering Process (SFP) on the PASCAL VOC 2012 train set. bkg presents the background. mean presents the mean precision for all objects classes. 98

5.5 mIoU (%) considering ablation in our MIP expansion method, using the PASCAL VOC 2012 train and augmented train sets. SSS = Seed Sample Selection. SFP = train set Sample Filtering Process. SeT = Self-enhanced Training. The best results are highlighted in bold. 98

5.6 mIoU (%) of pseudo-masks with different refinement process on the PASCAL VOC 2012 train set. The best results are shown in bold. Mask means the refined pseudo-masks by both dCRF and IRNet post processes. 100

5.7 mIoU (%) of segmentation results on the PASCAL VOC 2012 validation and test set. All methods in table take ResNet-38 as backbone network and only use image-level label as supervision. 102

5.8 mIoU (%) of segmentation results on the PASCAL VOC 2012 validation and test set. All methods in table take ResNet-101 as backbone network and only use image-level label as supervision. 103

A.1 mIoU (%) calculé sur l'ensemble de la base d'entrainement de PASCAL VOC pour les pseudo masques obtenus à partir du CAM classique (noté BaseCAM) et de notre CAM (noté EmeCAM). 122

A.2 mIoU (%) des pseudo masques en fonction des éléments considérés. La base de donnée est la base d'entrainement augmené de PASCAL VOC 2012. 124

A.3 Comparison entre la vérité terrain et la sortie du SVM en utilisant comme métrique le IoU pour chaque classe et le mIoU (%) global, en fonction du choix de $\boldsymbol{S'}$: F_0 (l'image originale), F_1, F_2, F_3, F_4 et F_5 (les différentes couches de ResNet-50). La base de données est l'ensemble d'entrainement de PASCAL VOC 2012. Les meilleurs résultats sont surlignés en gras. Notez que 'bkg' représente la classe arrière-plan. 129

A.4 mIoU (%) en fonction des étapes considérées pour générer les pseudo masques, utilisant les ensembles d'entraînement PASCAL VOC 2012. SSS pour Seed Sample Selection pour obtenir $\mathcal{T}_1$. SFP pour Sample Filtering Process pour obtenir $\mathcal{T}_2$. SeT pour Self-enhanced Training (SVM auto-amélioré). Les meilleurs résultats sont en gras. 130

A.5 mIoU (%) obtenu en segmentation sur la base PASCAL VOC 2012 de validation et de test, en utilisant ResNet-38. 130

A.6 mIoU (%) obtenu en segmentation sur la base PASCAL VOC 2012 de validation et de test, en utilisant ResNet-101. 131

Contents

List of Figures **5**

List of Tables **13**

1 Introduction **1**

1.1 Context 1

1.2 Outline of the thesis 4

2 Background knowledge **7**

2.1 Introduction 8

2.2 Building components and forward mechanics for the neural networks 8

2.2.1 Neurons 8

2.2.2 Forward computation in the Multi-Layer Perceptron 9

2.2.3 Activation functions 10

2.3 Neural networks learning process 15

2.3.1 Objective 15

2.3.2 Loss functions 17

2.3.3 Optimizers 20

2.3.4 Training from a pre-trained network 22

2.4 Convolutional Neural Networks 23

2.4.1 Convolutional layers 24

2.4.2 Pooling layers 26

2.4.3 Fully connected layers 28

2.5 CNN-based classification networks 28
2.5.1 AlexNet 28
2.5.2 Visual Geometry Group (VGG) Networks 29
2.5.3 Residual Networks (ResNets) 30
2.6 CNN-based segmentation networks 32
2.6.1 Fully Convolutional Network 33
2.6.2 U-Net and Feature Pyramid Network 34
2.6.3 DeepLab networks 34
2.7 Conclusion 37

3 Related works 39

3.1 Introduction 39
3.2 Semantic segmentation under different levels of supervision 40
3.2.1 Fully supervised semantic segmentation 40
3.2.2 Unsupervised semantic segmentation 40
3.2.3 Semi-supervised semantic segmentation 41
3.2.4 Weakly supervised semantic segmentation 41
3.3 Overview of the two-stage training pipeline for WSSS 42
3.3.1 The generation of CAM 44
3.3.2 Obtaining pseudo-masks from CAM 47
3.3.3 Training a segmentation model with pseudo-masks 49
3.4 Recent methods in WSSS 52
3.4.1 Generating enhanced CAM 52
3.4.2 Obtaining improved pseudo-masks 56
3.4.3 Adaptive training strategies for the segmentation models . . 59
3.5 Auxiliary cues for WSSS 59
3.6 Conclusion 61

4 New classification network training methods to enhance CAM 63

4.1 Introduction . . . 63

4.2 Motivation . . . 64

4.3 Proposed methods . . . 65

4.3.1 New training method to decrease overlapping regions in CAM 66

4.3.2 New training method to enlarge activation regions in CAM . 67

4.4 Experimental results . . . 73

4.4.1 Experiments settings . . . 73

4.4.2 Experimental results for enhanced CAM with decreased overlapping regions . . . 74

4.4.3 Experimental results for enhanced CAM with enlarged activation regions . . . 79

4.5 Conclusion . . . 83

5 A multi-feature image-specific pixel classification approach for pseudo-masks improvement 85

5.1 Introduction . . . 85

5.2 Motivation . . . 86

5.3 Proposed method . . . 87

5.3.1 The generation of seed . . . 88

5.3.2 Sample selection for pixel classifier training . . . 89

5.3.3 Self-enhanced SVM . . . 91

5.4 Shallow feature selection and method's performance limit . . . 91

5.4.1 Experimental settings . . . 92

5.4.2 Experiment results . . . 94

5.5 Experiments . . . 95

5.5.1 Experimental settings . . . 95

5.5.2 Choice of the shallow feature and the iteration number in the self-enhanced SVM . . . 96

5.5.3 Ablation studies . . . 97

5.5.4 Comparison with state-of-the-arts . . . 102

5.6 Conclusion . . . 104

6 Conclusions and prospects **107**

6.1 Conclusions 107

6.2 Prospects 109

A Résumé en français **111**

A.1 Introduction générale 111

A.2 Méthodes d'amélioration des CAM 115

A.2.1 Génération des CAM et des pseudo masques 116

A.2.2 CAM excluant les zones de recouvrement 117

A.2.3 CAM obtenu à partir d'un classifieur entrainé avec différentes contraintes 118

A.2.4 Résultats expérimentaux 120

A.3 Méthode d'amélioration des pseudo masques 123

A.3.1 Méthode d'expansion proposée 124

A.3.2 Sélection de la couche du CNN et limites de performance de la méthode 127

A.3.3 Résultats expérimentaux 129

A.4 Conclusions et perspectives 131

A.4.1 Conclusions 131

A.4.2 Perspectives 134

Bibliography **136**

Chapter 1

Introduction

Contents

1.1 Context **1**

1.2 Outline of the thesis **4**

1.1 Context

Semantic segmentation is an important task in computer vision, which aims at assigning each pixel into a specific class for a given image [1, 2]. It has been widely applied in real scenario such as medical imaging [3, 4], autonomous driving [5, 6, 7], or remote sensing imagery [8, 9]. In recent years, deep-learning approaches, especially methods based on Convolutional Neural Networks (CNN) [10, 11, 12, 13, 14, 15, 16], have shown outstanding performances in semantic segmentation, thanks to their versatile representation learning ability. However, a main weakness of using the CNN-based methods in semantic segmentation is that a large amount of fully-annotated data, i.e., images with pixel-level labels, are required for training such networks. Figure 1.1 presents examples of images from different fields with the corresponding pixel-level labels ground truth. The acquisition of fully-annotated dataset is time-consuming and expensive [17]. Compared to annotating general images, pixel-level manual annotation of specialized images, like medical images, requires expert knowledge within specific domains, thus increasing the cost of annotation [9]. Then, in these fields, the shortage of annotated dataset is common [3].

Semantic segmentation can be grouped into different approaches based on the levels of used supervision. In Fully Supervised Semantic Segmentation (FSSS), the segmentation models are trained by images with pixel-level labels ground truth. Meanwhile, Unsupervised Semantic Segmentation (USS) aims to group pixels within images into meaningful semantic segments without using any annotations. However, this approach is challenging as the performance substantially degrades due to the lack of correspondence between the images and classes. Semi-supervised Semantic Segmentation (SSSS) has been proposed to alleviate the an-

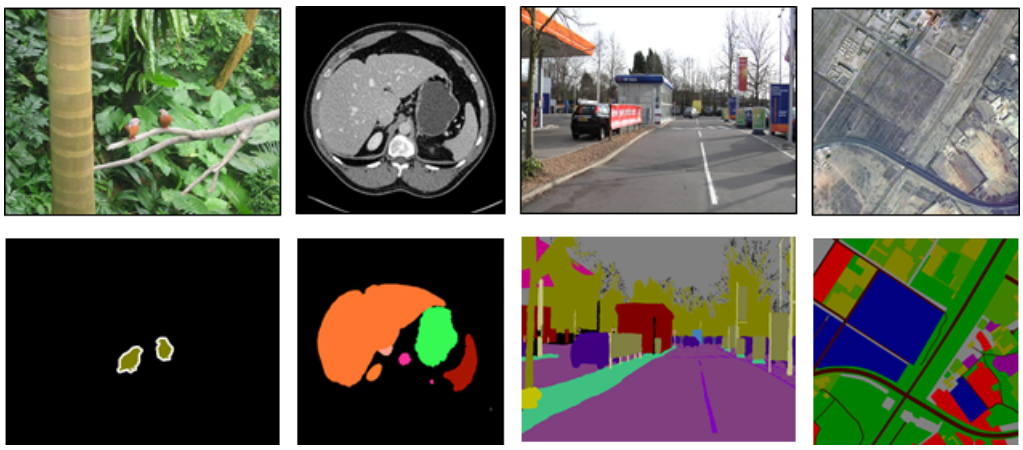


Figure 1.1: Examples of images and corresponding pixel-level labels ground truth. From left to right: images and ground truth from dataset for natural scene image [18], *medical image* [4], *road view* [19] *and remote sensing image* [9].

notation burden of fully supervised semantic segmentation by leveraging a limited number of annotated images with pixel-level labels and a large amount of unlabeled images. This approach still requires a certain amount of images with pixel-level ground truth, and the number of these labeled images significantly impacts the final segmentation results. Similarly, Weakly Supervised Semantic Segmentation (WSSS) has also been introduced to reduce the annotation burden of fully supervised approaches. This approach uses images annotated with weak supervision, such as image-level labels [20, 21, 22, 23, 24, 25, 26, 27, 28, 29], points [30, 31], scribbles [32, 33], and bounding boxes [34, 35]. Specifically, image-level labels only describe the presence of certain classes in the image without offering any location information for the objects. Labeled points are defined as a few discrete annotated pixels within objects. Scribbles provide sets of labeled pixels for classes by drawing continuous curves. Bounding boxes indicate the presence and location of objects by enclosing each object in a rectangular box. Compared with the pixel-level labels, weak supervision provides less location information for the objects in the given images, which significantly reduce the labeling time. Among all weak supervision, image-level labels are the easiest annotations to obtain, where only the existence of objects in an image is noted without any location information. In this thesis, we mainly focus on this framework.

The basic training pipeline for recent WSSS methods is shown in Figure 1.2. In this pipeline, pseudo-masks play a crucial role. They are firstly generated by only using image-level labels, and then served as the training targets to train a segmentation model. It is obvious that the quality of the pseudo-masks directly influence the performance of the segmentation. While WSSS has become increasingly popular with many research developments, the accuracy of pseudo-masks still has a significant gap compared with the ground truth. Therefore, the primary research objective of this thesis is to investigate approaches to improve the accuracy of pseudo-masks, which is also the most popular research direction in recent WSSS methods.

The generation of pseudo-masks contains two steps: The first step is to obtain

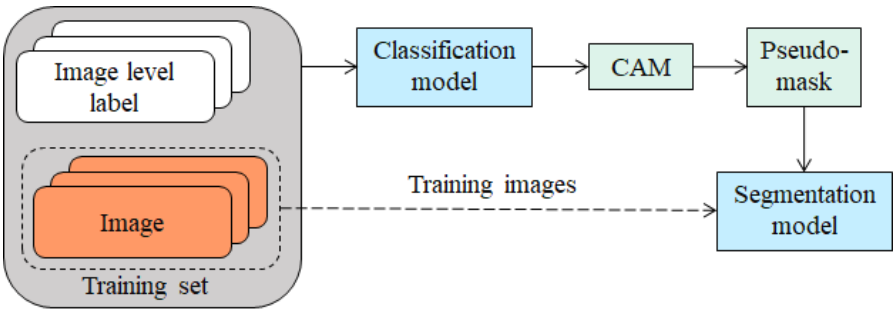


Figure 1.2: Basic training pipeline for weakly supervised semantic segmentation with image-level labels.

the Class Activation Map (CAM), which is a three-dimensional matrix produced from the feature given by a classification network. This classification network has been trained from images with their corresponding image-level labels. The values in CAM can be interpreted as scores for pixels to be assigned to each class. Due to the pooling process used in the network, the spatial size (height times width) of CAM is smaller than the input image one. The depth of the CAM equals the number of classes predefined in the dataset, with each channel of the activation map related to a specific class. The second step is to generate two-dimensional pseudo-masks from the CAM. A typical process is to firstly resize CAM to the spatial size of the input image and then threshold the resized CAM by a fixed threshold with an argmax process to assign a label to each pixel. The values in the pseudo-masks represent the class labels assigned to each pixel of the input image. Therefore, the research objective of this thesis can be achieved by addressing two research questions aimed at these two steps:

1. How to obtain more accurate CAM that will generate better pseudo-masks?
2. How to produce improved pseudo-masks from given CAM?

For the first question, recent methods attribute the origin of the CAM issues to the mismatch between the image-level classification task, that produces features used to generate the CAM, and the need for detailed object segmentation in CAM [21, 36, 37, 38, 39, 40]. CAM tends to highlight only the most valuable and discriminative information for classification, which does not match the need for segmentation of locating accurately the classified items. In Chapter 4, we propose two methods to address this mismatch by modifying the training objective function of the network to take into account the produced CAM and its properties while training the classifier. The modifications tend to address two known problem: reduce ambiguity and avoid the CAM to highlight only the discriminative regions that correspond to the most significant clues of each object. Ambiguity exists where activation values for different classes appear at the same spatial location, which results in the appearance of overlapping regions in the CAM. To tackle this issue, a method that penalizes these regions during the training of the classification network is introduced. To overcome the limitation of CAM that it concentrates only on discriminative regions, we propose a method to increase activation across less-discriminative regions by reducing the impact of the discriminative regions on the classification task. The classification network is tuned to focus more on

global activation rather than being distracted by local high-response regions. In fact, these two issues are to some extent contradictory, which makes this problem especially hard to tackle.

For the second question, the main objectives when generating improved pseudo-masks from CAM involve refining the object boundaries and compensating for the missing details of objects within CAM, thereby obtaining sufficiently accurate pseudo-masks that can be used to train a segmentation model. One solution is to expand the confident and reliable location information provided by CAM toward the less-confident regions and infer the corresponding pixel labels [41, 25, 42]. This process requires effectively balancing resolution and semantics in used features. Since the semantic features are related to the images context, it means that exploiting the unique characteristics in each image should help to improve the segmentation results. In Chapter 5, we propose a new approach to obtain improved pseudo-masks from the CAM based on an image-specific pixel classifier using multi-feature to make a balance between both high-resolution shallow feature and low-resolution deep feature with semantics.

1.2 Outline of the thesis

The work and developments carried out in this thesis are organized in 5 chapters in addition to this introduction.

Chapter 2 introduces the background knowledge for this thesis. We present the foundation details of the Neural Networks (NN) and the Convolutional Neural Networks. Typical supervised CNN-based image classification models and semantic segmentation models are presented.

Chapter 3 presents the related works in the weakly supervised semantic segmentation. Various levels of supervision in semantic segmentation, including fully supervised, unsupervised, semi-supervised, and weakly supervised approaches, are compared. An overview of weakly supervised semantic segmentation using only image-level labels is introduced. Recent works are analyzed to provide insights for new methods proposed in this thesis. The auxiliary data, which are widely used and able to improve the final segmentation performance, are discussed in the end of this chapter.

Chapter 4 proposes two methods in training the classification model to obtain enhanced CAM. They are respectively proposed to decrease CAM ambiguity in the overlapping regions, and increase activation in less-discriminative regions to avoid that CAM only focuses on the most relevant parts of the classified objects.

Chapter 5 proposes a new method based on a multi-feature image-specific pixel classification approach to generate improved pseudo-masks from CAM. The proposed method aims at expanding the reliable location information provided by CAM to infer pixel labels in the whole image, by using both high semantics deep features and high resolution shallow features. This approach enables to consider

the specificity of each image. The segmentation networks trained by the obtained pseudo-masks reach outstanding performance.

Chapter 6 draws the conclusion and discuss the prospects based on the studies of this thesis.

Chapter 2

Background knowledge

Contents

2.1 Introduction 8

2.2 Building components and forward mechanics for the neural networks 8

2.2.1 Neurons 8

2.2.2 Forward computation in the Multi-Layer Perceptron . . 9

2.2.3 Activation functions 10

2.3 Neural networks learning process 15

2.3.1 Objective 15

2.3.2 Loss functions 17

2.3.3 Optimizers 20

2.3.4 Training from a pre-trained network 22

2.4 Convolutional Neural Networks 23

2.4.1 Convolutional layers 24

2.4.2 Pooling layers 26

2.4.3 Fully connected layers 28

2.5 CNN-based classification networks 28

2.5.1 AlexNet 28

2.5.2 Visual Geometry Group (VGG) Networks 29

2.5.3 Residual Networks (ResNets) 30

2.6 CNN-based segmentation networks 32

2.6.1 Fully Convolutional Network 33

2.6.2 U-Net and Feature Pyramid Network 34

2.6.3 DeepLab networks 34

2.7 Conclusion 37

2.1 Introduction

This chapter outlines the background knowledge of deep learning methods cared in this thesis. Firstly, in Section 2.2, we introduce the basic components and forward pass of the Neural Networks (NN). Then, the learning process of the neural networks is presented in Section 2.3. In order to simplify the presentation, the fully supervised learning is considered. We highlight the hierarchical architecture with different operations of convolutional neural networks (CNN) in Section 2.4. As one of the most popular methods in deep learning, CNN processes image data successfully, reduces parameter counts, improves model generalization, and achieves superior performance in image-related problems, such as image classification and segmentation. Typical CNN models for these two tasks are introduced in Section 2.5 and Section 2.6, respectively. We give concluding comments in Section 2.7.

2.2 Building components and forward mechanics for the neural networks

Neural networks are important methods in machine learning and deep learning, which mimic the human brain's neuron-based information processing [43]. The basic element in the neural networks is the single neuron. It performs basic computations to process data, which is similar to the transmission of the electrical signals in biology neuron with various activation strength. Activation functions are used to introduce non-linearity with respect to the input, which is essential for tackling complex non-linear problems far beyond the capability of linear models. A neural network is formed by connecting multiple neurons. Theoretically, neural networks have the capability to approximate any continuous function [44]. This universality means they can tackle diverse and complex problems, if appropriate parameters can be found.

2.2.1 Neurons

Neurons are the basic computational units in neural networks. The computation of a neuron is related to the inputs, weights, bias and an activation function, all of which are combined to produce an output, as depicted in Figure 2.1.

Specifically, we define $\boldsymbol{x} = [x_1, ..., x_n]^\top$ as the input data that consists of n variables ($\boldsymbol{x} \in \mathbb{R}^n$) and $\boldsymbol{w} = [w_1, ..., w_n]^\top$ as the corresponding weights. The input bias term is set to 1 and its corresponding weight is b is used to guarantee that there will still be an response in the neuron, even when all the inputs are zeros. The learnable parameters $\boldsymbol{\theta}$ for a single neuron consists of $\boldsymbol{w}$ and b.

The neuron firstly computes a dot product of the input data $\boldsymbol{x}$ with its corre-

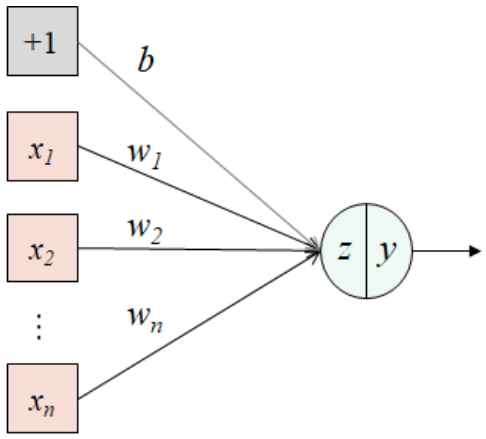


Figure 2.1: Diagram of a neuron.

sponding weights $\boldsymbol{w}$ plus the bias b:

$$z = \boldsymbol{w}^\top \boldsymbol{x} + b = \sum_{i=1}^{n} w_i x_i + b, \tag{2.1}$$

where z is named as the net input of the neuron [45].

Then, the activation function $f(\cdot)$ is applied to obtain the output y of the neuron:

$$y = f(z). \tag{2.2}$$

The activation function $f(\cdot)$ is crucial in neural networks, as it can introduce non-linearity, allowing the network to learn beyond simple linear relationships. We explore various activation functions and their impacts in the Section 2.2.3.

2.2.2 Forward computation in the Multi-Layer Perceptron

Multi-Layer Perceptron (MLP) is a type of neural networks composed of multiple layers of neurons, which are interconnected from one layer to the next. Generally, MLP consists of an input layer, an output layer and one or several hidden layers.

Similar to the notations of the computation in a single neuron, suppose that the first layer, i.e., the input layer, consists of n input data, which is $\boldsymbol{x} = [x_1, x_2, ..., x_n]^\top$. For the layer o, $o > 1$, which contains k neurons and each neuron receives $m + 1$ input from the previous layer, the learnable parameters $\boldsymbol{\theta}^{(o)}$ comprise a $k \times (m+1)$ matrix, which consists of $\boldsymbol{W}^{(o)}$ and $\boldsymbol{b}^{(o)} = [b_1^{(o)}, b_2^{(o)}, ..., b_k^{(o)}]$. $\boldsymbol{W}^{(o)}$ is a $k \times m$ matrix, where each parameter $w_{i,j}^{(o)}$ is defined as the weight value associate with the neuron i in the layer o related to the neuron j in the previous layer. The total parameters $\boldsymbol{\theta}$ for the MLP consists of all learnable parameters from each layer $\boldsymbol{\theta}^{(o)}$.

Mathematically, following by the order from the input to the output, the forward pass of the MLP is described as:

$$z_i^{(o)} = \sum_{j=1}^{m} w_{i,j}^{(o)} a_j^{(o-1)} + b_i^{(o)}, \tag{2.3}$$

where $z_i^{(o)}$ is the net input of the neuron i in the layer o, and

$$a_i^{(o)} = f(z_i^{(o)}), \tag{2.4}$$

where $a_i^{(o)}$ is the output of the neuron i in the layer o, activated by the non-linear function $f(\cdot)$.

The final output of the MLP $\boldsymbol{y}$ consists of results given by each neuron i in the last layer, defined as:

$$y_i = f(z_i^{(O)}), \tag{2.5}$$

where O is the number of layers in the MLP.

Figure 2.2 illustrates the structure of a typical MLP which contains 3 input data, 2 hidden layers with 4 neurons in each layer, and 2 outputs.

The layers in a MLP are fully connected, which means every neuron in one layer is connected to all neurons in the next layers, with weights determining the connection strength. Neurons within the same layer do not interconnect. The MLP is theoretically capable of approximating any continuous function by adequate neurons parameters and layers, demonstrating high versatility in function learning [44]. They have been applied to a variety of image-related challenging tasks such as face recognition [46] and handwritten digit recognition [47, 48].

However, the MLP faces challenges, like the substantial number of parameters which may lead to overfitting, especially when limited data are used for training. Its dense connectivity, with each neuron connected to every neuron in next layers, intensifies this issue. Additionally, the MLP is not translationally invariant and struggle with capturing hierarchical features in data. Therefore, the convolutional neural networks, introduced in Section 2.4, have been proposed to address these challenges.

2.2.3 Activation functions

The activation functions are non-linear transformations applied in the computations of the neurons. Without these functions, the computation in neural networks would essentially be reduced to linear combinations of inputs, equivalent to a single affine transformation. The commonly used activation functions in deep learning

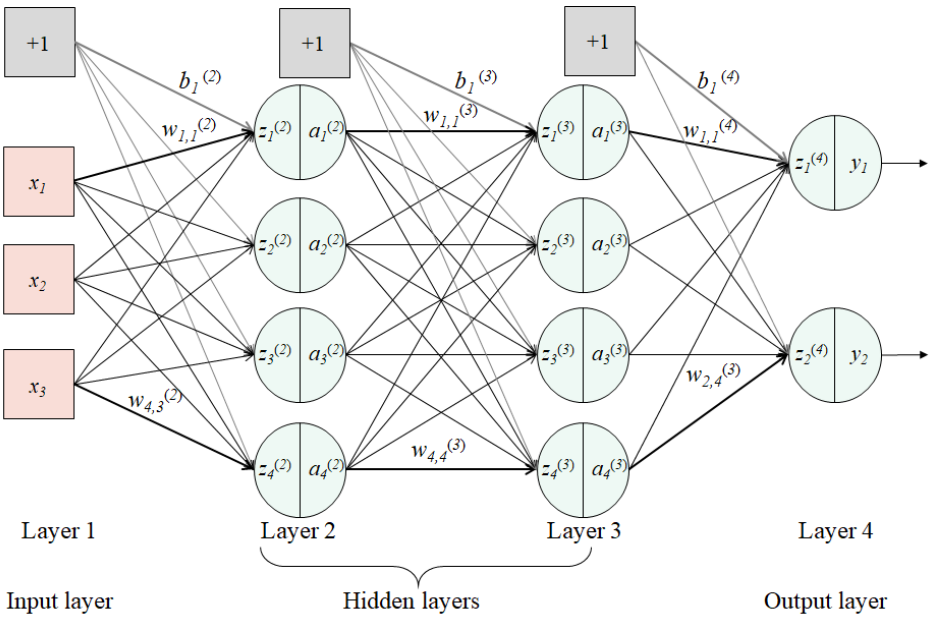


Figure 2.2: Structure of a Multi-Layer Perceptron with 2 hidden layers. Arrows are used to represent the connections that are associated to learnable parameters. Gray arrows represent the bias and the black arrows represent the weights associated to the input. In the figure, the bold arrows are related to the corresponding annotations.

include sigmoid, Hyperbolic Tangent (tanh), REctified Linear Unit (ReLU) and softmax. They are explained as what follows.

Sigmoid

As shown in Figure 2.3.(a), the sigmoid activation function squashes a neuron's net input z, which ranged from $(-\infty, +\infty)$, into $(0, 1)$. The output of sigmoid can be used as the probability, which helps to explain the model's decision.

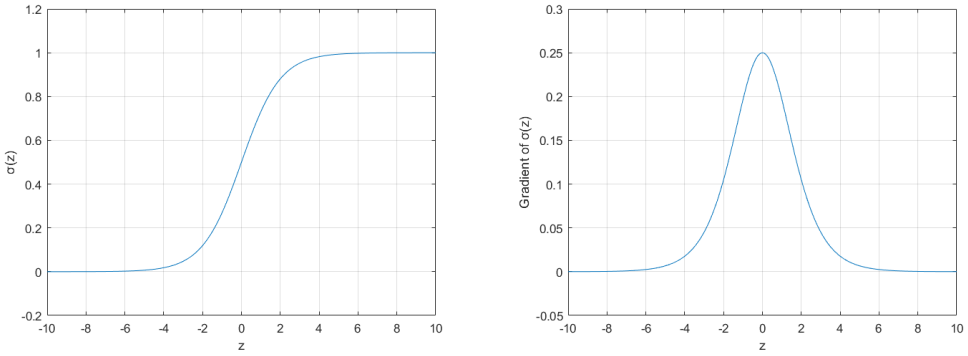
The sigmoid function is defined as:

$$\sigma(z) = \frac{1}{1 + e^{-z}}. \tag{2.6}$$

The gradient of sigmoid can be computed as:

$$\sigma'(z) = \sigma(z)(1 - \sigma(z)). \tag{2.7}$$

Figure 2.3.(b) presents the gradient of sigmoid in response to the input z. It is observed that, the gradient of the sigmoid function reaches the maximum of 0.25 when the input equals 0, which can lead to potential gradient vanishing issues. Besides, the non-zero-centered output of the sigmoid function, leads to gradients that uniformly push weight updates in one direction. This homogeneous gradient



(a) (b)

Figure 2.3: (a) Sigmoid activation function and (b) its gradient.

behavior can cause inefficient zig-zagging paths in the optimization and slow down the convergence process [49].

Tanh

An alternative for the sigmoid function is the tanh activation function, which has zero-centered output and a broader range of positive and negative values. As shown in Figure 2.4.(a), tanh compresses a neuron's net input z ranged from $(-\infty, +\infty)$ into $(-1, 1)$.

The tanh activation function is defined as:

$$\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}. \tag{2.8}$$

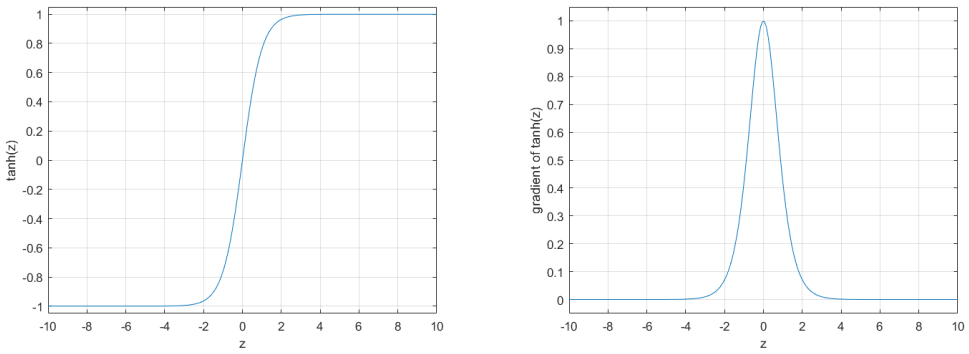
The gradient of the tanh activation function is computed as:

$$\tanh'(z) = 1 - \tanh^2(z). \tag{2.9}$$

The gradient of the tanh activation function is shown in Figure 2.4.(b).

We can find that, both sigmoid and tanh activation functions suffer from the vanishing gradient problem: when the functions outputs approach to -1 or 1, their gradients with respect to the input values approach to 0. This can lead to convergence issues when updating the weights via backpropagation, which is presented in Section 2.3.

Even tanh function does not fully solve the vanishing gradient problem, it improves upon the output range issue of the sigmoid, making it more suitable for certain applications by providing a zero-centered output [50, 51]. A further use for tanh function in our method is presented in Chapter 4.



(a) (b)

Figure 2.4: (a) Tanh activation function and (b) its gradient.

ReLU

As shown in Figure 2.5.(a), ReLU introduces non-linearity by leaving positive values and setting negative values to zero. It compresses a neuron's net input z ranged from $(-\infty, +\infty)$ into $[0, +\infty)$.

The ReLU activation function is defined as:

$$\mathrm{ReLU}(z) = \max(0, z). \tag{2.10}$$

The gradient of ReLU activation function is computed as:

$$\mathrm{ReLU}'(z) = \begin{cases} 0 & \text{if } z < 0, \\ 1 & \text{otherwise.} \end{cases} \tag{2.11}$$

It can be noted from Equation (2.10) that, the ReLU function is not differentiable at the point 0. However, it is conventionally defined that $\mathrm{ReLU}'(0) = 1$ for practical purposes [52].

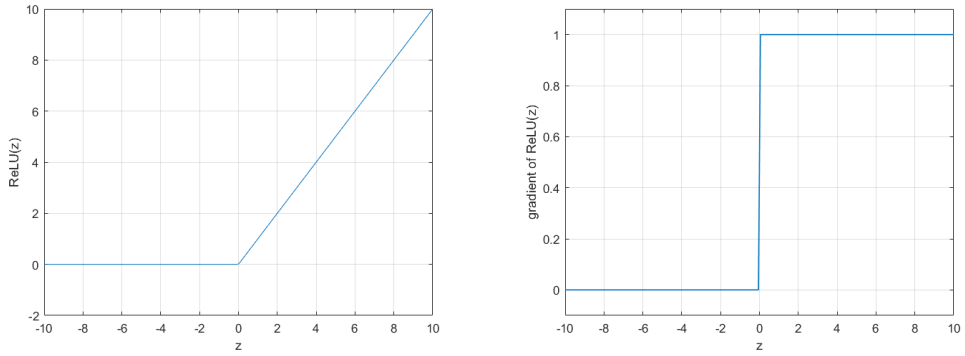
The gradient of ReLU function is shown in the Figure 2.5.(b).

It is observed that, compared to the gradients of sigmoid and tanh functions which reach a smaller value when the input is large, the gradient of ReLU is 1 as long as the input is positive, which is effective in mitigating the vanishing gradient problem.

The ReLU activation function is commonly used in recent WSSS methods for generating activation maps [53], which is introduced in Chapter 3.

Softmax

Different with the activation functions mentioned above, which are applied on the net input of each neuron independently, the softmax activation function is applied



(a) (b)

Figure 2.5: (a) ReLU activation function and (b) its gradient.

across all neurons in the same layer to normalize their net inputs into a probability distribution [54]. Specifically, supposing that $\boldsymbol{z} = [z_1, z_2, ..., z_i, ..., z_k]^\top$ is the vector for all k neurons' net inputs in the same layer and z_i is the net input for the neuron i. The softmax activation function is defined as:

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}}. \tag{2.12}$$

The gradient of softmax activation function is computed as:

$$\frac{\partial \text{softmax}(z_i)}{\partial z_j} = \text{softmax}(z_i)(\delta_{i,j} - \text{softmax}(z_j)), \tag{2.13}$$

where

$$\delta_{i,j} = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{otherwise.} \end{cases} \tag{2.14}$$

The softmax activation function is generally used in the output layer, since the output of softmax(z_i) can be interpreted as the probability of the corresponding class i, with all values summing up to 1 (i.e., $\sum_{j=1}^{k} \text{softmax}(z_j) = 1$).

Activation functions enable the neurons computation to capture complex data structures and improve generalization. By introducing non-linearities, these functions allow for more discriminative and effective categorization of inputs, thereby enhancing the model's overall performance.

2.3 Neural networks learning process

In the previous section, we present the basic components and explore the forward propagation mechanisms of neural networks, detailing the steps in processing input data. In this section, we focus on the learning process. The learning process involves adjusting the learnable parameters, which are the weights and biases of neurons, to achieve specific objectives. The goal of the learning process is to enable the neural networks to accurately and effectively produce appropriate outputs. As mentioned before, we mainly focus on the fully supervised learning case in this chapter to simplify the presentation. In the Chapter 3, we will discuss further about the learning process under different levels of supervision.

2.3.1 Objective

Neural networks aim at mapping the input data from an input space to an output space, with the mapping function $g(\cdot)$, driven by a set of learnable parameters $\boldsymbol{\theta}$. Training a neural network involves adjusting $\boldsymbol{\theta}$ to ensure that for given input $\boldsymbol{x}$, the output of the neural network $\hat{\boldsymbol{y}} = g(\boldsymbol{x}; \boldsymbol{\theta})$ of the neural network with parameters $\boldsymbol{\theta}$ closely matches the true label $\boldsymbol{y}$. This is achieved through minimizing the expectation of a loss function $L(\hat{\boldsymbol{y}}, \boldsymbol{y})$ as a measure of the error between the network's outputs $\hat{\boldsymbol{y}}$ and corresponding labels $\boldsymbol{y}$:

$$\underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, \mathbb{E}[L(\hat{\boldsymbol{y}}, \boldsymbol{y})]. \tag{2.15}$$

The details of loss functions are presented in the Section 2.3.2.

In practice, since we deal with data, the empirical estimator of the expectation, noted $J(\hat{\boldsymbol{y}}, \boldsymbol{y})$, is used:

$$J(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \frac{1}{N} \left(\sum_{i=1}^{N} L(\hat{\boldsymbol{y}}^i, \boldsymbol{y}^i) \right), \tag{2.16}$$

where N is the number of training data in the whole training set, $\hat{\boldsymbol{y}}^i = g(\boldsymbol{x}^i, \boldsymbol{\theta})$ is the output of the neural network for the training data $\boldsymbol{x}^i$, and $\boldsymbol{y}^i$ is its label.

The goal is to minimize the cost function J using optimization methods, also named as optimizers. Section 2.3.3 represents the different choice of optimizers. To drive the explanation, we introduce a straightforward optimizer, the Batch Gradient Descent (BGD) optimizer [55]. This method involves beginning with a first solution for $\boldsymbol{\theta}_0$ corresponding to initial parameter values. Generally, $\boldsymbol{\theta}_0$ is randomly initialized, and then repeatedly adjusted in an effort to locate a minimum of $J(\hat{\boldsymbol{y}}, \boldsymbol{y})$. Reaching the global minimum value is most of the time intractable, but a close solution is expected in term of cost value. The strategy is achieved by adjusting the learnable parameters in the direction opposite to the gradient of $J(\hat{\boldsymbol{y}}, \boldsymbol{y})$.

A learnable parameter in the layer o of the MLP is updated at step t as follows:

$$\theta_{i,j,t}^{(o)} = \theta_{i,j,t-1}^{(o)} - \alpha \frac{\partial J_{t-1}(\hat{\boldsymbol{y}}, \boldsymbol{y})}{\partial \theta_{i,j,t-1}^{(o)}}, \tag{2.17}$$

where i and j represent the position of the parameter in the learnable parameter matrix $\boldsymbol{\theta}$. α represents the learning rate, which can either remain constant or vary over iterations.

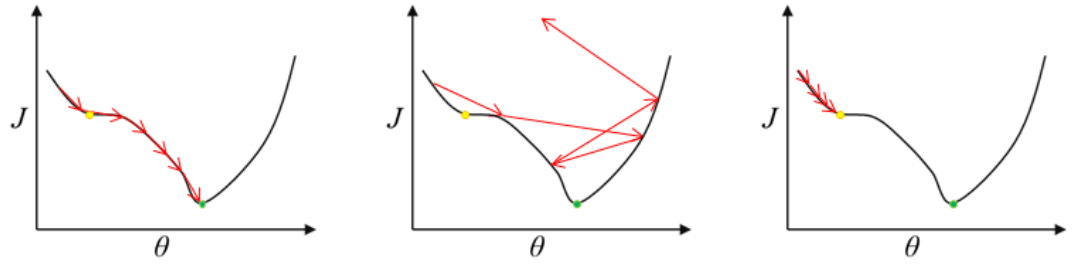
As a critical hyperparameter, the learning rate α significantly influences the convergence. If the learning rate α is set too high, it might prevent the cost function from reaching its lowest possible value and can even diverge, as shown in Figure 2.6.(b). Conversely, if the learning rate is set too low, the network will converge very slowly, which increases the time to find a minimum value that might still appear to be a poor minimal solution, as illstruated in Figure 2.6.(c). The learning rate must be chosen in a way to avoid those two problems, as shown in Fig 2.6.(a).

Specifically, followed by the chain rules, the gradients for the learnable parameter $\theta_{i,j}^{(o)}$ are calculated as follows:

$$\frac{\partial J(\hat{\boldsymbol{y}}, \boldsymbol{y})}{\partial \theta_{i,j}^{(o)}} = \delta_i^{(o)} a_j^{(o-1)}, \tag{2.18}$$

where $z_i^{(o)}$ is the net input for the neuron i in the layer o, and $a_j^{(o-1)}$ is the activated output for the neuron j in the layer $o-1$, $\delta_i^{(o)}$ is the gradient of cost function $J(\hat{\boldsymbol{y}}, \boldsymbol{y})$ to the net input $z_i^{(o)}$ of the neuron i in the layer o. Since these variables are at the same iteration step, the subscript indicating the iteration step has been omitted for simplicity. The following two equations follow the same notation.

In the learning process, the gradients of learnable weights are computed from the final layer back to the first layer, which is known as backpropagation. The $\delta_i^{(O)}$ of



(a) (b) (c)

Figure 2.6: Illustrations for the converges of learning process with different value of learning rate. Convergence (a) with appropriate learning rate, (b) and (c) for too large or too low learning rate, respectively. The yellow circle represents a local minimum value and the green circle represents the global minimum value.

neuron i in the final layer is calculated as:

$$\delta_i^{(O)} = f'(z_i^{(O)}) \frac{\partial J(\hat{\boldsymbol{y}}, \boldsymbol{y})}{\partial \hat{y}_i}, \tag{2.19}$$

where $\hat{y}_i$ is the output of neuron i, and $f'(\cdot)$ is the gradient of activation function $f(\cdot)$ associated to neuron i in layer O.

The value $\delta_i^{(o)}$, where $o \neq O$, is calculated as:

$$\delta_i^{(o)} = f'(z_i^{(o)}) \sum_{j=1}^{u} \theta_{j,i}^{(o+1)} \delta_j^{(o+1)}, \tag{2.20}$$

where u is the number of neurons in the $o+1$ layer.

The two critical hyperparameters in the learning process are the choices of the loss function and the optimizer. The loss functions relate directly to the target task itself, which are introduced in Section 2.3.2. The optimizer determines the model's convergence rate and speed, thereby deciding if the model can successfully update its parameters during the learning process, which is introduced in Section 2.3.3.

2.3.2 Loss functions

Loss functions are used to quantify the disparity between the model's predictions for the training data and the corresponding labels [56]. They are differentiable so that the minimisation can be reached by gradient-based methods. Notice that loss functions are generally associated with the objective of the task. In this section, we introduce 4 typical loss functions: Mean Square Error (MSE), Binary Cross Entropy (BCE), multi-class Cross Entropy and multi-label Cross Entropy loss functions. Among these loss functions, MSE is commonly applied to regression tasks, which focus on estimating a continuous output and modeling the relationship between input variables to predict quantitative values within a continuous range. The 3 other loss functions are widely used for classification, which are aimed at assigning discrete outputs to each data from a set of predefined categories.

Mean Square Error

As one of the most common used loss functions, mean square error is straightforward to understand and compute [57]. It involves calculating the squared differences between model's output $\hat{\boldsymbol{y}}^{\boldsymbol{i}}$ and the label $\boldsymbol{y}^i$ for any training data $\boldsymbol{x}^i$:

$$L(\hat{\boldsymbol{y}}^i, \boldsymbol{y}^i) = \frac{1}{l} \sum_{j=1}^{l} (\hat{y}^{i,j} - y^{i,j})^2, \tag{2.21}$$

where l is the number of elements in the label vector $\boldsymbol{y}^i$. $\hat{y}^{i,j}$ is the value of element j in the model's output vector $\hat{\boldsymbol{y}}^{\boldsymbol{i}}$, and $y^{i,j}$ is corresponding value in the label.

It penalizes larger errors more significantly due to the squaring operation. This character makes it suitable for tasks where large errors should be heavily penalized, such as regression.

Binary Cross Entropy

Binary cross entropy loss function excels in the binary classification task [58], when the label y^i for each training data $\boldsymbol{x}^i$ is a binary variable, i.e., $y^i \in \{0, 1\}$, and the output of the model is a scale value $\hat{y}^i \in [0, 1]$, which is the output activated by the sigmoid activation function. The binary cross entropy loss function is defined as:

$$L(\hat{y}^i, y^i) = -[y^i \log(\hat{y}^i) + (1 - y^i) \log(1 - \hat{y}^i)]. \tag{2.22}$$

Multi-class Cross Entropy

The multi-class cross entropy loss function is well-suited for the multi-class classification task [59], which is aimed at assigning only one class label from the set of predefined classes to each training data. Assuming that l classes are predefined in the dataset, we define $\mathcal{O} = \{1, 2, \cdots, l\}$ as the set of these classes. The label $\mathcal{Y}^i = \{c\} \in \mathcal{O}$ contains only one value c to represent the class related to the training data $\boldsymbol{x}^i$. In practice, the raw class label $\mathcal{Y}^i$ is generally encoded as a one-hot label vector $\boldsymbol{y}^i \in \{0, 1\}^l$. Each element $y^{i,j}$ in the $\boldsymbol{y}^i$ vector is defined as:

$$y^{i,j} = \begin{cases} 1 & \text{if} \quad j = c, \\ 0 & \text{otherwise.} \end{cases} \tag{2.23}$$

For example, considering an image dataset with three predefined classes: human, cat, and mouse. Each image in the dataset is only related to one class. The set of these three classes is defined as $\mathcal{O} = \{1, 2, 3\}$, where 1 represents human, 2 represents cat, and 3 represents mouse. If an image x_i is related to the class cat, its raw label $\mathcal{Y}^i = \{2\}$. The corresponding one-hot label vector $\boldsymbol{y}^i = [0, 1, 0]$.

In this case, the multi-class cross entropy loss function with the encoded one-hot label is defined as follows:

$$L(\hat{\boldsymbol{y}}^i, \boldsymbol{y}^i) = -\sum_{j=1}^{l} y^{i,j} \log(\hat{y}^{i,j}), \tag{2.24}$$

where $\hat{\boldsymbol{y}}^i \in (0, 1)^l$ is the output vector of the model, obtained through a softmax activation function. The element $\hat{y}^{i,j}$ in the vector is the value related to the class j.

Multi-label Cross Entropy

The multi-label cross entropy loss function is designed for multi-label classification, which allows each training data $\boldsymbol{x}^i$ to be classified with one or multiple categories

simultaneously [60]. Assuming that l classes are predefined in the dataset, we define $\mathcal{O} = \{1, 2, \cdots, l\}$ as the set of these classes. In this case, the label $\mathcal{Y}^i \in \mathcal{O}$ can contain one or multiple values for training data $\boldsymbol{x}^i$. Similar to the one-hot label derived by Equation (2.23), a multi-hot encoded label vector $\boldsymbol{y}^i \in \{0, 1\}^l$ for multi-label classification is obtained from the raw class label $\mathcal{Y}^i$. Each element $y^{i,j}$ in the $\boldsymbol{y}^i$ vector is defined as:

$$y^{i,j} = \begin{cases} 1 & \text{if} \quad j \in \mathcal{Y}^i, \\ 0 & \text{otherwise.} \end{cases} \tag{2.25}$$

The mathematical expression of multi-label cross entropy with multi-hot label is defined as follows:

$$L(\hat{\boldsymbol{y}}^i, \boldsymbol{y}^i) = -\frac{1}{l} \sum_{j=1}^{l} \left(y^{i,j} \log(\hat{y}^{i,j}) - (1 - y^{i,j}) \log(1 - \hat{y}^{i,j}) \right) \tag{2.26}$$

where $\hat{\boldsymbol{y}}^i \in (0, 1)^l$ is the output vector of the model, obtained through a sigmoid activation function. The element $\hat{y}^{i,j}$ in the vector is the value related to the class j.

Similarly to the explanation of one-hot label, considering a image dataset with three predefined classes: human, cat, and mouse. Each image in the dataset can be related to multiple classes. The set of these three classes is defined as $\mathcal{O} = \{1, 2, 3\}$, where 1 represents human, 2 represents cat, and 3 represents mouse. If an image x_i is related to the class cat and mouse, its raw label $\mathcal{Y}^i = \{2, 3\}$. The corresponding multi-hot label vector $\boldsymbol{y}^i = [0, 1, 1]$.

Comparing Equation (2.24) to Equation (2.26), it is observed that the nature of multi-label classification loss function is to compute binary cross-entropy for each class predefined in the dataset and average the total loss.

Besides, the mathematical expression of multi-label cross entropy can also be defined by using the raw class label $\mathcal{Y}^i$ as follows:

$$L(\hat{\boldsymbol{y}}^i, \mathcal{Y}^i) = -\frac{1}{l} \left(\sum_{j \in \mathcal{Y}^i} \log(\hat{y}^{i,j}) - \sum_{j \in \mathcal{O} \setminus \mathcal{Y}^i} \log(1 - \hat{y}^{i,j}) \right). \tag{2.27}$$

In fact, Equation (2.26) and Equation (2.27) are mathematically equivalent. They differ only in the form of labels, with one using the raw class labels and the other using one-hot encoded labels. Equation (2.27) stresses the relationship between the raw labels and the loss function. To maintain consistency, we will use this form to present the loss functions in Chapter 4.

2.3.3 Optimizers

As presented in Section 2.3.1, the learning of a neural network is the process of adjusting model's learnable parameters to reduce the value of the cost at each training step. Optimization algorithms (or optimizers) describe the rules for adjusting the learnable parameters in the learning process. In this section, we introduce 3 typical optimization algorithms which are the Stochastic Gradient Descent (SGD) [61], mini-batch Gradient Descent [55] and Adaptive Moment Estimation (Adam) optimizer [62].

Stochastic Gradient Descent

Unlike batch gradient descent optimizer, which computes the gradient of the cost function using the entire dataset as presented in Section 2.3.1, stochastic gradient descent optimizer updates the parameters using only one single training example $\boldsymbol{x}^k$ at each iteration and repeat N times, where N is the number of the training data [61]. The update rule for the learnable parameters by using SGD is defined as follows:

$$\theta_{i,j,t}^{(o)} = \theta_{i,j,t-1}^{(o)} - \alpha \frac{\partial L_{t-1}(\hat{\boldsymbol{y}}^k, \boldsymbol{y}^k)}{\partial \theta_{i,j,t-1}^{(o)}}, \tag{2.28}$$

where $\theta_{i,j,t}^{(o)}$ and $\theta_{i,j,t-1}^{(o)}$ are the learnable parameters at t and $t-1$ step, α is the learning rate, and $L_{t-1}(\hat{\boldsymbol{y}}^k, \boldsymbol{y}^k)$ is the value of loss at computed by the label $\boldsymbol{y}^k$ and the output $\hat{\boldsymbol{y}}^k$ for the training data x^k at $t-1$ iteration.

The main advantages of SGD are its low memory requirements, as it processes one training example at a time, and faster convergence in the large dataset compared with the batch gradient descent method. However, SGD is sensitive to the learning rate and its convergence behaviour can be more erratic due to the random sampling of individual training examples.

Mini-batch Gradient Descent

Mini-batch gradient descent optimizer partitions the training data into smaller subsets or namely mini-batches [55]. M training data, where $M < N$ (N is the number of training data in the whole training set) are processed in each iteration to calculate the cost function's gradient and update the learnable parameters.

The updating process at iteration t is defined as:

$$\theta_{i,j,t}^{(o)} = \theta_{i,j,t-1}^{(o)} - \alpha \frac{\partial J_{t-1}(\hat{\boldsymbol{y}}, \boldsymbol{y})}{\partial \theta_{i,j,t-1}^{(o)}}, \tag{2.29}$$

with

$$J_{t-1}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \frac{1}{M} \left(\sum_{i=1}^{M} L_{t-1}(\hat{\boldsymbol{y}}^i, \boldsymbol{y}^i) \right). \tag{2.30}$$

In fact, this approach benefits in balancing the computational efficiency of stochastic gradient descent, which uses individual data for updating, with the stability of batch gradient descent, which uses the entire dataset, making it well-suited for training on large datasets. Notice that $J(\hat{\boldsymbol{y}}, \boldsymbol{y})$ is an estimator of $\mathbb{E}(\hat{\boldsymbol{y}}, \boldsymbol{y})$. The quality of this estimator depends on the mini-batch size value M. A larger mini-batch size reduces the variance of the gradient estimate, leading to smoother updates and potentially faster convergence. Conversely, a smaller mini-batch size requires less memory and computational resources, and allows for more frequent updates, which can accelerate the training process, but it introduces more noise in the gradient estimates, which might lead to less stable updates.

Adaptive Moment Estimation

The methods mentioned above use the same learning rate α during the whole learning process, which does not account for the varying sensitivity of parameters or the differing gradients across the parameter space and makes it inefficient in the optimization. Adaptive Moment Estimation optimizer overcome this limitation by dynamically adjusting the learning rate of each parameter based on the first and second moments of the gradient, ensuring efficient and fast convergence [62].

Specifically, following by the rules of Adam [62], the first moment m and second moment v are initialized as 0 at the beginning step, i.e., $m_0 = 0$ and $v_0 = 0$. The biased first moment estimate is updated by:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) \frac{\partial J_{t-1}(\hat{\boldsymbol{y}}, \boldsymbol{y})}{\partial \theta^{(o)}_{i,j,t-1}}, \tag{2.31}$$

where β_1 is a pre-set hyperparameter, with common default value $\beta_1 = 0.9$. t is the iteration number. The computation of m_t provides a smoothed estimate of the gradient, reducing noise and improving the stability of update.

The second moment estimate is updated by:

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) \left(\frac{\partial J_{t-1}(\hat{\boldsymbol{y}}, \boldsymbol{y})}{\partial \theta^{(o)}_{i,j,t-1}} \right)^2, \tag{2.32}$$

where β_2 is a pre-set hyperparameter, with common default value $\beta_2 = 0.999$. The computation of v_t captures the variance of the gradient changes, allowing for effectively training in both high and low gradient regions.

Then, the bias-corrected first moment estimate $\hat{m}$ is computed by:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}, \tag{2.33}$$

where β_1^t denotes β_1 to the power t.

The bias-corrected second moment estimate $\hat{v}$ are updated by:

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}. \tag{2.34}$$

where β_2^t denotes β_2 to the power t.

This bias correction step, i.e., Equation (2.33) and Equation (2.34), is to ensure that during the early stages of training, when m and v have not yet accumulated sufficient information, adjustments can be made to correct for this initial bias to reflect the true gradient information accurately and prevent the updates from being too small or too large.

Finally, the learnable parameter $\theta_{i,j,t-1}^{(o)}$ is updated by moving in the direction of the corrected first moment estimate $\hat{m}$, scaled inversely by the square root of the corrected second moment estimate $\hat{v}$, plus a small term ϵ to prevent division by zero, which are defined as follows:

$$\theta_{i,j,t}^{(o)} = \theta_{i,j,t-1}^{(o)} - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}, \tag{2.35}$$

where ϵ is a small value to avoid the denominator being zero.

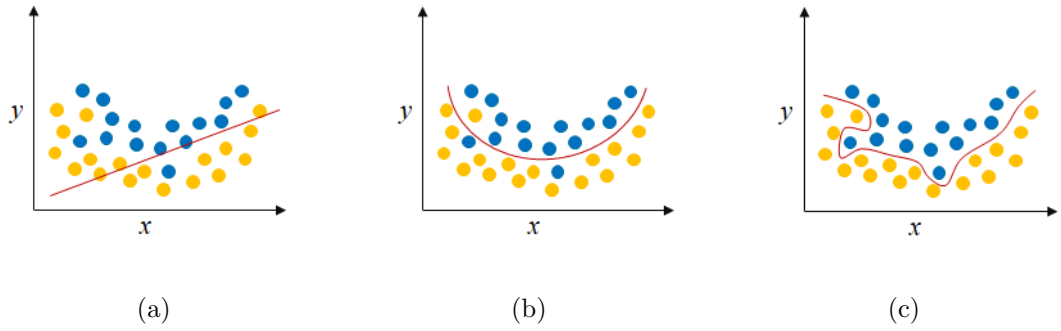
In Adam, one adaptive learning rate is computed for each parameter. This dynamic adjustment of the learning rates based on the magnitude of the gradients improves optimization efficiency and stability.

2.3.4 Training from a pre-trained network

Training the networks from scratch means that the learnable parameters are initialized with no prior knowledge about the data when starting the learning process. It demands significant computational resources and has the risk of over-fitting, especially when the networks has a large number of parameters. Figure 2.7 illustrates the under-fitting, appropriate fitting and over-fitting situations in the model's training. Under-fitting occurs when a model is too simple to capture the underlying structure of the data, resulting in poor performance on both the training data and unseen data. Over-fitting, on the other hand, happens when a model captures noise and random variations in the training data as if they were significant features, leading to excellent performance on the training data but weak generalization ability, i.e., it performs poorly on unseen data.

In order to provide the networks with certain feature knowledge and enhance their generalization ability, it is useful to pre-train the network on a large dataset to learn a good representation of the data, instead of training from scratch. After the pre-training, the model uses the parameters obtained as the initial parameter values for further training.

In practice, there are two typical methods to further update the parameters when training from a pre-trained network. The first one is to freeze all the parameters



(a) (b) (c)

Figure 2.7: Illustrations of (a) under-fitting, (b) appropriate fitting and (c) over-fitting.

in certain layers (typically the early layers that learn general patterns), and only update the parameters in the other layers (which is generally the final layer for classifying). The second one is to update all parameters during the training but with different learning rate. In common practice, a low learning rate is assigned to the early layer, since they are supposed to learn the shallow features which have been trained quite well during pre-training process, and a larger learning rate is given to the late layers to adapt the model for the target new dataset. We use the second one in the experiments of the proposed methods in Chapter 4 and Chapter 5.

2.4 Convolutional Neural Networks

In the previous sections, we introduce the MLP method, which can deal with input data associated with the label. However, the MLP method is not ideally suited for image-related tasks due to its inability to efficiently process image as the input data. This limitation comes from that MLP treats images as flattened one-dimensional vectors, thereby losing the two-dimensional spatial relationships between pixels and failing in capturing crucial features like edges, textures, or corners in the image. When the size of the input image increases, the number of learnable parameters in the MLP also significantly increase which makes the model difficult to converge.

To address these issues, a specialized form of neural networks known as convolutional neural networks (CNN) has been introduced. Tracing their origins to the 1980s, the earliest CNN model, LeNet-5, marked a significant leap in handwritten digit recognition [47], as shown in Figure 2.8.

Similar to the LeNet-5 model, CNN models are distinguished by their hierarchical structure, which is composed of convolutional layers, pooling layers and fully connected layers to work progressively for extracting informative image features. CNN models maintain the spatial hierarchy of images, support translational invariance, and require fewer parameters, making them more efficient for image-related tasks

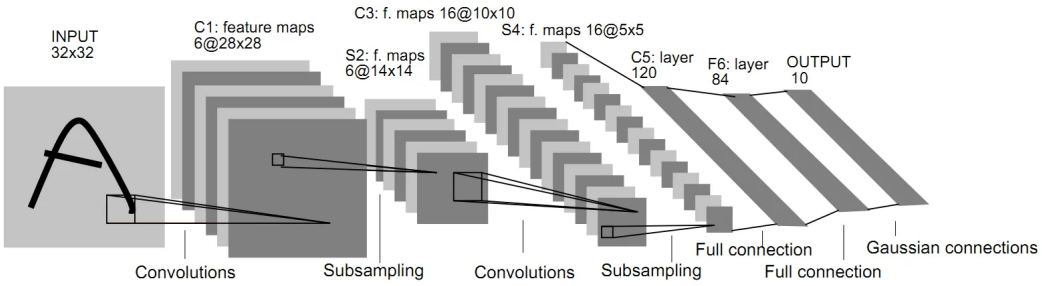


Figure 2.8: Architecture of LeNet-5 [47].

compared to the MLP method. In this section, we introduce the operations in each type of layers and present their purposes in the hierarchical structure of CNN models.

2.4.1 Convolutional layers

The primary foundation of the convolutional neural networks is the convolutional layer. It processes the input data, which is a three-dimensional feature map $\boldsymbol{I}$ representing either the raw image (for the first convolutional layer) or extracted feature information (for the other layers), into a new three-dimensional feature map $\boldsymbol{F}$ with more abstract and informative representation.

A convolutional layer consists of a set of parameters that need to be learned, referred as the convolutional kernels $\mathcal{L} = \{\boldsymbol{K}^1, \boldsymbol{K}^2, ..., \boldsymbol{K}^c\}$. c is the number of convolution kernels in the layer, which determines the output feature's dimension, often referred as the number of channels.

Each kernel $\boldsymbol{K}^p \in \mathcal{L}$ is a three-dimensional filter, which dimension is defined as $h \times w \times d$. h and w are the width and height, and $h \times w$ is generally refereed as the kernel size. d defines the input depth of the kernel, which equals the number of channels for input feature $\boldsymbol{I}$. The convolution operation is applied by a kernel $\boldsymbol{K}^p$ on a specific region, which is named as the receptive field, on the input feature map $\boldsymbol{I}$. When the region of input feature $\boldsymbol{I}$ matches the filter $\boldsymbol{K}^p$, the filter acts as a detector that produces an output value of specific importance. This output value reflects the degree of match between the input feature region and the filter: the better the match, the higher the output value, indicating that a more distinct feature has been detected.

Specifically, the convolutional operation involves element-wise multiplication of the convolutional kernel $\boldsymbol{K}^p$ with the corresponding values in the input feature map $\boldsymbol{I}$ and then aggregation. The mathematical expression of the convolution operation is defined as follows:

$$\boldsymbol{F}^p_{i,j} = (\boldsymbol{K}^p * \boldsymbol{I})_{i,j} = \sum_{m=0}^{h-1} \sum_{n=0}^{w-1} \sum_{q=1}^{d} \boldsymbol{I}^q_{i,j} \boldsymbol{K}^{p,q}_{i+m,j+n}, \tag{2.36}$$

where $\boldsymbol{I}_{i,j}^{q}$ and $\boldsymbol{K}_{i,j}^{p,q}$ is the value in the location (i, j) of the channel q within the input feature $\boldsymbol{I}$ and the kernel $\boldsymbol{K}^{p}$. $\boldsymbol{F}^{p}$ is a two dimensional feature map outputted by the convolution operation related to the kernel $\boldsymbol{K}^{p}$, and $\boldsymbol{F}_{i,j}^{p}$ is the value in the location (i, j) of the $\boldsymbol{F}^{p}$, which is then activated by a non-linear activation function. Each two-dimensional feature map is stacked together to form the final output, which is the three-dimensional feature map $\boldsymbol{F} = [\boldsymbol{F}^{1}, \boldsymbol{F}^{2}, ..., \boldsymbol{F}^{c}]$.

Additionally, the stride s and zero padding are very important hyperparameters as well: The stride s denotes the number of pixels by which the convolution window moves after each operation. Zero padding adds zeros to every side of the input's borders, to increase the use of the edge areas in the input data. Both stride and zero padding determines a model's perception of features and influence the size of output feature map, which is important to detail-sensitive tasks like image segmentation. Figure 2.9 show 3 examples with different stride and zero padding for the same input feature and convolutional kernel.

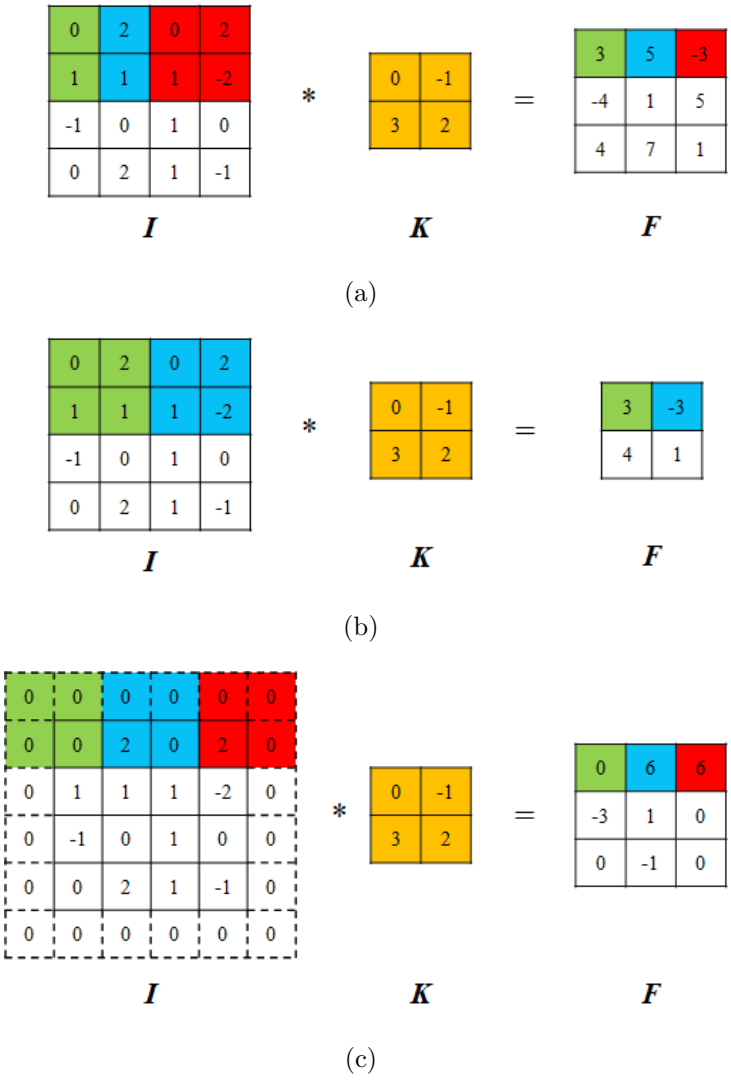


Figure 2.9: Illustration of obtaining result $\boldsymbol{F}$ by convolution operations using 2D input $\boldsymbol{I}$ and kernel $\boldsymbol{K}$. (a) and (b) use stride that is equal to 1 and 2, respectively, without zero padding. (c) uses stride that is equal to 2 with zero padding to enlarge the input size.

As shown in Figure 2.9, we use two-dimensional examples to illustrate the convolution operation for understanding and visualization the core principles of the process with different hyperparameters. The convolution kernel $\boldsymbol{K}$ in the examples is of size 2×2. The size of the original input $\boldsymbol{I}$ is 4×4. Figure 2.9.(a) and Figure 2.9.(b) give examples of the convolution operation without the zero padding but with the stride equal to 1 and 2, respectively. The size of input feature in the example shown in Figure 2.9.(c) is enlarged into 6×6 by zero padding, and the convolution operation is with the stride equal to 2.

A commonly used convolution operation in the segmentation networks is atrous convolution (or namely dilated convolution). It is designed to increase the receptive field of the filter without increasing the amount of parameters, by introducing gaps (zero values) into the convolution kernel. The dilation rate is a parameter to determine the number of zeros added in the kernel. When the dilation rate is equal to 1, it is a standard convolution operation. The larger the rate, the larger the receptive field of the filter becomes. Figure 2.10 presents an example with an atrous convolution kernel, which is derived from the one used in Figure 2.9.(a) by applying a dilation rate equal to 2. Atrous convolution allows for the adjustment of sampling rates, significantly enlarging the receptive field and enabling multi-scale feature extraction.

The intermediate outputs of the network from the convolutional layers can be thought as the intermediate representations of the data learned by the network. [63] visualized feature maps coming from different convolutional layers in different CNN models, which are presented in Section 2.5, as shown in Figure 2.11. We can find that, in early convolutional layers, low-level and class-agnostic geometric representations, such as corner and edge are highlighted, while high-level and discriminative semantic features, such as symbolic patterns are learnt in deep layers.

2.4.2 Pooling layers

Pooling layers in the CNN models are used to selectively reduce the spatial size of the input feature map while retain essential information [67]. The pooling process involves sliding a fixed-size window across the input feature map, outputting a

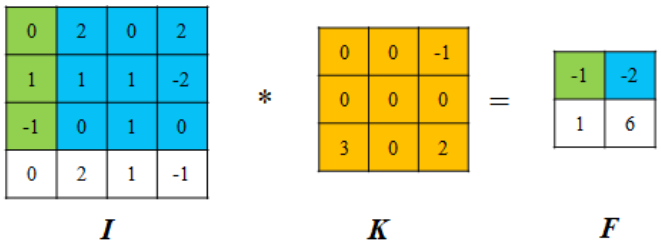


Figure 2.10: Illustration of obtaining result $\boldsymbol{F}$ by applying an atrous convolution operation on the input feature map $\boldsymbol{I}$, with a stride of 1 and no zero padding. The dilation rate of atrous convolution kernel $\boldsymbol{K}$ is equal to 2.

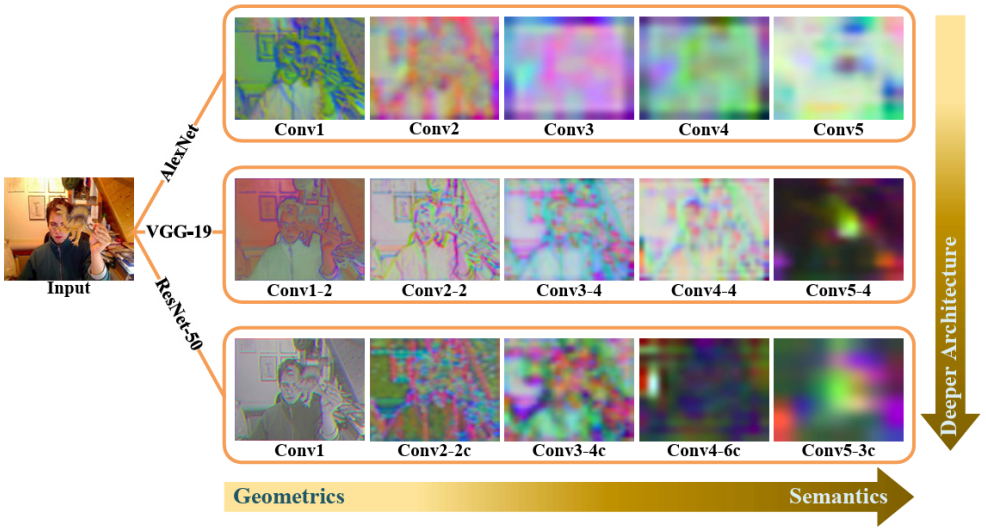
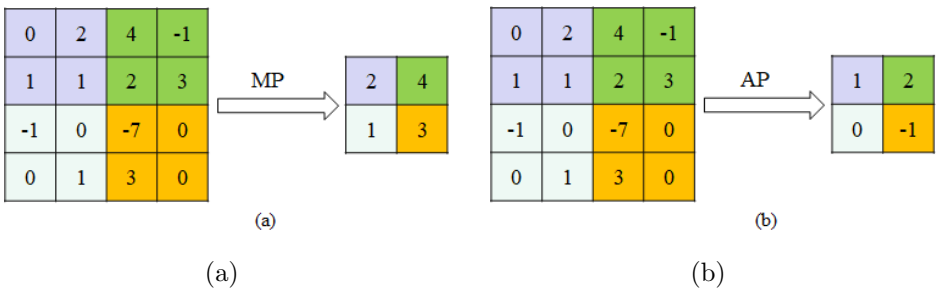


Figure 2.11: Visualization of feature maps from different convolutional layers in different CNN architectures by [63]*, including AlexNet* [64] *(top row), VGG-19* [65] *(middle row) and ResNet-50* [66] *(bottom row).*

single value from each window. The window size is a key parameter in pooling operations. Similar to convolutional operations, pooling operations also have stride and padding parameters, which determine the step size of the window and whether to add zeros to the edges of the feature map. Typically, the stride matches the window size. Two common types of pooling operations are Max Pooling (MP) and Average Pooling (AP). The max pooling is operated by selecting the maximum value, while the average pooling is operated by computing the averaged value within each window slid across the input feature. Figure 2.12.(a) and Figure 2.12.(b) illustrate a max pooling and an average pooling on the same input feature, respectively. The window size is 2×2, the stride equals 2 and padding equal to 0 in both two examples.

When the size of the pooling window equals the size of the input feature, the entire feature map is pooled into a single value. This operation is known as global pooling. Depending on the pooling rule, it can be classified into Global Max Pooling (GMP) and Global Average Pooling (GAP). [68] compares GMP and GAP



(a) (b)

Figure 2.12: Illustration of (a) max pooling and (b) average pooling operations.

operations, and highlights their advantages and disadvantages when processing the feature. GMP emphasizes promoting a high response for a single location, whereas GAP encourages high responses across all locations. Consequently, a segmentation network trained with GMP tends to underestimate object sizes, focusing on a single prominent region, while a network trained with GAP tends to overestimate sizes by distributing attention more evenly across the entire image. We give a further analysis for the pooling processes used in the recent WSSS methods in the Chapter 3.

2.4.3 Fully connected layers

The features processed from the convolution and pooling layers can be flattened as a one-dimensional vector and served as the input to the fully connected layers, which performs a desired task, such as classification. The computation in the fully connected layers resembles to a MLP model, where each neuron receives input from all neurons in the previous layer with weighted summation and then followed by activation functions.

However, fully connected layers often contains a substantial portion of the parameters in the entire network, which increases the risk of overfitting and computational complexity. [10] proposed a fully convolutional network to replace each fully connected layer by a convolutional layer for image segmentation, which is detailed in Section 2.6.1.

2.5 CNN-based classification networks

The target of image classification is to identify the main object or objects within an image from a list of predefined classes. Depending on the number of objects in the image and the number of predefined classes, there are binary image classification, multi-class image classification and multi-label image classification tasks. The difference of these tasks is to assign one class label for the given image from two (binary image classification) or multiple predefined categories (multi-class image classification), or assign one or more class labels from multiple predefined categories (multi-label image classification). Generally, the same image classification networks can be applied to different image classification tasks, requiring only the adaptation of the loss function, introduced in Section 2.3.2. In this section, 3 typical image classification convolutional neural networks are introduced.

2.5.1 AlexNet

The advent of GPUs and the availability of extensive datasets marked a turning point with the introduction of the first modern deep CNN, AlexNet [64], in 2012. As presented in Figure 2.13, with a deeper architecture and more convolutional

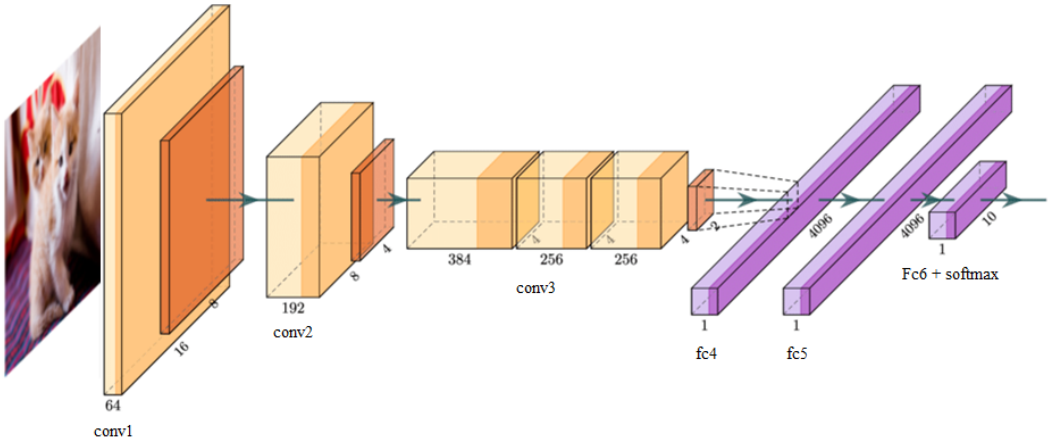


Figure 2.13: Architecture of AlexNet [64]. *Boxes indicate the output feature related to certain operations: Light yellow for convolution operations, darker yellow for a ReLU function, orange for max-pooling, light purple for the fully connected layers, and dark purple for the softmax operation.*

layers compared with LeNet-5 [47], AlexNet was adept at capturing complex features, markedly enhancing performance on challenging datasets. Trained on the ImageNet [69], a large dataset of over 1.2 million high-resolution images, AlexNet was the sate-of-the-art method at its moment, with a top-5 classification error of 15.3% compared with 26.2% in the second one at the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) in 2012. AlexNet's success propelled the widespread adoption of CNNs in computer vision challenges. Subsequently, researchers developed numerous potent deep networks capable of surpassing human-level accuracy across a diverse range of tasks.

2.5.2 Visual Geometry Group (VGG) Networks

In 2014, Karen Simonyan and Andrew Zisserman from Visual Geometry Group proposed a series of CNN architectures named as VGG networks [65]. These networks outperformed AlexNet by achieving greater depth, with configurations of 11, 13, 16, and 19 layers with learnable parameters, respectively. Among them, VGG-16 is the most commonly used one and performs the best in the ILSVRC in 2014 with a top-5 classification error of 7.32%. The architecture of VGG-16 is shown in Figure 2.14.

Specifically, the convolutional kernels in VGGs have the kernel size of 3×3. The authors demonstrated that stacked small convolutional filters yield an equivalent receptive field as a single large convolutional filter but with a reduced number of parameters. This approach aims to enhance generalization capabilities by leveraging increased layer depth. The success of VGG-16 reveals that with an increase in network depth, there is an improvement in classification performance. VGG-16 is also one of the common used models adopted as the backbone classification network in WSSS task.

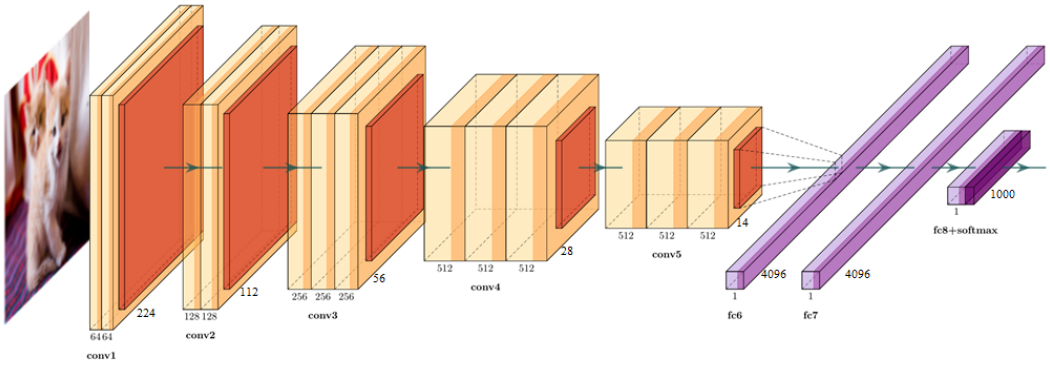


Figure 2.14: Illustration of VGG-16 [65]. *The boxes with colors represent different process, and the numbers above represent the output feature's size after corresponding processes. The input for the VGG-16 is assumed as a RGB image with the spatial size of 224 × 224.*

However, VGG model's 138 million parameters give rise to the issue of exploding gradients. Therefore, ResNets are introduced to prevent exploding gradients problem that occurred in VGG-16.

2.5.3 Residual Networks (ResNets)

The Residual Networks (ResNets) were introduced in 2015 [66]. ResNets use two types of skip connections with 2 and 3 convolutional layers in the main path as illustrated in Figure 2.15.(a) and Figure 2.15.(b), respectively.

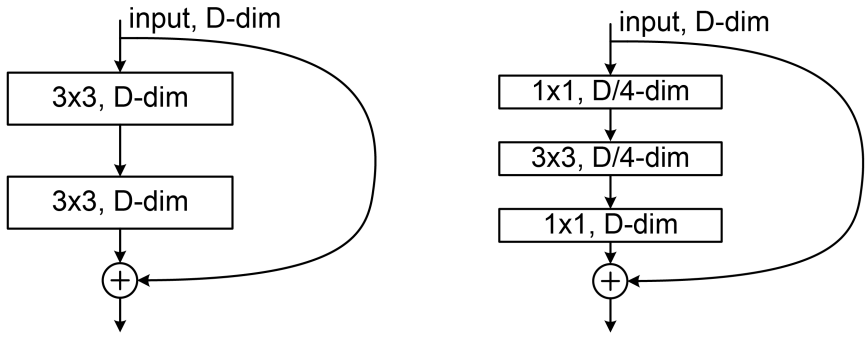


Figure 2.15: Illustration of two types of skip connections used in the ResNets with (a) two and (b) three convolutional layers in the main path [66]. *The notation* Input, D-dim *represents an input feature map with the* D *channels. The* 3 × 3, D-dim *box represents a convolutional layer with convolutional kernels' size of 3 × 3 and the number of kernels in the layer is* D. *For* 1 × 1, D-dim, *and* 1 × 1, D/4-dim *boxes in the figure, the meanings are similar.*

In the skip connections, the input is added with the output of the final convolution layer. The skip connections are proposed to perform identity mapping between several stacked layers. This approach provides the direct flow of gradients through these skip connections, mitigating the vanishing gradient issue and enabling the effective training of deeper networks.

Table 2.1 represents the initial five ResNet structures, comprising 18, 34, 50, 101, and 152 layers with learnable parameters, respectively. ResNets have been shown to achieve state-of-the-art results on a variety of computer vision tasks. ResNet-50 won the ILSVRC in 2015 with a top-5 error rate of 3.57%. By comparing their structures, it can be seen that these models all ultimately classify outputs through a 1000-dimensional fully connected layer. Aside from the fully connected layer, the architecture can be divided into five stages, named conv1, and conv2_x to conv5_x. The initial stage, namely conv1, employs a single convolutional operation and the convolution kernels have the kernel size of 7×7. The convolution kernels in the subsequent stages of ResNet-18 and ResNet-34 are in the kernel size of 3×3. Both ResNet-18 and ResNet-34 use short connection with 2 convolutional layers in the main path, as shown in Figure 2.15.(a), in each stage. For ResNet-50, ResNet-101 and ResNet-152, they use the skip connections with 3 convolutional layers in the main path, as shown in Figure 2.15.(b), and the convolution kernels in the subsequent stages are in the kernel size of 1×1 or 3×3 .

It is worth mentioning that, different with the initial ResNets designed for image classification tasks, ResNet-38 is an advanced variation tailored specifically for semantic segmentation task [70]. The key adjustments within ResNet-38 include optimizing layer configurations to enhance feature extraction and incorporating techniques such as atrous convolutions, which has been introduced in Section 2.4.1. These adjustments ensure that ResNet-38 maintains a balance between depth and computations to work efficiently in the semantic segmentation task.

In recent WSSS methods, ResNet-50 is commonly adopted as the backbone classification networks. ResNet-38 and ResNet-101 can be used as the backbone for segmentation models, like DeepLabV2 [14] which is introduced in the next section.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Table 2.1: The ResNets achitectures [66]. *FLOPs are the floating point operations.*

2.6 CNN-based segmentation networks

The target of image segmentation is to assign a predefined label for each pixel in the image. Therefore, the image segmentation is also considered as pixel-wise classification problem for the whole image. As illustrated in Figure 2.16, image segmentation can be broadly categorized into binary segmentation, semantic segmentation, and instance segmentation. Binary segmentation divides an image into foreground objects and background, suitable for simple scenes [71]. Semantic segmentation divides an image into various categories without distinguishing between different instances of the same category, widely used in applications like autonomous driving [5, 6, 7], medical image diagnostics [3] and satellite imagery analysis [9]. Instance segmentation, the most complex, identifies individual instances of each category within an image, crucial for detailed analysis in video surveillance [72] and augmented reality [73, 74]. In this thesis, we mainly focus on semantic segmentation.

Before implementing CNN for semantic segmentation, traditional methods [75, 76, 77] largely relied on hand-crafted features [78, 79] which depended on explicit programming to identify textures, shapes, and edges. These features often result in inaccurate segmentation outputs when facing complex scenes [80, 81].

CNN models automatically learn feature representations directly from the data. This shift enabled models to capture intricate patterns and distinctions in images, significantly enhancing segmentation accuracy [10, 11, 12, 13, 14]. As we mentioned in Section 2.4, the key to CNNs' success lies in the ability to learn hierarchical features through multiple layers of convolutions and nonlinearities. This process allows CNNs to understand both low-level details (such as edges and textures) and high-level semantic information (such as the presence of specific objects), improving the precision and reliability of semantic segmentation across diverse scenarios.

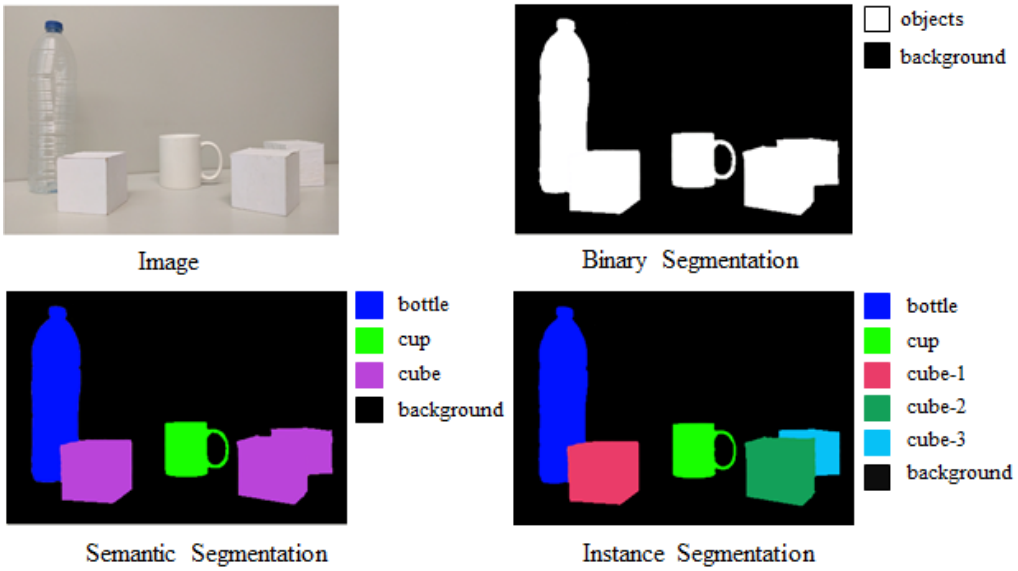


Figure 2.16: Illustration of different segmentation tasks adapted from [82].

2.6.1 Fully Convolutional Network

The Fully Convolutional Network (FCN) [10] stands as a pioneering architecture in the field of semantic segmentation. It introduces an end-to-end, pixel-to-pixel approach, effectively enabling the direct predictions of segmentation maps from images, as shown in Figure 2.17.

One of the critical innovations within FCN is the use of convolution operation with the kernel size of 1×1. Traditionally, convolutional layers are used to extract image features, while fully connected layers are employed to perform classification. However, fully connected layers lose spatial information, which is necessary in semantic segmentation. The convolution operation with the kernel size of 1×1 acts as an alternative to perform class-specific channel-wise feature learning while preserving spatial dimensions. Specifically, FCN uses the VGG-16 model as its backbone network but replaces all fully connected layers by convolutional operations with the kernel size of 1×1. This modification results in 2-dimensional class-wise predictions instead of 1-dimensional class scores. This replacement is the key factor in the FCN's ability to predict semantic labels for every pixel in an image, setting a new standard for semantic segmentation task and inspiring subsequent innovations in the field. When using the mean Intersection-of-Union (mIoU) evaluation metric (this metric will be detailed in Section 3.3.3), the VGG16-based FCN segmentation model finally reaches 62.2% score [10] on the test set of the PASCAL VOC 2012 dataset [18].

When taking VGG-16 as the backbone network, the spatial size of the output from the last convolutional layer is 1/32 compared with the original image. Only using simple upsampling process, the resized outputs are quite coarse, leading to the loss of many fine details. To address this issue, the authors introduced skip connections strategy, presented as the blue lines in Figure 2.17. Finer-scale predictions are obtained after the third and fourth pooling layers, and then summed for the recovery of fine-grain details. These skip connections effectively combine coarse and fine predictions to reconstruct denser segmentation results.

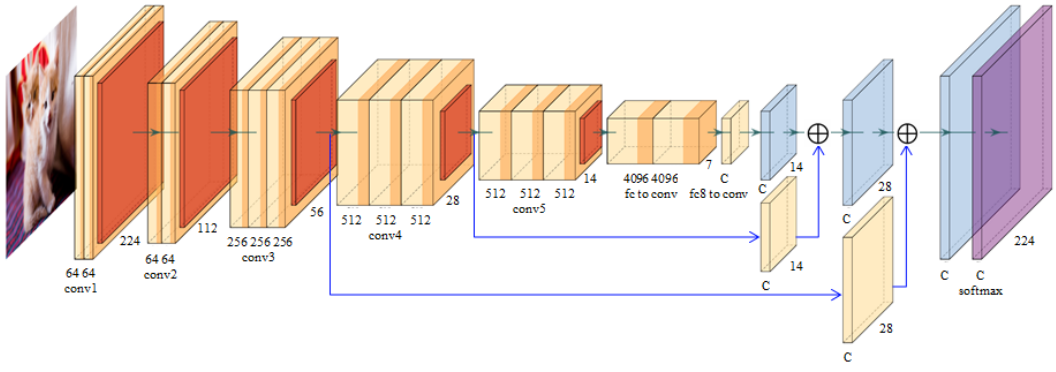


Figure 2.17: The architecture and skip connections in FCN [10]. *Light blue boxes are the resized output features while the purple box indicates the final output processed by the softmax operation. The* $\bigoplus$ *symbol represents the feature summation process.*

FCN marks the first model where CNNs are applied to segmentation task, resulting in a significant improvement. It demonstrates that, by replacing the fully connected layers with convolutional operations with the kernel size of 1×1, an image classification network that originally outputs one-dimensional classification results can be transformed into an image segmentation network capable of producing two-dimensional segmentation masks. Besides, the skip connection strategy is helpful to recover the details and avoid too coarse predictions, which has provided valuable inspiration for designing the architecture of networks to use high-resolution features generated in early layers, such as U-Net [11] and Feature Pyramid Network (FPN) [12] which are introduced in the next section.

2.6.2 U-Net and Feature Pyramid Network

U-Net [11] and Feature Pyramid Network (FPN) [12] are two typical architectures which are designed to leverage high-resolution shallow features to improve the final output. These models highlight that accurate results require not only high-semantic deep features but also the integration of multi-features across the network to preserve spatial details.

U-Net, as shown in Figure 2.18, is known for its symmetric encoder-decoder structure, effectively combines high-semantic features from deep layers with the high-resolution features from early layers through skip connections [11]. This approach ensures that the final segmentation map retains both the context captured by deep layers and the details captured by shallow layers. Aurthos in [83] implement the U-Net based on ResNet-101 [66] backbone network, which finally reaches 65.4% mIoU score on the test set of the PASCAL VOC 2012 dataset [18].

Similarly, the FPN model proposed a building block to progressively refine the segmentation results at multiple scales from deep to shallow features [12], as shown in Figure 2.19. This architecture works efficiently in handling objects of varying sizes, ensuring that each level of the pyramid contributes to a composite understanding of the scene, from broad semantic understanding to fine-grained segmentation details.

In our task, WSSS often relies on weak supervisions, which do not provide the detailed location information needed to fully exploit their architectural advantages in U-Net and FPN. However, the architectural advantages of U-NET and FPN models provide us with insights to use multi-features from deep to shallow layers and inspire the proposed method in Chapter 5.

2.6.3 DeepLab networks

The DeepLab networks comprise a series of models designed to address the challenges of semantic segmentation and achieve accurate segmentation results. Through continuous innovation and architectural optimizations, these models have consistently improved the accuracy and efficiency of image segmentation. In the DeepLab series, there are DeepLabV1 [13], DeepLabV2 [14], DeepLabV3 [15],

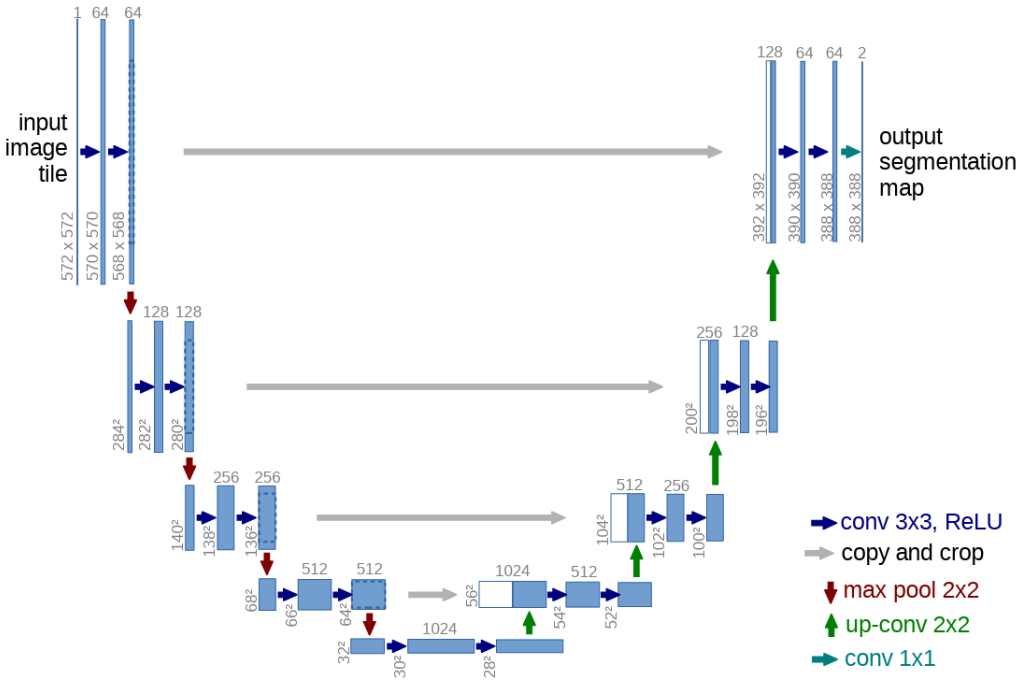


Figure 2.18: The architecture of U-Net [11].

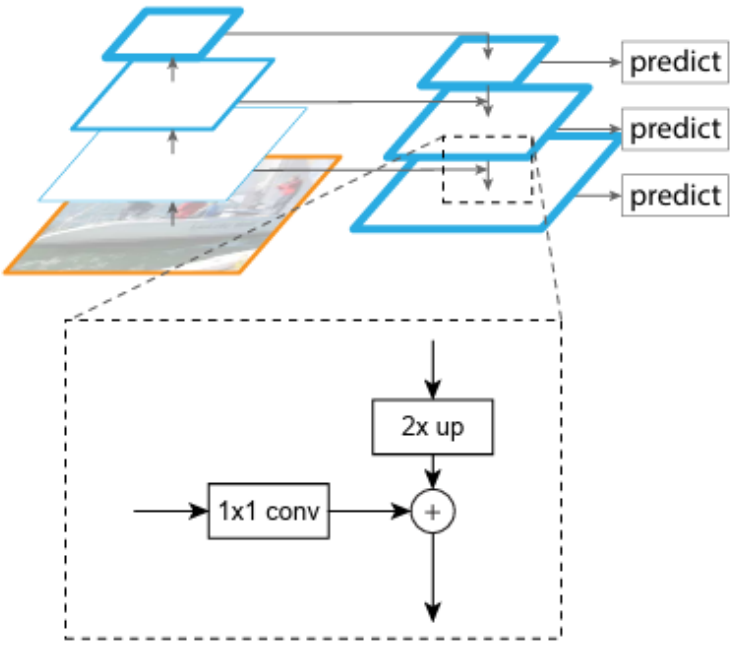


Figure 2.19: The building block proposed by [12] *to use multi-feature from deep to shallow.*

and DeepLabV3+ [16]. Considering that DeepLabV1 and DeepLabV2 are the two most commonly used segmentation models in recent WSSS methods, we provide a detailed presentation of these two models in this section.

DeeplabV1

As illustrated in Figure 2.20, DeepLabV1 is the first model in DeepLab networks to address the critical challenges of signal downsampling and spatial invariance that limited the performances of early CNN-based semantic segmentation model [13]. These challenges, stemming from repeated pooling process, led to reduced resolution and compromised spatial accuracy, hindering the network's ability to make precise predictions of the objects. DeepLabV1 tackled these issues by introducing two key innovations: atrous convolution, which has been introduced in Section 2.4.1, and dense Conditional Random Field (dCRF). dCRF refines segmentation results by accurately capturing fine details and edges from image. Since dCRF is commonly used in the WSSS methods, a detailed explanation is given in Chapter 3. The combination of atrous convolution and dCRF not only boosted the model's accuracy, but also ensured efficient computation. [13] uses VGG-16 as the backbone network to implement DeepLabV1, which reaches 71.6% mIoU score in the test set of the PASCAL VOC 2012 dataset.

DeeplabV2

DeepLabV2 [14], advancing from DeepLabV1, incorporates the Atrous Spatial Pyramid Pooling (ASPP) module to enhance semantic segmentation by addressing scale variability and improving feature resolution, as illustrated in Figure 2.21. ASPP applies atrous convolution at multiple scales in parallel to capture multi-scale context without increasing computational complexity. It enables the model to capture contextual information, addressing the challenges of object size diversity in segmentation tasks. Another significant upgrade in DeepLabV2 is replacing the VGG-16 [65] by ResNet-101 [66] as its backbone network, further boosting its ability to handle complex segmentation scenarios. The ResNet-101 based DeepLabV2 model reaches 79.7% mIoU score in the test set of the PASCAL VOC 2012 dataset.

The evolution from DeepLabV1 to DeepLabV2 highlights significant improvements

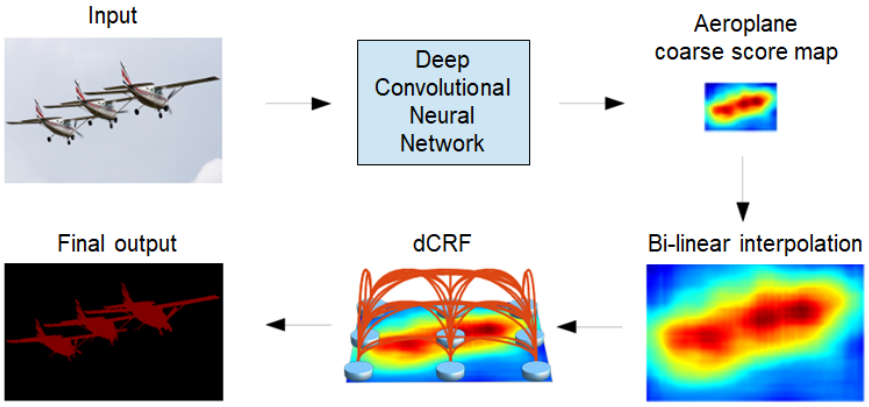


Figure 2.20: The pipeline of DeepLabV1 [13].

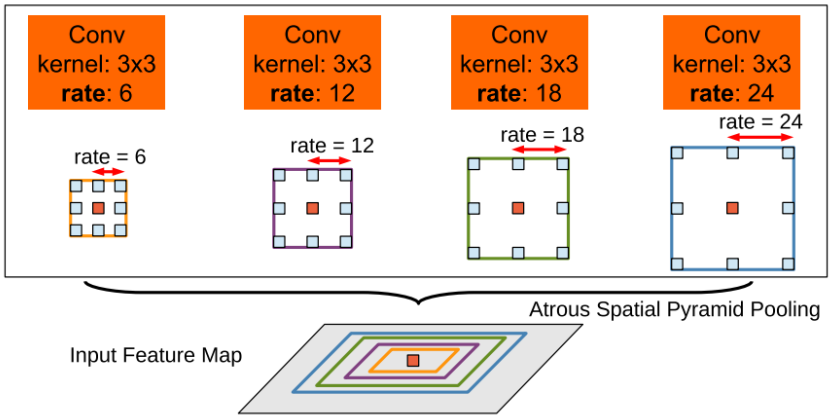


Figure 2.21: An illustration of Atrous Spatial Pyramid Pooling module in DeepLabV2 [14]. *In the figure,* rate *is the dilation rate to determine the number of zeros added in the kernel (see Section 2.4.1).*

in two critical aspects that impact semantic segmentation: the resolution of the features used and the backbone network architecture. Using the features in high-resolution with sufficient semantics minimizes blurring and errors during interpolation for more accurate segmentation results. Additionally, the adoption of a deeper network architecture allows for the extraction of richer semantic information, which is crucial for understanding complex image content. These advancements offers valuable insights when designing our methods to purse high-quality segmentation results in WSSS task. Both DeepLabV1 and DeepLabV2 are used in the experiments of Chapter 5 as the segmentation models to make fair comparison with recent WSSS methods.

2.7 Conclusion

In this chapter, we present an overview of neural networks and convolutional neural networks and emphasized their relevance to this thesis. Specifically, we start with the introduction of basic building components, and then explained the forward pass and learning process of neural networks. Next, convolutional neural networks are introduced. They perform effectively in the image-related tasks, likes image classification and segmentation, thanks to their hierarchical architectures. Besides, we give typical CNN-based classification and segmentation models, and present their evolution to uncover key insights in the corresponding tasks. These insights serve as valuable references for designing the proposed methods in the thesis. In addition to explain the background knowledge, we also highlight how they connect with the methods proposed in this thesis.

Chapter 3

Related works

Contents

3.1 Introduction 39

3.2 Semantic segmentation under different levels of supervision 40

3.2.1 Fully supervised semantic segmentation 40

3.2.2 Unsupervised semantic segmentation 40

3.2.3 Semi-supervised semantic segmentation 41

3.2.4 Weakly supervised semantic segmentation 41

3.3 Overview of the two-stage training pipeline for WSSS 42

3.3.1 The generation of CAM 44

3.3.2 Obtaining pseudo-masks from CAM 47

3.3.3 Training a segmentation model with pseudo-masks . . . 49

3.4 Recent methods in WSSS 52

3.4.1 Generating enhanced CAM 52

3.4.2 Obtaining improved pseudo-masks 56

3.4.3 Adaptive training strategies for the segmentation models 59

3.5 Auxiliary cues for WSSS 59

3.6 Conclusion 61

3.1 Introduction

This chapter presents works related to weakly supervised semantic segmentation. Section 3.2 introduces in detail semantic segmentation under different levels of supervision, which are fully supervised, unsupervised, semi-supervised and weakly supervised semantic segmentation. For weakly supervised semantic segmentation, among all forms of weak supervision, the image-level labels require the least annotation time but provide the weakest location information. In the thesis, we

mainly consider weakly supervised semantic segmentation using only image-level labels. Section 3.3 draws an overview of two-stage training pipeline for WSSS, with the analysis of main limitations in each step. Next, Section 3.4 is devoted to recent methods focused on improving the segmentation performance for WSSS. We present the advantages and challenges of those methods, and draw the links to the proposed methods in Chapter 4 and Chapter 5. Besides, auxiliary cues likes saliency and superpixel are introduced in section 3.5, since they are widely used in recent WSSS methods and substantially influence the final segmentation results. Finally, Section 3.6 gives a conclusion for this chapter.

3.2 Semantic segmentation under different levels of supervision

The goal of semantic segmentation is to assign a predefined class label to each pixel in images, as introduced in Chapter 2. Depending on the levels of supervision, i.e., the labels in the training dataset, it can be roughly categorized into four types: fully supervised semantic segmentation, unsupervised semantic segmentation, semi-supervised semantic segmentation, and weakly supervised semantic segmentation, which are explained as follows.

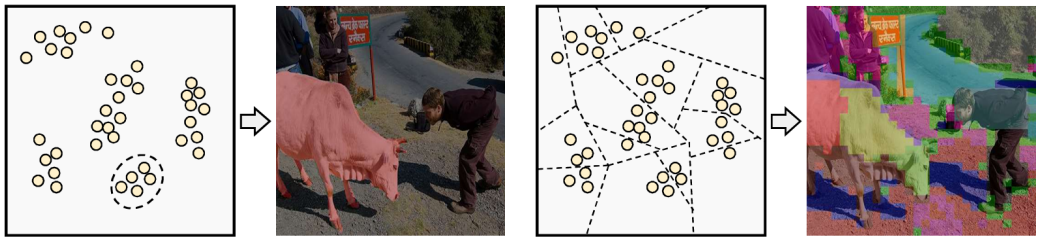
3.2.1 Fully supervised semantic segmentation

Fully supervised semantic segmentation is proposed to train a segmentation model with images labeled by pixel-wise annotations [1]. Formally, the training dataset which contains m labeled images for FSSS is defined as $\mathcal{D} = \{(\boldsymbol{I}^1, \boldsymbol{Y}^1), ..., (\boldsymbol{I}^m, \boldsymbol{Y}^m)\}$, where $\boldsymbol{Y}^i$ is all pixel-wise annotations for the image $\boldsymbol{I}^i$, with $\boldsymbol{Y}^i = \{y^{i,1}, ..., y^{i,k_i}\}$ and $\boldsymbol{I}^i = \{p^{i,1}, ..., p^{i,k_i}\}$. Each pixel $p^{i,j}$ in image $\boldsymbol{I}^i$ can only be labeled as one class in the corresponding pixel-wise annotation $y^{i,j}$, where $j \in \{1, ..., k_i\}$ and k_i is the number of total pixels in image $\boldsymbol{I}^i$. The supervised learning and typical segmentation models [10, 11, 13, 14, 15, 16] are detailed in Chapter 2.

3.2.2 Unsupervised semantic segmentation

Unsupervised semantic segmentation aims at grouping pixels for the images into meaningful semantic segments without using any annotations [84]. Formally, the training dataset that contains m unlabeled images for USS is defined as $\mathcal{D} = \{\boldsymbol{I}^1, ..., \boldsymbol{I}^m\}$, where $\boldsymbol{I}^i$ is a training image without any labels.

Recent USS methods [85, 86, 87, 88] firstly learn pixel-wise visual features by using methods based on Transformer [89, 90], and then applying clustering algorithms, like K-means [91], on the features to generate segments. However, this paradigm is challenging because, in the absence of any annotations, the learned pixel-wise features might not have a clear and direct correlation to the semantic information



(a) (b)

Figure 3.1: Illustration of (a) under-clustering and (b) over-clustering given by [85].

in the images, which may not always be perfectly suitable for clustering. Besides, the number of segments for different images varies and is a crucial hyperparameter in related methods, which can significantly impact the results: setting this number too low may lead to under-clustering, as shown in Figure 3.1.(a), while setting it too high can result in over-clustering, as shown in Figure 3.1.(b).

3.2.3 Semi-supervised semantic segmentation

Semi-supervised semantic segmentation is proposed to alleviate the annotations burden of FSSS by leveraging on a limited tiny amount of labeled images in conjunction with a large amount of unlabeled images [92]. Formally, the dataset which contains l images labeled in pixel-wise and $m-l$ unlabeled images for SSSS is defined as $\mathcal{D} = \{(\boldsymbol{I}^1, \boldsymbol{Y}^1), ..., (\boldsymbol{I}^l, \boldsymbol{Y}^l), \boldsymbol{I}^{l+1}, ..., \boldsymbol{I}^m\}$, with $l \ll (m-l)$, and $\boldsymbol{Y}^i$ is all pixel-wise annotations for the image $\boldsymbol{I}^i$.

Most state-of-the-art SSSS methods [93, 94, 95, 96, 97] are based on the pseudo labeling scheme. Pseudo-masks for the unlabeled images are obtained based on the predictions given from a segmentation model trained by the labeled images. Then, images labeled by the pseudo-masks along with the tiny set of labeled image in the dataset are used to train a segmentation network. One main challenge of these methods is that, since unlabeled images constitute a large proportion in the dataset (i.e., $l \ll (m-l)$), the quality of pseudo-masks heavily determines the final segmentation performance. Recent methods [98, 99] have been proposed to address this issue by refining the generated pseudo-masks before using them as training targets for unlabeled images. For example, [98] focuses on exploring semantic features in the unlabeled images based on a cross-attention mechanism [100], while method in [99] uses random rotations, flips, and translations to augment the data and applies consistency regularization to ensure stable predictions under these transformations. These processes reduce the uncertainty and increase the accuracy of pseudo-masks, making them more reliable for training the segmentation network.

3.2.4 Weakly supervised semantic segmentation

Besides the semi-supervised semantic segmentation approaches, the other group of methods to reduce the pixel-wise annotations burden of FSSS is weakly supervised

semantic segmentation (WSSS), which has been proposed using weak supervision, such as points [30, 31], scribbles [32, 33], bounding boxes [34, 35] and image-level labels [20, 21, 22, 23, 24, 25, 26, 27, 28, 29]. Figure 3.2 provides full supervision and forms of weak supervision for semantic segmentation, arranged in descending order of estimated annotation time [30]. Specifically, as shown in Figure 3.2, scribbles provide sets of pixels that are labeled with classes by drawing continuous curves [33]. Bounding boxes indicate the presence and location of objects by enclosing each object in a rectangular box, but they are not precise in defining the exact shape of the object. Labeled points are defined as a few discrete annotated pixels within objects.

Among all forms of weak supervision, the image-level labels are the weakest supervision which provide the weakest location information, but also require the least annotation time. An image-level label only describes each image by the classes in the image, without offering any location information for the objects. Training images with image-level labels are easily to obtain since a large number of public datasets with such labels are accessible, e.g., PASCAL VOC 2012 [18], ImageNet [69], Microsoft Common Objects in Context (MS COCO 2014) [101] and CUB-200-2011 [102]. In this thesis, we mainly focus on this approach.

Formally, the dataset which contains m images labeled at image-level for WSSS is defined as $\mathcal{D} = \{(\boldsymbol{I}^1, \mathcal{Y}^1), ..., (\boldsymbol{I}^m, \mathcal{Y}^m)\}$, where $\mathcal{Y}^i$ is the image-level label of the image $\boldsymbol{I}^i$. Since there can be multiple object classes in the image $\boldsymbol{I}^i$, the number of classes contained in the label $\mathcal{Y}^i$ can also be multiple. Assuming that l classes are labeled in the dataset, $\mathcal{O} = \{1, 2, ..., l\}$ is defined as the set of all predefined classes. The image-level label $\mathcal{Y}^i \in \mathcal{O}$.

3.3 Overview of the two-stage training pipeline for WSSS

To learn semantic segmentation models with only image-level labels as supervision, existing approaches can be roughly categorized by using one-stage training pipeline or two-stage training pipeline [17]. Approaches [20, 21, 22, 23, 24] using one-stage training pipeline often establish an end-to-end framework with constrained strategies for optimization. Typically, the one-stage training pipeline has two parallel

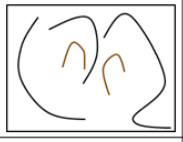
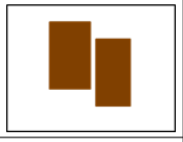
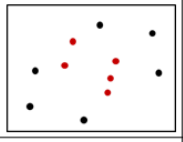
Data	Full supervision	Weak supervision			
Image	Pixel-level label	Scribbles	Bounding boxes	Labeled points	Image-level label
					
Annotation time per image (seconds):	240.0	34.9 (per scribble)	33.8 (per box)	22.1 (per point)	20.0

Figure 3.2: A comparison of annotation times given by [30].

branches, which are a classification branch and a semantic segmentation branch. The classification branch is trained by using image-level labels and provides class-wise features to generate pixel-level annotations, while these annotations are used as supervision for the training in the segmentation branch. During training, the backbone networks used in these two branches are updated simultaneously [21]. However, one significant drawback of these approaches is that the segmentation accuracy falls behind [23, 24].

Similar to the semi-supervised semantic segmentation methods, approaches in weakly supervised semantic segmentation can also follow a two-stage training pipeline with pseudo labeling scheme [28, 29, 36, 37, 38, 39, 40, 103, 104, 105], as illustrated in Figure 3.3. The first stage aims to produce pseudo-masks by only using image-level labels. This stage contains 2 steps: the generation of Class Activation Map (CAM), and obtaining pseudo-masks from the CAM, as presented in Section 3.3.1 and Section 3.3.2, respectively. Since there is no full supervision available in WSSS, in the second stage, pseudo-masks are used to replace the ground truth as the training targets to train a fully supervised segmentation model, as presented in Section 3.3.3. Compared with the one-stage training pipeline, the two-stage pipeline trains first the classification model and then the segmentation model. Learning with the two-stage training pipeline is the dominant approach for recent WSSS methods [17] and also the one concerned in this thesis. We detail the process in each step and offer an analysis of their limitations in the next subsections.

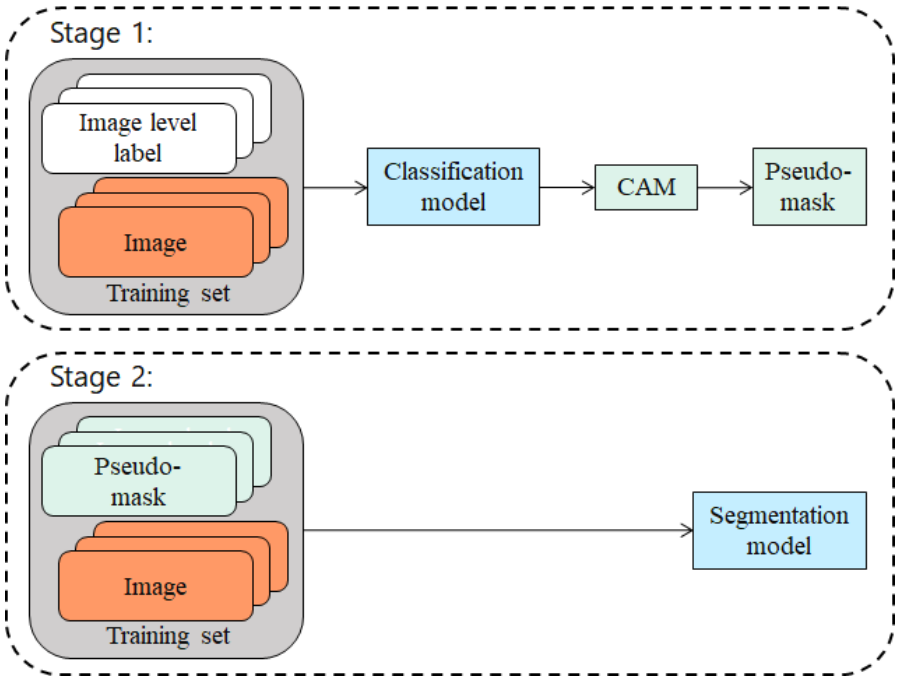


Figure 3.3: Overview for the two-stage training pipeline of the weakly supervised semantic segmentation with only image-level labels.

3.3.1 The generation of CAM

Class Activation Map provides location information for the objects in the images, which is the start point to tackle the WSSS task. Values in CAM are related to the importance of corresponding pixels to identify the presence of a certain class in the image. The generation steps of CAM are as follows:

Obtaining the class-wise feature map

As presented in Section 2.5, trained with image-level labels, a classification network can produce a class score $\boldsymbol{s}$ for an input image. However, the class score $\boldsymbol{s}$ only indicates the relevance of the image to certain classes but does not provide objects locations in the image. In order to extract the class wise feature which can reveal objects location from the classification network, method in [53] is proposed using a classification training pipeline as illustrated in Figure 3.4.

Specifically, the backbone network in Figure 3.4 is a fully convolutional network, which consists of only convolutional and pooling layers without fully connected layers, as presented in Section 2.6.1. The backbone network can be any classification models, such as VGG-16 [65] or ResNet-50 [66]. It acts as a feature extractor which produces the feature map $\boldsymbol{F} \in \mathcal{R}^{h \times w \times n}$ for an input image $\boldsymbol{I} \in \mathcal{R}^{H \times W \times 3}$, where n is the number of channels in the feature $\boldsymbol{F}$. Notice that, in the previous sections, $\boldsymbol{I}_i$ and $\mathcal{Y}_i$ are used to represent a specific image and its corresponding image-level label in the training set. In the following sections, for the sake of clarity and specific explanations, we omit the subscript and use $\boldsymbol{I}$ and $\mathcal{Y}$ to denote the image and its label.

By using a global average pooling process (as detailed in Section 2.4.2) on the feature $\boldsymbol{F}$, the feature vector $\boldsymbol{a^{fc}}$ is obtained. According to the background knowledge about the forward computation presented in Chapter 2, the net input $\boldsymbol{z}_i^{fc}$ of the fully connected layer is:

$$\boldsymbol{z_i^{fc}} = \sum_{j=1}^{n} \boldsymbol{w_{i,j}^{fc}} \boldsymbol{a_j^{fc}}, \tag{3.1}$$

where $\boldsymbol{w^{fc}}$ are the weights in the fully connected layer, while the bias term is ignored in [53], since it has little to no impact on the performance. The output $\boldsymbol{s^{fc}}$ of the fully connected layer is:

$$\boldsymbol{s_i^{fc}} = \sigma(\boldsymbol{z_i^{fc}}), \tag{3.2}$$

where $\sigma(\cdot)$ is the sigmoid activation function, and $\boldsymbol{s_i^{fc}}$ represents the class score for class i. The class score $\boldsymbol{s^{fc}} \in [0, 1]^l$, with l is the number of predefined classes in the dataset (see Section 3.2.4) and it is used to compute a multi-label classification loss function, which is detailed in Section 2.3.2, to train the model with the image-level label $\mathcal{Y}$.

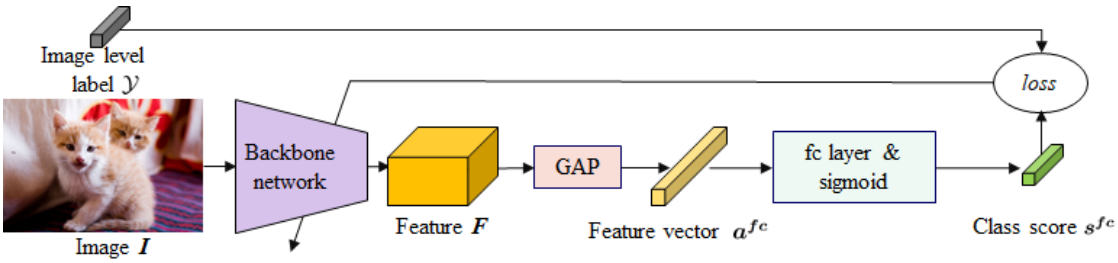


Figure 3.4: Illustration of the classification training pipeline used in method [53]. *The* GAP *box is the global average pooling process. The* fc layer & sigmoid *box represents a fully connected layer with sigmoid activation function.*

When the model is trained, the class-wise features $\boldsymbol{D} \in \mathcal{R}^{h \times w \times l}$, is obtained by linearly combining the weights $\boldsymbol{w}^{\boldsymbol{fc}}$ with the feature map $\boldsymbol{F}$:

$$\boldsymbol{D}^i = \sum_{j=1}^{n} \boldsymbol{w}_{i,j}^{\boldsymbol{fc}} \boldsymbol{F}^j, \tag{3.3}$$

where $\boldsymbol{F}^j$ represents the 2-dimensional map in the channel j of the 3-dimensional feature $\boldsymbol{F}$, and $\boldsymbol{D}^i$ represents the 2-dimensional class-wise feature map for class i.

The main limitation for [53] is that the generation of the class-wise feature required a 2-step process. The network should be trained at first, then the weights of fully connected layer are used.

Therefore, [106] proposes an online method to directly obtain the class-wise feature in the forward pass without demanding the presence of a fully connected layer in the network. As shown in Figure 3.5, a convolutional layer with the 1×1 convolutional kernels is added to process the output feature $\boldsymbol{F}$ produced from the directly into the class-wise feature map $\boldsymbol{D}$:

$$\boldsymbol{D}^i = \sum_{j=1}^{n} \boldsymbol{w}_{i,j} \boldsymbol{F}^j, \tag{3.4}$$

where $\boldsymbol{w}$ are the weights in the new added convolutional layer, $\boldsymbol{D}^i$ is a 2-dimensional feature map for class i.

Then, a global average pooling process is applied on the class-wise feature map $\boldsymbol{D}$ to generate the class-wise feature vector $\boldsymbol{a}$. With a sigmoid process, the prediction score $\boldsymbol{s}$ is obtained:

$$\boldsymbol{s}_i = \sigma(\boldsymbol{a}_i), \tag{3.5}$$

where $\sigma(\cdot)$ is the sigmoid function, and the prediction score $\boldsymbol{s}$ is used to compute the classification loss with the image-level label $\mathcal{Y}$ to train the model.

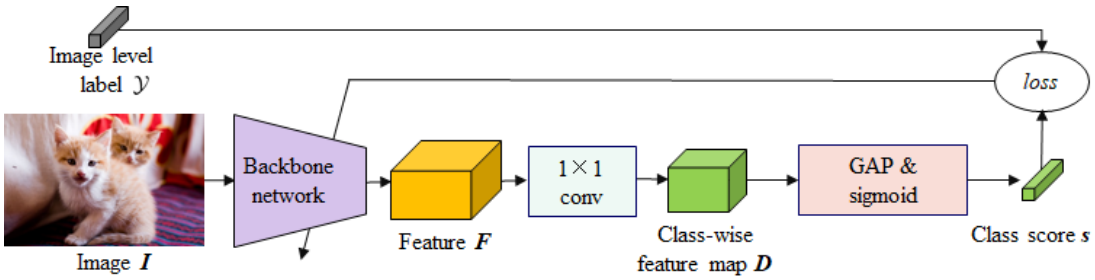


Figure 3.5: Illustration of the classification training pipeline used in method [106]. *The* GAP & sigmoid *box is the global average pooling process with activation by sigmoid function. The* 1×1 conv *box represents the convolutional layer with convolutional kernels' size of* 1×1.

In practice, the class-wise feature maps obtained using method in [53] and method in [106] are very similar and highlight the same target regions except for some marginal differences caused by the stochastic optimization process [106]. Since method in [106] can be directly and explicitly obtained in the forward pass rather than requiring a post-processing step, it is commonly used in recent WSSS methods to generate class-wise feature map during the training process of the classification network.

Generating the Class Activation Map

Given the training is done, the class-wise feature map $\boldsymbol{D}$ is fed into a ReLU layer, and then normalizing to obtain CAM $\boldsymbol{A}$:

$$\boldsymbol{A}_i^c = \frac{\text{ReLU}(\boldsymbol{D}_i^c)}{\max(\boldsymbol{D}^c) + \epsilon}, \tag{3.6}$$

where ϵ is a small value to avoid the denominator being zero, and $\boldsymbol{A}_i^c$ is the activation value, ranging from 0 to 1, at the location index i (using lexicographic ordering) of $\boldsymbol{A}$ of class c, with $i \in \mathcal{U}$ the set of all location indexes in $\boldsymbol{A}^c$. (In Chapter 2, we used two indexes to represent a specific location of a given matrix to better explain operations such as convolution process. For simplicity, we now use only a single index within a set of location indexes to describe the corresponding location.)

Generally, the spatial resolution of $\boldsymbol{A}$, i.e., $h \times w$, is relatively small compared with the original size of the image, i.e., $H \times W$. A bilnear interpolation process is used to resize $\boldsymbol{A}$ to $\boldsymbol{A'}$ by giving it the spatial size of $\boldsymbol{I}$. The value of $\boldsymbol{A'}_i^c$ represents the importance of pixel at the location i to be classified as the class c. Examples of resized CAM for image with one or two objects are shown in Figure 3.6 and Figure 3.7.

We can observe that the CAM provides limited location information for the objects in the given image, by highlighting the most discriminative areas of each class. However, such location information is not sufficient enough to offer accurate segmentation results for the objects.



Figure 3.6: Examples of CAM for images with one class. From top to bottom: image and CAM of the image. These CAMs are obtained from a classification network which backbone is ResNet-50.



Figure 3.7: Examples of CAM for images with two classes. From top to bottom: image, CAM for the 'person' class, and CAM for the other class of the image. These CAMs are obtained from a classification network which backbone is ResNet-50.

As widely discussed, the main limitations for CAM are as follows:

1. CAM only focuses on the discriminative regions while it fails to provide entire segmentation for the whole objects.
2. Generated from the deep feature given by the last convolutional layer in the network, the spatial size of CAM is quite smaller compared to the size of original image. A simple interpolation resize can not recover details for the activation, which makes CAM fuzzy and without clear boundaries.
3. Ambiguous regions appears in the CAM, when the same spatial locations have significant activation values for more than one class.

Most of the recent WSSS methods focus on obtaining enhanced CAM by designing new loss functions [36, 37, 38, 39, 40, 107] or augmenting the input data [21, 108, 109]. Typical related methods are presented in Section 3.4.1.

3.3.2 Obtaining pseudo-masks from CAM

Since training a segmentation model requires images with pixel-wise annotations, 2-dimensional pseudo-masks that encode inferred labels for all pixels need to be

obtained from the 3-dimensional resized CAM $\boldsymbol{A}'$.

Specifically, a 2-dimensional mask $\boldsymbol{M}$, which is generally referred as seed, is finally obtained by applying a fixed threshold and an argmax:

$$\boldsymbol{M}_i = \begin{cases} \underset{c \in \mathcal{Y}}{\operatorname{argmax}}(\boldsymbol{A}_i'^c) & \text{if } \underset{c \in \mathcal{Y}}{\max}(\boldsymbol{A}_i'^c) > T_{fg}, \\ 0 & \text{otherwise,} \end{cases} \tag{3.7}$$

where the argmax function is used to find the argument that maximizes the expression, $\boldsymbol{A}_i'^c$ is the value of the resized CAM for class c at location i and $\boldsymbol{M}_i$ is the hard label at location i, with $i \in \mathcal{U}$. T_{fg} is a threshold to fix. The background is encoded as class 0 by convention.

Note that if the threshold T_{fg} is too high, seed masks will only label the most discriminative part of the object, while if it is too low, the risk of labeling background pixels as objects increases.

The seed $\boldsymbol{M}$ has the same size as the size of original image $\boldsymbol{I}$. When processing CAM to obtain pseudo-masks, a resize operation has to be performed (since the the spatial resolution of CAM $\boldsymbol{A}$ is small compared with the original size of the image $\boldsymbol{I}$, as mentioned before). The resize operation can be done before [41, 68, 110, 111, 112] or after the thresholding step [27, 39, 113, 114, 115] without significant impact on the accuracy of the final results. In this thesis, the resize operation is done before the thresholding step.

Theoretically, as 2-dimensional masks, the seed can be directly used as the pseudo-masks to train a segmentation model. However, the inherited inaccuracies from CAM can negatively impact the performance of segmentation models. Examples of seed mask are given in Figure 3.8 and Figure 3.9, for images with single and two objects, respectively. These examples are the same as the ones used in Figure 3.6 and Figure 3.7. They clearly show that, the boundaries of objects in the seed masks are fuzzy and imprecise. Recent WSSS methods apply post-processing approaches [25, 26, 27, 28, 29, 116], which will be introduced in Section 3.4.2, to improve these seed masks, resulting in pseudo-masks with high accuracy. These pseudo-masks are used as the training targets to train a fully supervised segmentation model, which is presented in the next section.



Figure 3.8: Examples of seed mask for images with single class. From top to bottom: image and seed mask.

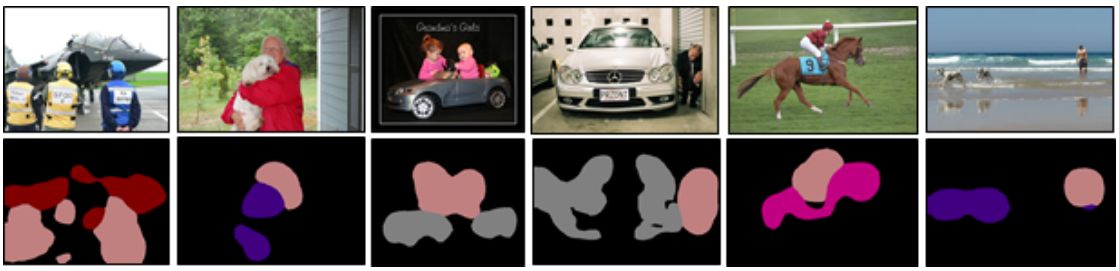


Figure 3.9: Examples of seed mask for images with two classes. From top to bottom: image and seed mask.

3.3.3 Training a segmentation model with pseudo-masks

Training process

As shown in Figure 3.3, the second stage in the two-stage training pipeline of WSSS is to train a fully supervised segmentation model using images and their corresponding pseudo-masks as the target, considering them as ground truth. The supervised training process and typical segmentation models [10, 11, 13, 14, 15, 16] have been introduced in Chapter 2. The most common choices are DeepLab segmentation models [13, 14, 15, 16] with ResNet-38 [70] or ResNet-101 [66] backbone networks.

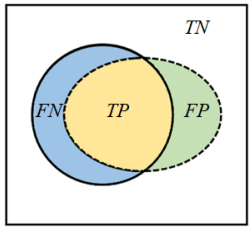
Evaluation in WSSS

To compare WSSS methods and analyze the results they give, we need to evaluate the accuracy of the trained segmentation network. Besides, we can also evaluate the accuracy of the pseudo-masks, since we believe that better pseudo-masks give better segmentation models.

Among accuracy measures, the Intersection-over-Union (IoU) metric for each class is very often adopted for segmentation problems. It measures the overlap between the predicted mask and the ground truth mask. To compute the IoU metric, a confusion matrix, as shown in Figure 3.10.(a), is constructed by comparing predicted labels of pixels with the ground truth. Four basic cardinalities of the confusion matrix are illustrated in Figure 3.10.(b) by [117].

		Predicted	
		Positive	Negative
GT	Positive	*TP*	*FN*
	Negative	*FP*	*TN*

(a)



(b)

Figure 3.10: An illustration of (a) the confusion matrix and (b) four basic cardinalities of the confusion matrix illustrated by [117]. *GT represents ground truth. TP, TN, FP and FN stand for True Positive, True Negative, False Positive and False Negative, respectively.*

Specifically, the circle with plain line boundary and oval with dot line boundary represent the ground truth and prediction for one class, respectively. The terms positive and negative in the confusion matrix for semantic segmentation represents whether pixels are associated to a certain class or not. *TP* stands for True Positive, indicating the number of pixels (in yellow regions) correctly classified as part of the class. *TN* stands for True Negative, representing the number of pixels (in white regions) correctly classified as not part of the class. *FP* stands for False Positive, referring to the number of pixels (in green regions) incorrectly classified as part of the class. *FN* stands for False Negative, showing the number of pixels (in blue regions) that belong to the class but were not classified as such. Then, the IoU metric is computed by:

$$IoU = \frac{TP}{TP + FP + FN}. \tag{3.8}$$

The mean Intersection-of-Union (mIoU) is also commonly used in the semantic segmentation task, which IoU scores across all classes in the dataset.

It is observed that, the IoU metric is sensitive to the prediction errors of tiny objects, which contain fewer pixels compared to the large ones [118]. This sensitivity can lead to misleading quantitative results when using IoU alone. Therefore, to compensate the limitations of IoU in capturing detailed accuracy, especially in tiny objects, we also visualize segmentation results for qualitative assessment in experiments.

Discussion

The accuracy of the pseudo-masks will of course directly influence the performance of the segmentation model. In the standard fully supervised framework, regardless of the correctness of the labels in the pseudo-masks, all the labels are used as training targets. In this case, incorrect labels inevitably degrade the segmentation capability of the model. Therefore, various recent methods [25, 26, 27, 28, 29, 36, 37, 38, 39, 40, 119] in WSSS focus on improving the accuracy of pseudo-masks as their primary goal. As we can see from Figure 3.11, the great improvement in the quality of the pseudo-masks leads to considerably enhanced final segmentation performance. Therefore, the main concern in this thesis is to propose new methods to improve the accuracy of the pseudo-masks for weakly supervised semantic segmentation.

However, the relevance to continue to train the segmentation models using the standard fully supervised framework, despite knowing that pseudo-masks can not be as accurate as the ground truth annotations could be questioned. Figure 3.12 provides some visual comparisons between the outputs of the segmentation models [14] trained using ground truth in the FSSS task, or pseudo-masks in the WSSS task. It is observed that, the model trained using pseudo-masks tends to generate predictions with imprecise and blurred boundaries, even if those pseudo-masks seem relatively accurate. This effect is mitigated with full supervision (as can be seen in the third row in figure 3.12). It suggests that the features learned by the WSSS model are adversely affected by the labeling errors of the pseudo-masks.

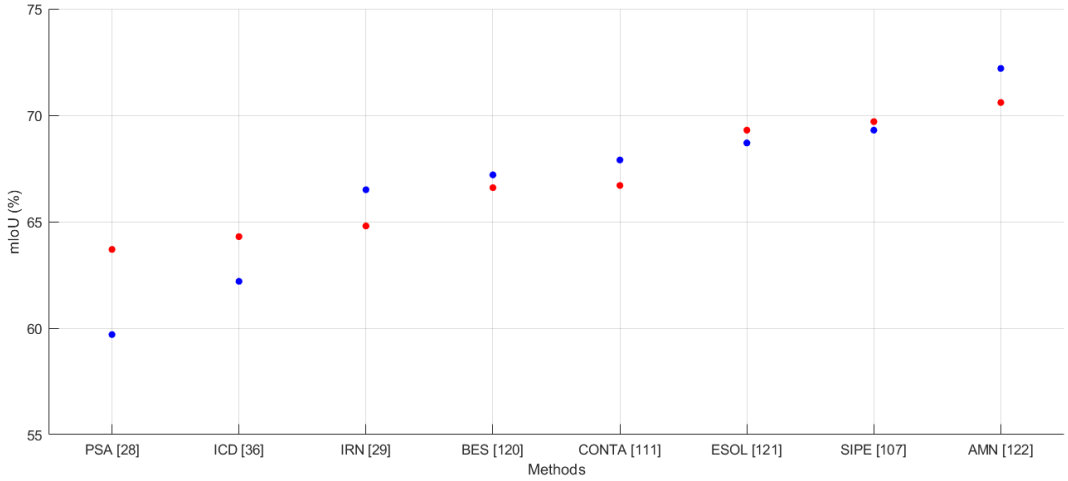


Figure 3.11: Illustration of the relationship between the accuracy of pseudo-masks with the segmentation performance: mIoU (%) of the pseudo-masks on train set (blue dot) and mIoU (%) of the prediction masks on test set (red dot) of PASCAL VOC 2012 [18], *given from DeepLabV2 segmentation model* [14] *trained by pseudo-masks from several WSSS methods* [28, 29, 36, 107, 111, 120, 121, 122].

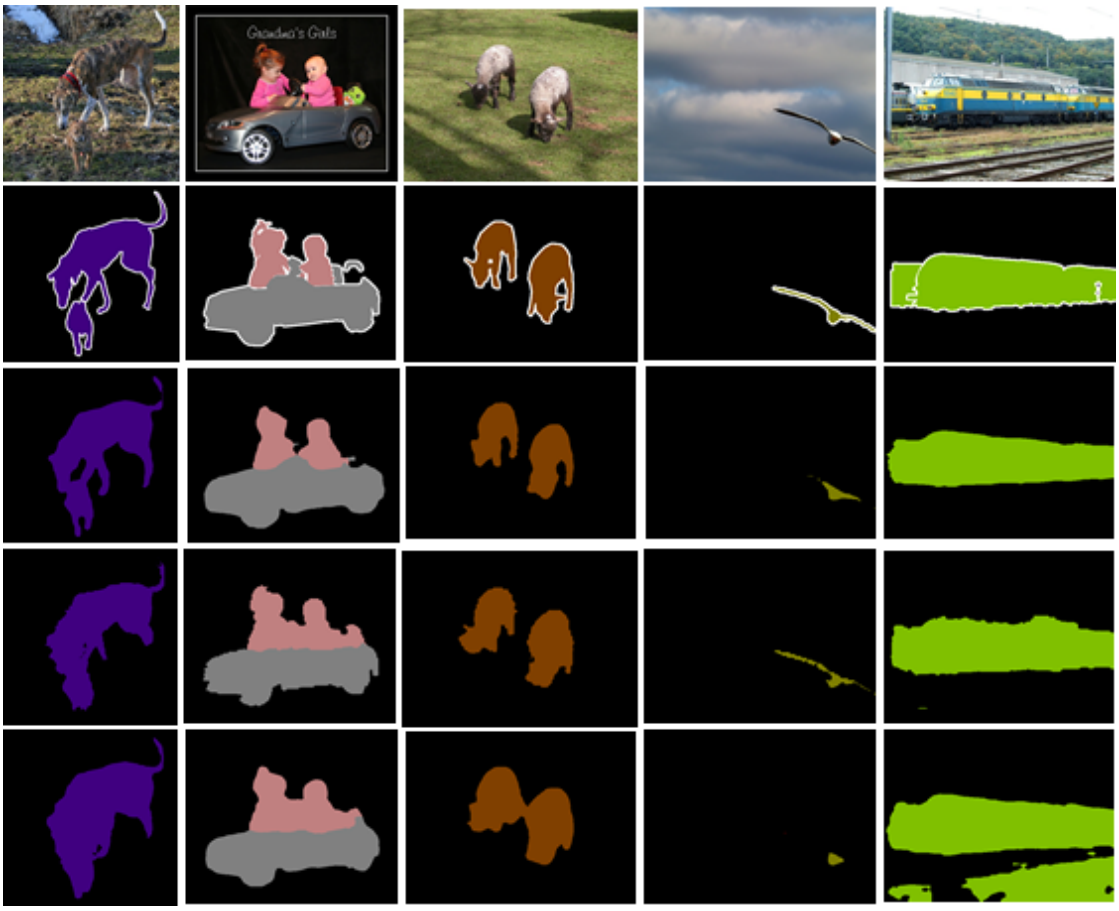


Figure 3.12: Visual comparison between the output of the DeepLabV2 segmentation model [14] *trained using ground truth or pseudo-masks generated by method in* [107]. *From top to bottom: image, ground truth, output of the DeepLabV2 network trained with ground truth, pseudo-masks, output of the DeepLabV2 network trained with pseudo-masks.*

Therefore, recent methods, which are introduced in Section 3.4.3, propose adaptive strategies for training the segmentation models to reduce the negative impact of the incorrect labels in the pseudo-masks [113, 115, 123, 124].

3.4 Recent methods in WSSS

This section introduces various recent methods aims at improving the segmentation performance of WSSS with only image-level labels. These methods are roughly divided into three groups. Each group is corresponding to one of the steps in the pipeline discussed in the previous section. Specifically, Section 3.4.1 presents methods which focus on generating the enhanced CAM to recognize the whole objects [21, 36, 37, 38, 39, 40, 108, 109]. The target of methods in this group is to eliminate the limitations of CAM by introducing new loss functions in the training of the classification model or by augmenting input data. Section 3.4.2 introduces methods which assume that, even the CAM is not accurate enough, the confident location information within CAM is worth exploiting [25, 26, 27, 28, 29, 41, 116]. Thus, new processes are proposed to obtain refined pseudo-masks from the CAM. Besides, Section 3.4.3 presents adaptive strategies in the training of the segmentation model with the pseudo-masks [113, 115, 123, 124]. Considering that incorrect labels inevitably exist in the pseudo-masks, the intuition of these strategies is to maximize the use of reliable annotations while minimizing the impact of unreliable annotations in the training process.

3.4.1 Generating enhanced CAM

As presented in Figure 3.6 and Figure 3.7, generated from the classification model trained with only image-level labels, CAM only provides limited location information, which is not sufficient to segment objects in images. Therefore, recent methods have been proposed to improve the accuracy of CAM. Generally, these methods follow 2 different research directions: In the first direction, new loss functions and constraints, which address specific limitations in the CAM, are introduced in the training of the classification model [36, 37, 38, 39, 40]. In the second direction, methods are independent of the training process, which focus on augmenting images to obtain enhanced CAM [21, 37, 108, 109]. Typical methods in these two research directions are described in the following paragraphs.

Optimizing the classification model for CAM

In recent publications [36, 38, 119], authors argue that, the limitations of CAM can be traced back to the different objectives between the image-level classification task and the segmentation task. A classification model tends to classify objects in given images by giving importance to their most discriminative characteristics, which causes the class-wise feature map, and thus the CAM to focus mostly on these regions. Besides, since CAM does not directly participate in the training process of the classification model, its properties are left uncontrolled and its limitations are not actively penalized or corrected throughout the training. Therefore,

recent methods [36, 37, 39, 40, 103, 119] focus on the characteristics of CAM and introduce newly designed loss functions in addition to the conventional classification loss function to jointly train the classification networks and enforce specific properties to the CAM. These methods enhance the semantic information in the class-wise feature, and improve the ability of CAM to describe the objects in the image.

For example, considering that CAM focuses on discriminative regions of foreground objects and lacks a detailed description of background areas, [36] introduced a loss function to enhance the pixel-wise feature contrast between foreground and background regions. This method increases the estimation accuracy of both foreground and background areas, resulting in clear intra-class boundaries and improved recognition capabilities in the enhanced CAM, as illustrated in Figure 3.13.

In [38], an iterative erasing process is proposed to replace discriminative regions in the image by the mean pixel values of all the training images. By removing the most characteristic parts of images, the training process needs to give importance to less discriminative regions. More regions for the object are perceived with active role in object identification. [39] proposes a suppressor applied on the intermediate feature maps to decrease the contributions from the discriminative regions to the final prediction in the classification model, as shown in Figure 3.14. The suppression module contains an extractor to select K maximum elements in the feature maps, and a controller to decide how much to suppress for the selected elements. The suppressor spreads the activation into adjacent less-discriminative regions during the training. Similarly, different with using the global average pooling process to obtain the averaged value $\boldsymbol{a}^c$ from the feature map $\boldsymbol{D}^c$ for class

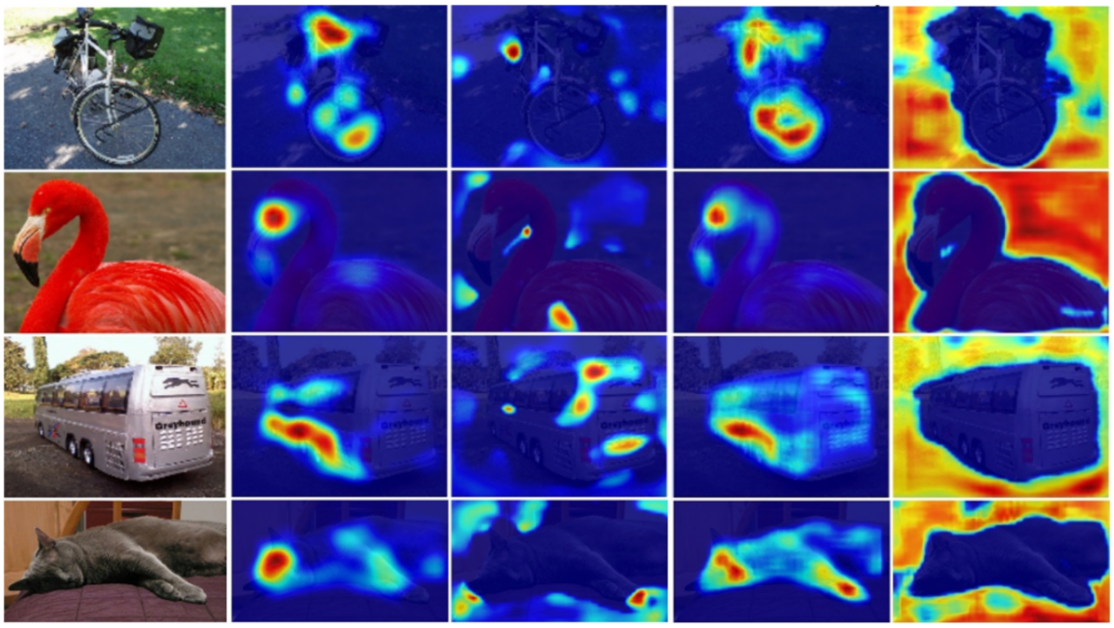


Figure 3.13: Visual comparison between the original CAM and enhanced CAM from [36]. *From left to right: image, foreground and background estimated in the original CAM, foreground and background estimated in the enhanced CAM obtained by* [36]. *The estimated background response is deduced as complementing the class-wise CAM.*

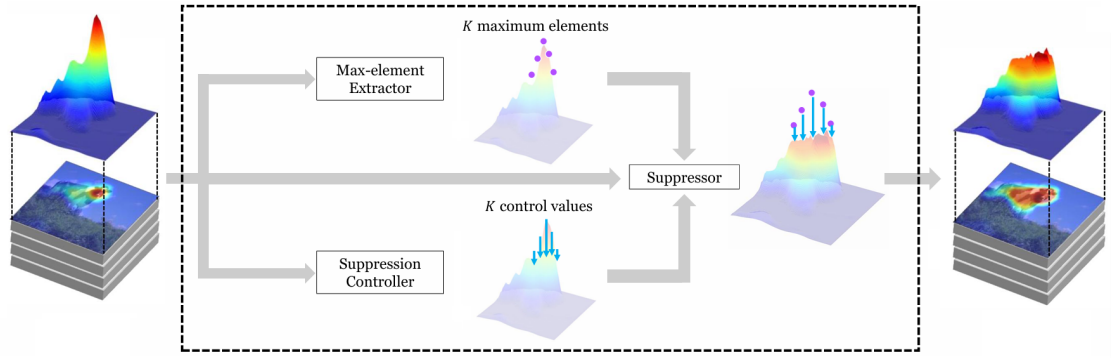


Figure 3.14: Illustration of the suppressor used in [39]. *The input and output are the original and suppressed feature maps.*

c, as presented in Section 3.3.1, [40] proposes a new pooling method, by selectively aggregating the values of the feature map $\boldsymbol{D}^c$ in spatial locations where CAM values are below a fixed threshold τ:

$$\boldsymbol{a}^c = \sum_{i \in \mathcal{U}_\tau} \boldsymbol{D}_i^c, \tag{3.9}$$

where $\mathcal{U}_\tau = \{i \in \mathcal{U} | \boldsymbol{A}_i^c \leq \tau\}$, with $\boldsymbol{A}_i^c$ the value of CAM $\boldsymbol{A}$ for class c in the spatial location of i, and $\mathcal{U}$ is a set of all spatial location indices in $\boldsymbol{D}^c$. Such aggregating method encourages less-discriminative regions to be further activated.

The main challenge in these methods is finding an effective connection between the implemented loss functions and the resulting enhancement of CAM. Due to the absence of location guidance, the addition of such loss functions does not always lead to enhance the CAM. Besides, a pointed limitation for methods aimed at activating less-discriminative regions is the lack of criteria to decide when to stop the operation. The practicality of these methods is mostly determined by experience. Activating less-discriminative specific regions often accompanies false positives in background areas, which can harm the accuracy of CAM. Generally, these methods [36, 38, 39, 40] need use auxiliary cues, as presented in Section 3.5, to determine the background and remove the false positive predictions.

Following the similar paradigm, in Chapter 4, we introduce 2 methods by using designed loss functions for jointly training the classification network with the classification loss function to obtain enhanced CAM.

Besides, as noted in the discussion for the Figure 2.11 in Section 2.4.1, different layers in the network produce distinct features. However, the class-wise feature map used for generating CAM is produced only by the last convolutional layer. This approach limits the range of semantic information available, as it neglects the diverse features captured in earlier layers.

Aimed at this issue, a Self-supervised Image-specific Prototype Exploration (SIPE) method to capture more comprehensive semantics from multiple features has been proposed in [107]. As shown in Figure 3.15, to extract these multiple features, SIPE adds four convolution layers with 1×1 convolutional kernels on four different layers

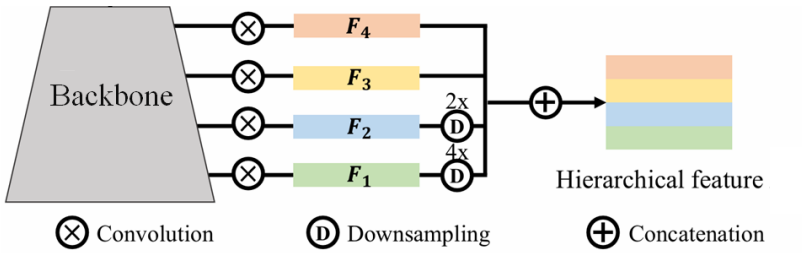


Figure 3.15: Illustration of the extraction of hierarchical feature in [107].

in the backbone network. These multi-scale features are resized to the same size and concatenate to form the hierarchical features. Based on location information derived from the original CAM, the hierarchical features are averaged over objects regions to obtain prototype feature vectors. Subsequently, an improved CAM is obtained by calculating the similarity between hierarchical features at each position and the prototype feature vector for each object. Finally, a consistency loss, which is formulated as a L1 normalization between the original CAM and the enhanced CAM, is introduced to minimize their differences, which further optimizes the feature representation and empowers a self-correction ability for the classification model. The enhanced CAM provided by SIPE is accurate. In Chapter 5, we choose the SIPE method as our baseline in the experiments.

Generating enhanced CAM by augmenting images

Enhanced CAM can also be obtained by applying augmentation process, such as multi-scaling [21, 119] and cropping [37, 108, 109], on the input images. Figure 3.16.(a) briefly illustrates the generation of CAM by only using the original image (scale = 1). By contrast, methods in [21, 119] propose a scale-aggregate pipeline to aggregate results for multi-scaling images to obtain stable CAM, as illustrated in Figure 3.16.(b). Besides, methods in [37, 108] use a crop-merge pipeline to enhance CAM by merging the results obtained for different cropped pieces of the original image from the trained classification network, as presented in Figure 3.16.(c). These methods are independent of the training of the classification network, which makes them easy to implement and use.

An explanation for the reason why cropping process can activate more regions in the CAM is that, when an image is cropped into different parts, the discriminative regions are likely to be separated, and this separation can make the previously less-discriminative areas in each part of the image more likely to be activated. In order to illustrate this behavior, a toy experiment has been designed and shown in Figure 3.17. The image is manually cropped and only keep the part which is less-discriminative before cropping. It is observed that, when the discriminative regions is removed, new regions are activated in the left part of the image, which are regarded as less-discriminative regions in the previous input.

In practice, above methods are popular due to their implementation simplicity and relative efficiency, especially for the scale-aggregating process. The use of multi-scaling inputs to enhance CAM highlights the importance of considering the impact of resolution of input on class-wise features and CAM itself. High-resolution inputs can recover more detailed information, but they also bring a number of

false predictions. Conversely, low-resolution inputs may only vaguely indicate the approximate regions of objects. Furthermore, the results from applying cropping and merging techniques to enhance CAM demonstrate that removing discriminative areas allows less-discriminative regions to become activated, thereby enlarging the recognition area for the target, which validates the strategy of reducing the contributions from discriminative regions to classification. These methods provide insightful guidance when we propose new methods in Chapter 4 and Chapter 5.

3.4.2 Obtaining improved pseudo-masks

In Section 3.3.2, we illustrate the process to obtain pseudo-masks from CAM. Without further post-processing methods, obtained pseudo-masks are not accurate

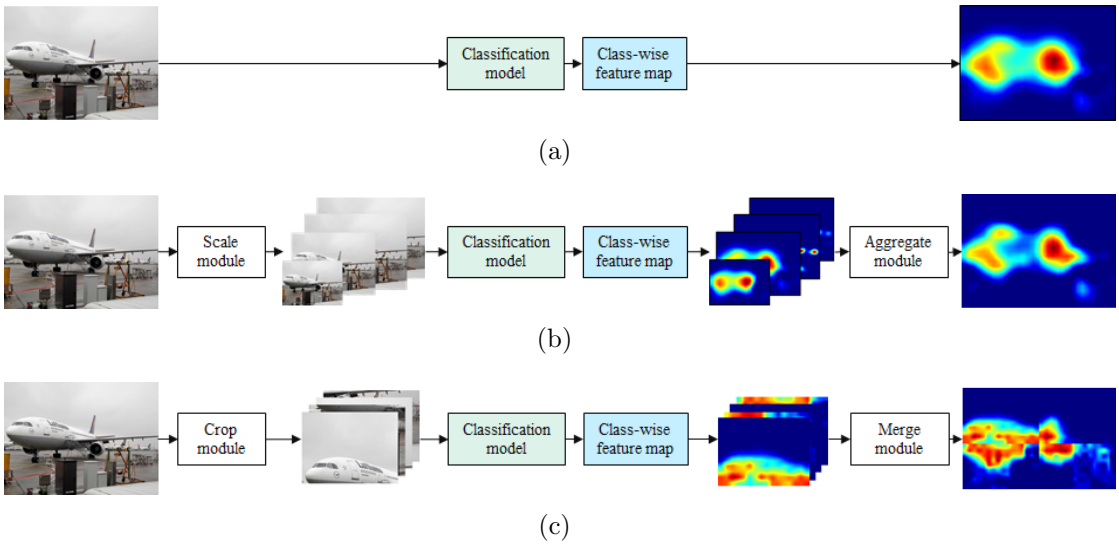


Figure 3.16: Illustration of the pipelines to generate CAM from (a) original image, (b) multi-scaling image in [21, 119], *and (c) cropped image in* [37, 108].



Figure 3.17: A toy example to obtain enhanced CAM by manually crop the image from the discriminative region. From left to right: the original CAM with the image, the cropped images with corresponding CAM. The parts of image under the gray masks are cropped and only the rest parts of the image are used as inputs when obtaining the CAM.

enough to train a segmentation model. Another approach followed by [25, 26, 27, 28, 29, 41, 116] is to improve pseudo-masks with existing CAM. These methods do not affect the classification model that generates the CAM, and can be applied after any method focused on producing enhanced CAM within the WSSS framework.

Typical methods [25, 41, 42] are proposed under assumption that, discriminative regions in the CAM of an image contains reliable information that can be used to infer labels for the rest in less activated region of the image. For instance, as shown in Figure 3.18, Deep Seeded Region Grows (DSRG) method [41] firstly only assigns labels to the discriminative regions identified by high values in CAM, while leaving the rest unlabeled (the white regions in the figure). Then, labels of unlabeled regions are iteratively inferred using a similarity criterion. If an unlabeled pixel shares similar feature, which is generated from the classification network, with its neighbor labeled pixel, it will be assigned to the same label as the labeled one. This process is referred as expansion, since it allows the confident labeled region to expand as the process continues.

Besides, methods in [28, 29, 116] are always refereed as refinement processes in recent WSSS approaches. These refinement methods are proposed to distinguish the object class and background for the given image, resulting in more accurate object boundaries in pseudo-masks.

Specifically, as a statistical model, the dense Conditional Random Field (dCRF) associates an input image with the output segmentation labels by globally considering the interactions between pixels in the given image [116]. It is inspired by the observation that, nearby pixels with similar color and texture are more likely to belong to the same class, and small, isolated regions may represent incorrect predictions.

Authors in [28] proposes the Pixel-level Semantic Affinity (PSA) to describe the semantic similarity between the pixel-wise features. Same label are assigned to pixels which share similar features. As illustrated in Figure 3.19, CAM is firstly obtained from a trained classification network, and then used to generate the initial seed mask with unlabeled regions, similar to the first step in DSRG. Binary labels are assigned to pairs of adjacent coordinates according to their class consistency in

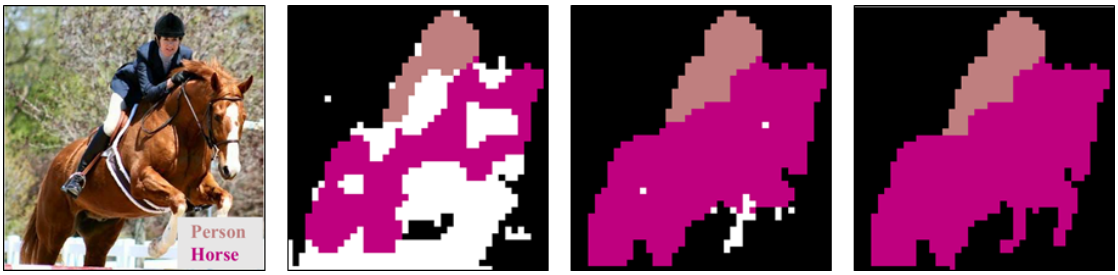


Figure 3.18: Illustration of the seeded region growing in method [41]. *From left to right: the original image with its image-level label, the seed mask at the beginning, after 6 iterations and after 12 iterations. The pink and purple regions in the seed masks represent the person and horse objects, black regions for background, and the white regions are temporary unlabeled regions.*

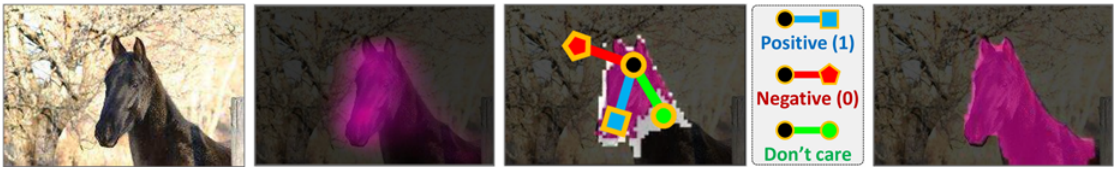


Figure 3.19: Illustration of semantic affinity labels in [28]. *From left to right: the original image, CAM for the object respect to the image, labels of location pairs with the explanation for the labels, and the refined pseudo-mask on the image.*

the seed mask. These labeled pairs are then used as the training targets to train a network, which in turn predicts semantic affinities for pixels in the given image to refine segmentation labels in the mask.

Method in [29] proposes Inter-pixel Relation Network (IRNet), which is regarded as an updated version from the PSA method [28]. There are two branches in the IRNet. The first branch is designed to predict semantic affinities, similar to method in [28]. The second branch computes a displacement vector field, where a 2D vector connects each image pixel to the centroid of the object to which that pixel belongs. This displacement field is converted into a segmentation map by assigning the same label to pixels whose displacement vectors point at nearby locations. By combining the outputs from both branches, refined segmentation masks are produced.

These refinement methods [28, 29, 116] are commonly used after generating the enhanced CAM by using methods mentioned in Section 3.4.1, to obtain more accurate pseudo-masks.

It is observed that, the main differences in these post-process methods are two aspects. The first one is the used pixel-wise features. For DSRG [41], PSA [28] and IRNet [29], the used semantic features are generated from a CNN-based model, but dCRF [116] uses the color feature from the image directly. The second one is that, the Affinity network and IRNet are trained globally among the whole dataset, while the DSRG and dCRF are implemented for each image. Compared with the results from DSRG and dCRF, PSA and IRNet fall short in modeling and distinguishing image-specific foreground and background.

Based on these process, new methods have been recently proposed by using online iteratively updating [104, 125, 126, 127], or introducing semantic features from unsupervised learning [27, 128]. However, these methods typically rely on either low-resolution deep features with rich semantics, or color of original image which is in high resolution but with limited semantics. Low-resolution features can result in smoothed boundaries and missing details, while insufficient semantic features can lead to noisy and inaccurate predictions in pseudo-masks. In addition, these methods do not take full account of each image specificity. To address these issues, we propose a new image-specific method to obtain improved pseudo-masks in Chapter 5, that seeks to strike a balance between resolution and semantics by using multiple features.

3.4.3 Adaptive training strategies for the segmentation models

As we mentioned before, pseudo-masks obtained after all previous steps are used as target labels to train a segmentation model using a standard fully supervised framework. However, as will be illustrated in the next 2 chapters, even with varied methods to improve the pseudo-masks, they can not be as accurate as the ground truth. Using a standard fully supervised approach may cause the incorrect labels in pseudo-masks to negatively influence the training of segmentation models. Therefore, some recent methods [113, 115, 123, 124] focus on training segmentation models introducing adaptive strategies, which aims at using the reliable annotations as much as possible, and avoid training with the unreliable annotations (which are likely to be incorrectly labeled) in pseudo-masks.

For example, in [113], the unreliable annotations are defined as annotations of pixels whose CAM values fall between a background threshold and a foreground threshold. These unreliable annotations are ignored when computing segmentation loss to alleviate the harm made by the potential incorrect annotations. In [115], an input image is scaled into different sizes for generating varied response maps. After resizing these maps to the initial size of image, it estimates uncertainty by calculating the consistency of response values at each location, where higher consistency indicates lower uncertainty. Less weights are associated for unreliable annotations in uncertain regions when computing the segmentation loss function. In these cases, selected reliable annotations will contribute more to the training, which leads the segmentation model to rely mostly on the confident labels in pseudo-masks. Compared with methods that focus on improving the accuracy of pseudo-masks, approaches aim at adaptive training strategies for segmentation models with pseudo-masks are relatively underrepresented, and this area of research deserves further exploration.

3.5 Auxiliary cues for WSSS

It is worth mentioning that, a number of WSSS methods [36, 39, 40, 41, 105, 112, 113, 129, 130, 131, 132, 133, 134] use auxiliary cues such as saliency and superpixels to improve segmentation accuracy, particularly in determining the background or object boundaries. Examples of saliency maps generated by method in [135] and superpixels segments generated by method in [136] are shown in Figure 3.20. Those approaches are briefly described in what follows.

Saliency

Saliency in image processing and computer vision research fields is often used to determine regions of an image which stand out from their surroundings and capture human attention [136]. Since salient regions are likely to be related with objects, recent methods use saliency to refine segmentation results by assigning background labels to the regions where values of saliency are lower than a fixed threshold [39, 40, 41, 105, 113, 130, 131, 132]. Table 3.1 presents published results

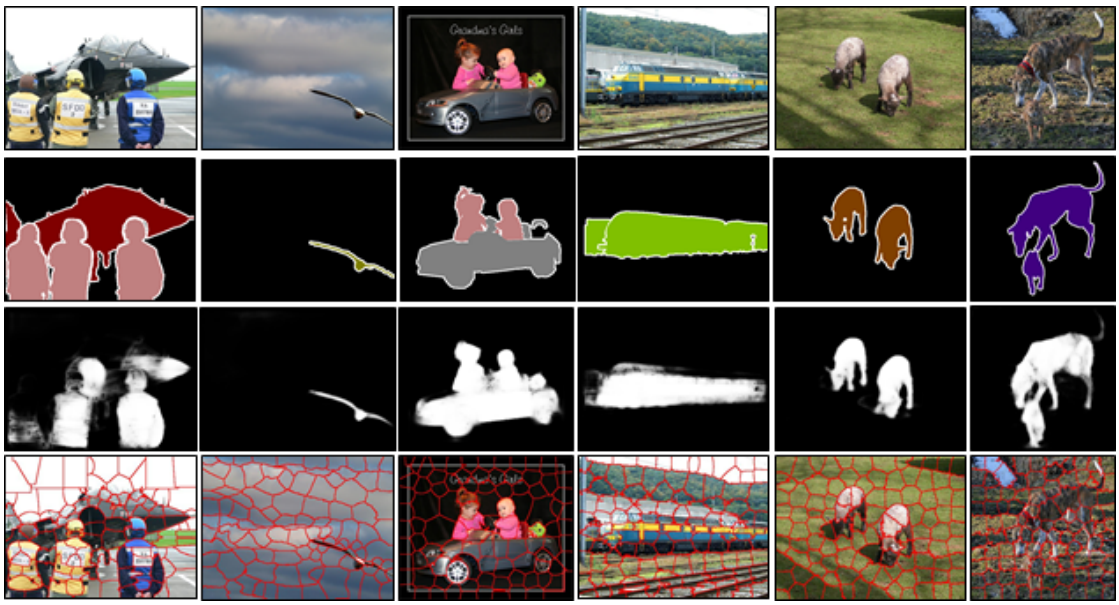


Figure 3.20: Examples of auxiliary cues used in weakly supervised semantic segmentation. From top to bottom: image, ground truth, saliency maps and superpixel segments. Saliency maps are generated by method in [135]. *Superpixel segments are generated by method in* [136].

for some WSSS methods [36, 40, 105, 113] with and without using saliency. It is obvious that, additional spatial and contextual information given by the salience can substantially improve the final segmentation results. However, the process of obtaining saliency, referred as saliency detection, typically requires training a model in a fully supervised manner, which heavily relies on large-scale supervision with pixel-wise annotations [136, 137, 138, 139]. This requirement introduces additional supervision into WSSS methods. While there are unsupervised saliency detection approaches proposed recently [140, 141, 142], they generally need to design more complex models with additional modules for achieving results comparable to fully supervised methods, which significantly increase the time and computation cost.

Superpixels

Superpixels are commonly used in image processing to group pixels in an image into large, perceptually meaningful segments, which share similar features such

Method	Val	Val*	Test	Test*
Zhang *et al.* [105]	61.7	66.6	62.7	66.7
ICD [36]	64.1	67.8	64.3	68.0
RIB [40]	68.3	70.2	68.6	70.0
SANCE [113]	70.9	72.0	72.2	72.9

*Table 3.1: The segmentation results for the WSSS methods with (noted by *) or without saliency in validation set and test set of PASCAL VOC 2012. The evaluation metric is mIoU (%).*

as color, intensity, and texture [136]. The extraction of superpixels are commonly relies on unsupervised methods to create segments that are internally homogeneous while being distinct from each other. Superpixels effectively capture the important structures and patterns within an image, which are used to provide the boundaries of objects for improving the segmentation results in WSSS methods [36, 112, 129, 133, 134].

The use of auxiliary cues is especially effective when the segmentation results contains too many false positive predictions (i.e., the pixels in background are wrongly labeled as the objects). This issue is a main challenge in the methods which aim at activating less-discriminative regions, as presented in Section 3.4.1.

Even though the use of auxiliary cues is able to provide location and boundaries information for the objects in the image, it still presents several drawbacks as follows:

1. It is not cost-free to obtain auxiliary cues. Generating saliency and superpixel data often involves using extra supervision or training additional models .
2. The inaccuracies of some auxiliary cues are inevitable, which impact negatively the overall quality and reliability of final the segmentation results.

Therefore, to ensure clarity and fairness in comparisons, we explicitly indicate when auxiliary cues is utilized in the presented results. By default, it is assumed that the methods listed in experiments rely solely on the supervision provided by image-level labels, unless specifically stated otherwise.

3.6 Conclusion

In this chapter, we introduced related works to the thesis. Firstly, we illustrate semantic segmentation under different levels of supervision. Forms of weak supervision for weakly supervised semantic segmentation are presented. Among all forms of weak supervision, the use of image-level labels is mainly concerned in this thesis. Then, we present an overview of two-stage training pipeline for weakly supervised semantic segmentation with only image-level labels, and present the main challenges of the different steps. Compared with one-stage training pipeline, two-stage pipeline performs better and is more popular in recent WSSS methods. Next, recent methods in WSSS are presented. These methods are grouped by their targets, i.e., generating enhanced CAM, obtaining improved pseudo-masks from CAM, and adaptively training the segmentation model with pseudo-masks.

The main limitations of actual approaches are relatively shared by the community. There are many aspects to address and various options can be explored. Among them, we mainly focus on generating enhanced CAM and obtaining improved pseudo-masks from CAM. Specifically, in the analysis of methods for generating CAM, we find that the primary shortcoming of CAM is due to the fact that during the training of classification networks, generating accurate CAM is not one of the objectives. This shortcoming can be addressed by introducing new designed loss

functions. Inspired by recent approaches, enhanced CAM with new classification network training methods are proposed in Chapter 4. Additionally, while studying methods for obtaining improved pseudo-masks from CAM, we notice that recent methods lack making the balance between semantics and resolution, and does not fully consider the specificity of each image. To address these issues, Chapter 5 introduces an image-specific pixel classification approach using multi-feature to improve the accuracy of pseudo-masks from CAM.

The use of auxiliary cues, such as superpixel and saliency, in WSSS is introduced and analysed. The improvements and drawbacks of these approaches are given. It is observed that, although the use of these cues can improve segmentation results, it also introduces issues such as using extra supervision, raising computational cost, and increasing the risk of additional errors. Methods presented in the next 2 chapters rely only on the image-level labels, unless specifically stated.

Chapter 4

New classification network training methods to enhance CAM

Contents

4.1 Introduction 63

4.2 Motivation 64

4.3 Proposed methods 65

4.3.1 New training method to decrease overlapping regions in CAM 66

4.3.2 New training method to enlarge activation regions in CAM 67

4.4 Experimental results 73

4.4.1 Experiments settings 73

4.4.2 Experimental results for enhanced CAM with decreased overlapping regions 74

4.4.3 Experimental results for enhanced CAM with enlarged activation regions 79

4.5 Conclusion 83

4.1 Introduction

As mentioned in Chapter 3, the generation of CAM is the first step in the training pipeline of WSSS. However, CAM only highlights limited parts of the objects, which is not sufficient and accurate for semantic segmentation learning. The limitations of CAM come from the objective difference between the image-level classification task, that produces the features to generate CAM, and the required properties of CAM to segment the objects properly. To bridge the gap between the objective in the generation of CAM and the requirement for segmentation, new methods are proposed in this chapter to train the classification model with the

generation of enhanced CAM that includes additional targets for segmentation. In this chapter, we propose two new methods to train the classification network and obtain enhanced CAM. Specifically, Section 4.2 presents the motivation for the proposed methods. Section 4.3 introduce the proposed methods in details. Experiments setting and results are described and shown in Section 4.4. Section 4.5 gives a chapter conclusion.

4.2 Motivation

Figure 4.1 presents the conventional classification model training pipeline and the CAM $\boldsymbol{A}$ generation, as introduced in Chapter 3. In this pipeline, the class-wise feature map $\boldsymbol{D}$ are deduced from the last convolutional layer of the backbone network. The CAM $\boldsymbol{A}$ is obtained after applying ReLU and normalization processes to the class-wise feature map $\boldsymbol{D}$. Since the objective of the classification model is to assign objects class to a given image, the class-wise feature tends to focus on the most discriminative regions to distinguish classes from one to another. While the class-wise feature is valuable for classification, it is insufficient for segmentation. When CAM is used to generate pseudo-masks for training a segmentation model, this becomes an issue. However, the limitations of CAM are not actively penalized or corrected throughout the conventional training phase, first due to the fact that the production of accurate CAM is not the goal of a classification model, and second because of the absence of direct feedback targeting these shortcomings. Such disconnection impacts the accuracy and reliability of the CAM in terms of object localization. Therefore, in this chapter, new methods are proposed. By introducing loss functions to enhance specific properties of CAM, generating enhanced CAM becomes an additional objective when training the classification network.

Ideally, each pixel of an image should be assigned to only one class (a predefined object class in the dataset or background). Therefore, the ideal CAM is expected to have activation value for only one class at any given location. However, it has

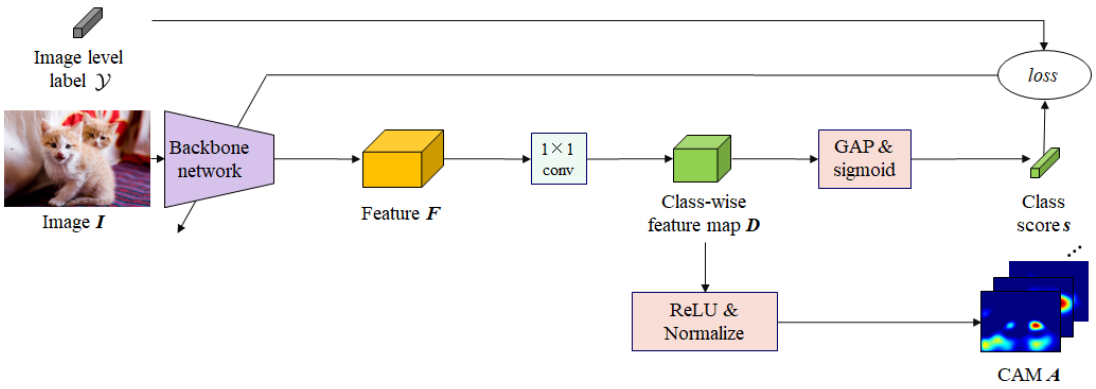


Figure 4.1: A pipeline for classification network training with CAM generation, following [106]. *The* 1×1 *conv box represents the convolutional layer with convolutional kernels' size of* 1×1. *The* GAP & sigmoid *box is the global average pooling process with activation by sigmoid function.*

been observed that, there are certain regions where multiple activation values in CAM for different classes appear at the same spatial location. Such regions are referred as overlapping regions, as presented in Chapter 3. Ambiguity exists in the overlapping regions when CAM provides location information of the objects and these regions indicate that features generated from the classification model lack distinction between classes. The first limitation for CAM we attempt to eliminate is the occurrence of such overlapping regions. In Section 4.3.1, we propose a method to obtain improved CAM where each spatial location has activation value for only one class by penalizing the overlapping regions when training the classification model.

The second limitation of CAM discussed in this chapter is that, CAM only focuses on the most discriminative class specific regions and fails to provide a significant response for the whole objects. As presented in Chapter 3, this limitation is widely discussed in recent methods [17, 36, 103, 114, 119], and is attributed to the fact that discriminative regions play the major role in classification. Specifically, for a present class in an image, the discriminative regions of objects provide local high feature values, which boost the class scores, and the training process tends to favor these regions over others and thus diminishes the potential to develop significant activation in less discriminative regions for the model.

Recent methods [38, 39, 40, 68, 143, 144, 145] activate less-discriminative regions by suppressing the contribution to the class score from the discriminative regions in the images. These methods can be roughly divided into two groups. Approaches in the first group decrease the local high response values in the class-wise features by erasing the discriminative regions in the input images [38, 143] or introducing suppression modules in the backbone network [39, 40, 144]. The second groups aims at designing new pooling methods, which make a balance between global average pooling and global max pooling processes [68, 145], or directly assign less weights to the large values in the class-wise feature map when computing the class score [40]. Details of typical related methods are introduced in Section 3.4.1. Inspired by these methods, in Section 4.3.2, a new training method with two additional branches is proposed to train the classification network with the objective to activate less-discriminative regions in CAM. Specifically, the first branch in the proposed method aims at processing the class-wise feature map to mitigate the impact of local high responses on training. The second branch uses a weighted pooling process for tuning the classification model to enforce more global activation and avoid focusing on discriminative regions only.

4.3 Proposed methods

As mentioned in Section 4.2, the limitations of CAM are caused by the difference in objective between the image-level classification task and the segmentation task. When training a classification model, the issues of CAM are not taken into account. In this section, two methods are proposed to train the classification model with the generation of controlled CAM thanks to additional objective.

4.3.1 New training method to decrease overlapping regions in CAM

In logic and probability theory, mutually exclusive is a term describing two or more events that can not happen simultaneously [146]. For the CAM, ideally, the same spatial location should not have activation values for more than one class, which means the accurate activation maps of for different classes should be element-wise mutually exclusive. However, in practice, overlapping regions occur in the CAM, where more than one activation values are shown in the same spatial location. In this section, we propose a method to obtain improved CAM to ensure that different activation maps for a given image are element-wise mutually exclusive by penalizing the occurrence of overlapping regions in the classification training process.

As shown in Figure 4.2, the proposed method is implemented within the training of the classification model. Assuming that l classes are labeled in the dataset, and $\mathcal{O} = \{1, 2, ..., l\}$ is defined as the set of classes for image-level labels. For a given image $\boldsymbol{I}$, the image-level label set is defined as $\mathcal{Y}$, with $\mathcal{Y} \in \mathcal{O}$. In this process, we mainly consider images related to multiple classes, i.e., $|\mathcal{Y}| > 1$. To describe the occurrence of overlapping regions, a matrix $\boldsymbol{U}$ is obtained, where each element $\boldsymbol{U}_{xy}$ is computed by:

$$\boldsymbol{U}_{xy} = \frac{\boldsymbol{e}^{x\top}\, \boldsymbol{e}^{y}}{||\boldsymbol{e}^{x}||\ ||\boldsymbol{e}^{y}|| + \epsilon}, \tag{4.1}$$

where $\boldsymbol{e}^{x}$ and $\boldsymbol{e}^{y}$ are one-dimensional vectors obtained by reshaping activation map $\boldsymbol{A}^{x}$ and $\boldsymbol{A}^{y}$ for class x and y, with $x, y \in \mathcal{Y}$. Each obtained vector has $n = h \times w$ elements, where h and w are the height and width of CAM $\boldsymbol{A}$. The value of $\boldsymbol{U}_{xy}$ is in fact the cos-similarity between $\boldsymbol{e}^{x}$ and $\boldsymbol{e}^{y}$. ϵ is a small value to avoid the denominator being zero.

The obtained matrix $\boldsymbol{U}$ is a symmetric matrix of size $|\mathcal{Y}| \times |\mathcal{Y}|$. All diagonal elements in the matrix $\boldsymbol{U}$ are equal to 1, and non-diagonal elements range from 0

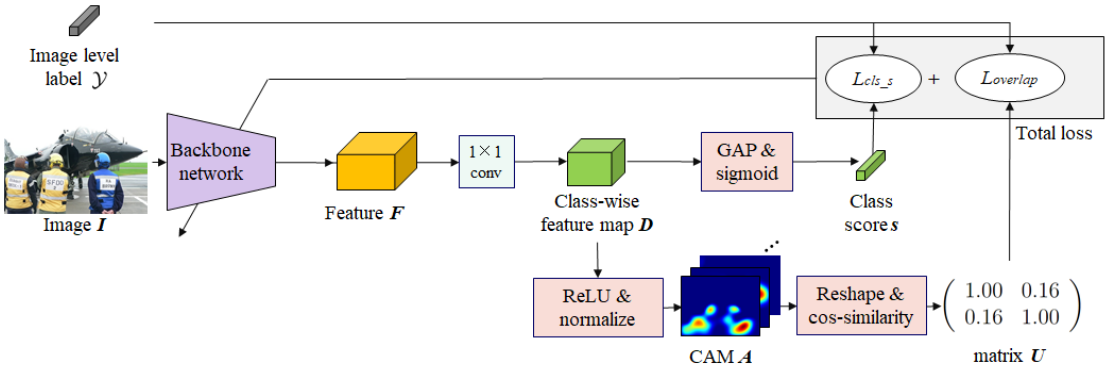


Figure 4.2: A pipeline for classification network training with the proposed $L_{overlap}$ loss to obtain improved CAM without overlapping regions.

to 1. The overlapping regions for the class x and class y exists when the element $\boldsymbol{U}_{xy}$ does not equal 0, and increases with the extent of the overlapping regions between class x and class y, where the extent is influenced by both the region size and the activation values within that region. An example of the matrix $\boldsymbol{U}$ is shown in Figure 4.2 to illustrate. For the given image in the figure related to the airplane and person class, the matrix $\boldsymbol{U}$ is of size 2×2, and the value 0.16 describes the extent of the overlapping regions in the CAM between the airplane and person class.

To penalize the overlapping regions for the classes in the given image, a new loss function $L_{overlap}$ is proposed:

$$L_{overlap}(\boldsymbol{U}, \mathcal{Y}) = \frac{1}{2} \sum_{x \in \mathcal{Y}} \sum_{y \in \mathcal{Y} | y \neq x} \boldsymbol{U}_{xy}. \tag{4.2}$$

$L_{overlap}$ is designed to minimize the value of $\boldsymbol{U}_{xy}$ to orthogonalize the vector $\boldsymbol{e}^x$ and $\boldsymbol{e}^y$. Such orthogonalization encourages the corresponding activation maps to be element-wise mutually exclusive, ensuring a decrease for the overlapping regions in CAM.

The total loss aggregates both the classification loss L_{cls_s} and the proposed loss $L_{overlap}$ when training the network:

$$L = L_{cls_s}(\boldsymbol{s}, \mathcal{Y}) + \alpha L_{overlap}(\boldsymbol{U}, \mathcal{Y}), \tag{4.3}$$

where $\boldsymbol{s}$ is the class score obtained by firstly applying global averaging pooling process on the class-wise feature map $\boldsymbol{D}$, followed by activation using a sigmoid function, L_{cls_s} is a multi-label classification loss function, detailed in Section 2.3.2, taking $\boldsymbol{s}$ and $\mathcal{Y}$ as inputs. α is the hyper-parameter to adjust the weight for the proposed loss function and to prioritize the objective task between classification performances and the generation of non-overlapping CAM.

Details of network training settings and the experiments results are given in Section 4.4.1 and Section 4.4.2, respectively.

4.3.2 New training method to enlarge activation regions in CAM

In practice, we found that the obtained CAM from the method proposed in the previous section tends to focus intensively on the most confident region, leading to a reduction in the activation regions. Considering that focusing on limited discriminative areas is one of the main concerns for CAM [17], in this section, a new method is proposed to enlarge the activation regions in the CAM. Unlike the previous section where the values in CAM is used when computing the proposed loss function, this section emphasizes on the distribution of class-wise features. In

the following of this section, an analysis of the class-wise feature is given at first, and then details of the proposed new method are presented.

As mentioned in Section 4.2, the discriminative regions always have significantly large values in the class-wise feature map. Such large values make strong contributions when generating class score by using the global average pooling, and thus causes the model to lose the potential for exploring the less-discriminative regions. Obtained after applying ReLU and normalization processes on the feature map, the CAM also concentrates in the discriminative regions, failing to activate across the whole object. Recent methods are proposed to activate less-discriminative regions by reducing the significant contributions made by local large values in the class-wise feature [17, 38, 39, 40, 68], which are detailed in Chapter 3.

Similarly, in this section, we propose a method with two additional branches to train the classification network, as shown in Figure 4.3. The first branch uses the global average pooling process to output a response score from the response map, which is generated by applying a non-linear activation function on the class-wise feature. Compared to the class-wise feature, the range of the values in the response map is smaller, which suppresses the contribution of local large values to the summarized response score. The second branch uses a new weighted pooling method, which assigns less weights to large values in the class-wise feature map, to output a weighted class score. Both scores in these two branches are related to the response map. They are used to compute additional loss functions, which tunes the classification model jointly with the initial classification loss function. Details for each branch are given as follows.

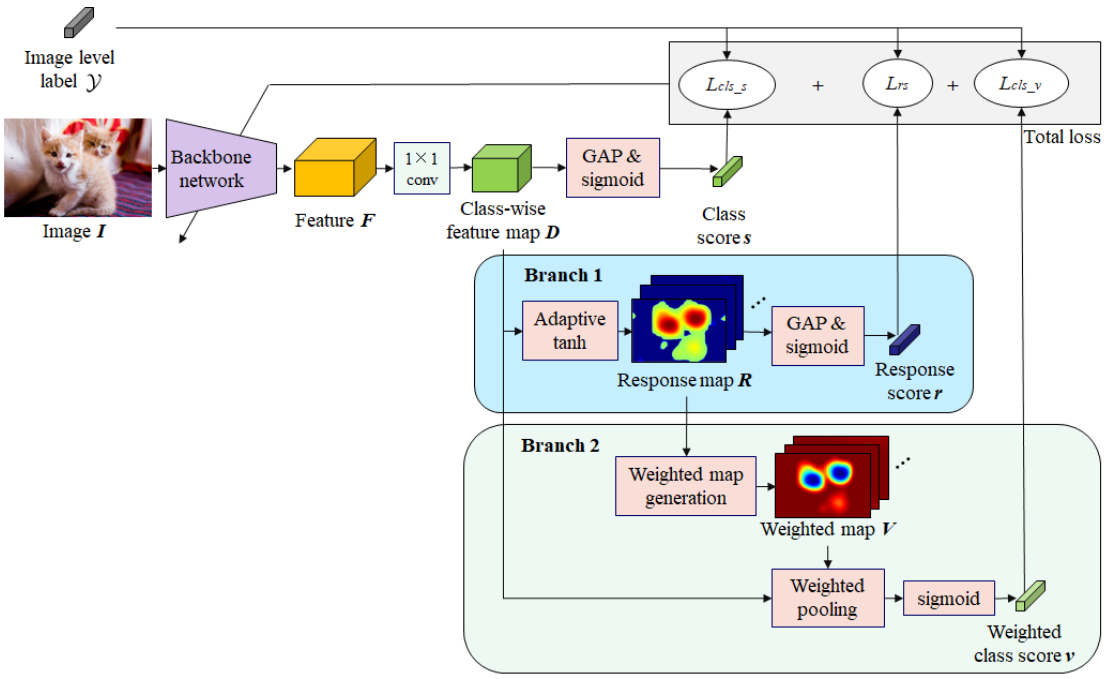


Figure 4.3: A pipeline for classification network training with the proposed L_{rs} and L_{cls_v} loss to activate less-discriminative regions in the improved CAM.

Branch 1: Training with the response score

As shown in Figure 4.3, in the first added branch, the first step is deriving the response map $\boldsymbol{R}$ from the class-wise feature map $\boldsymbol{D}$. The response map is expected to have similar distribution as the class-wise feature, which can reflect the importance for the regions in the image to a certain class. Moreover, the numerical range (i.e., the difference between the maximum and minimum values) of the response map should be smaller compared to that of the class-wise feature map, which aims at reducing the significant contribution of local large values when computing the averaged score.

In practice, we choose activation function tanh, which is introduced in Chapter 2, to achieve the goals above. Specifically, $\boldsymbol{R}_i^c \in [-1, 1]$ is obtained by $\boldsymbol{R}_i^c = \tanh(\boldsymbol{D}_i^c)$, where $\boldsymbol{D}_i^c$ and $\boldsymbol{R}_i^c$ is the value of the feature map $\boldsymbol{D}$ and response map $\boldsymbol{R}$ for class c in location i. As an illustration, for the given image in Figure 4.4.(a), the feature map for class cat is visualized in Figure 4.4.(b), and Figure 4.4.(c) shows the response map applying tanh activation function on the feature map. It is observed that, most of the positive values in the class-wise feature are converted as 1 when simply using tanh function. This leads to the vanishing gradient problem during the training process, as explained in Chapter 2.

In order to range most of the positive values in the gradient sensitive regions, an adaptive tanh activation function is proposed as:

$$\boldsymbol{R}_i^c = \begin{cases} \tanh(\tau \boldsymbol{D}_i^c) & \text{if } \boldsymbol{D}_i^c >= 0, \\ \\ \tanh(\boldsymbol{D}_i^c) & \text{otherwise.} \end{cases} \tag{4.4}$$

where τ is an image-specific parameter to rescale the positive values. It is determined by two factors, T_i and T_k, i.e., $\tau = \frac{T_i}{T_k}$, where T_i is a threshold to be fixed to rescale the positive input values into the range where the gradient of the tanh function is relatively sensitive (as illustrated in Figure 4.5). T_k represents the kth smallest value among all positive input feature values. This selection criterion is used to proportionally adjust the positive values of the input features, ensuring that the smallest k positive values fall within the region where the gradient of the tanh function is significant. Considering that the negative values in the class-wise feature map are likely related to the background, and to simplify the computation process in model training, we perform the rescaling process with τ only on the positive values.

Figure 4.4.(d) shows the response map produced by adaptive tanh function on the class-wise feature map in Figure 4.4.(b), by setting $T_i = \text{arctanh}(0.9)$ and $k = 0.8|\mathcal{U}_+|$, where $\mathcal{U}_+$ represents the set of indexes corresponding to the locations of the positive values in $\boldsymbol{D}^c$. In this case, more than 80% of the positive input feature values are in the gradient-sensitive range of tanh function. In fact, applying a scale factor τ is a common practice to rescale the output when using the tanh function. For example, method in [51] sets $T_i = 2$ and $k = |\mathcal{U}_+|$ (i.e., $T_k = \max(\boldsymbol{D}_i^c)$) when they obtaining activation maps from the class-wise feature maps by using an adaptive tanh function.

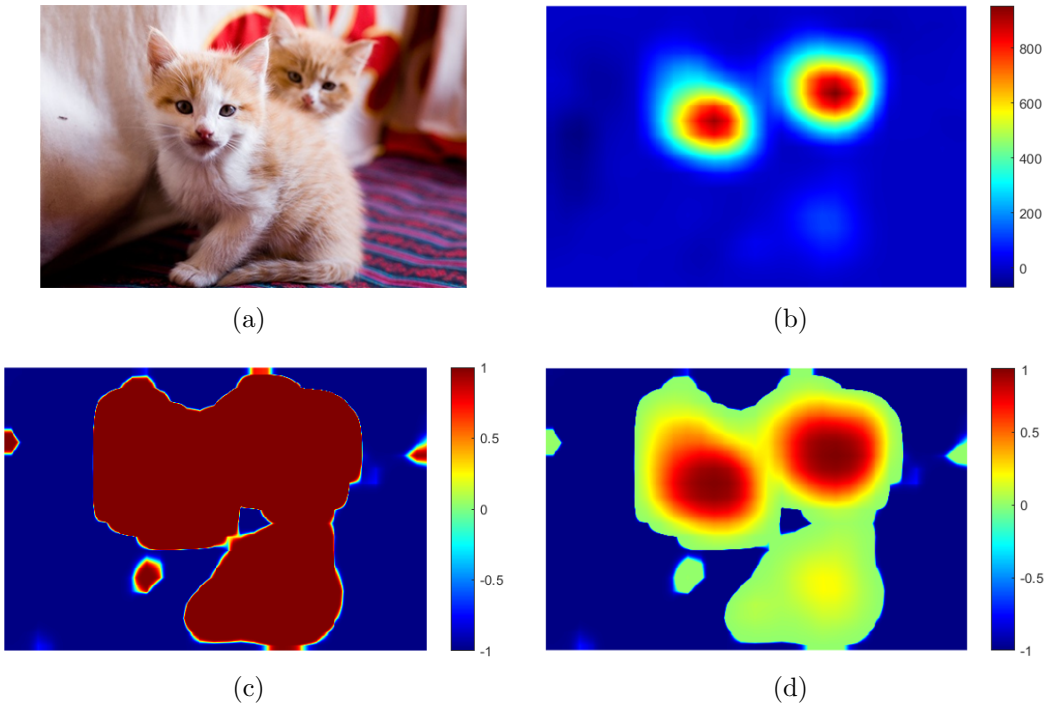


Figure 4.4: Illustration of the adaptive tanh activation function: (a) the original image, (b) the feature map for the class cat, the response map obtained by (c) applying the original tanh function and (d) the adaptive tanh function on the feature map.

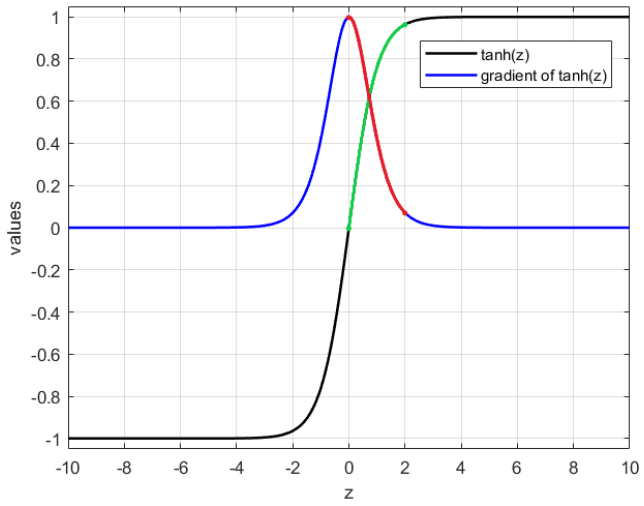


Figure 4.5: Illustration of gradient-sensitive regions of the tanh function for positive input. The red and green curves indicate the corresponding output of tanh and its gradients for regions with high gradient sensitivity.

The response score $\boldsymbol{r}^c$, for class c, is obtained by averaging the values from response map $\boldsymbol{R}^c$:

$$\boldsymbol{r}^c = \sigma(\text{GAP}(\boldsymbol{R}^c)), \tag{4.5}$$

where $\sigma(\cdot)$ is a sigmoid activation function and GAP represents a global average pooling process.

Then, we propose a new loss function as:

$$L_{rs} = -\frac{1}{l}\left(\sum_{c\in\mathcal{Y}} \boldsymbol{r}^c - \sum_{c\in\mathcal{O}\setminus\mathcal{Y}} \boldsymbol{r}^c\right). \tag{4.6}$$

The first term in the proposed loss function L_{rs} aims to enhance the response map $\boldsymbol{R}^c$ by increasing the response score $\boldsymbol{r}^c$ for the class $c \in \mathcal{Y}$ relevant to the image $\boldsymbol{I}$. It results in larger activated regions in the feature map $\boldsymbol{D}^c$ for corresponding class. Meanwhile, the second term suppresses the scores for other irrelevant classes to reduce activation in the corresponding feature maps.

Branch 2: Training with the weighted class score

The second branch is designed to limit the transmission of information in class-wise feature from the discriminative regions by using a new weighted pooling method, i.e., Response Weighted Pooling (RWP). The intuition of the Response Weighted Pooling is to assign weights inversely proportional to the numerical values on the class-wise feature when producing the class score, assigning lower weights to discriminative regions and higher weights to less-discriminative regions, thereby leading the classification model to give more importance to less-discriminative regions when training.

Specifically, a weighted map $\boldsymbol{V}^c$ for class c is obtained by:

$$\boldsymbol{V}_i^c = \min(1 - \boldsymbol{R}_i^c, 1), \tag{4.7}$$

where $\boldsymbol{V}_i^c \in [0, 1]$ represents the weight value at location index i of class c, with $i \in \mathcal{U}$, and $\mathcal{U}$ is the a set of all spatial location indexes in $\boldsymbol{V}^c$. $\boldsymbol{R}$ is the response map obtained from the first branch.

Figure 4.6.(a) shows a weighted map generated from the response map in Figure 4.4.(d). It is observed that, for the regions in the response map with high numerical values (for example, the regions corresponding to the center of the faces of cats in Figure 4.4.(a)), the corresponding regions in the weighted map have low values. Besides, in regions where the response map has negative values, the corresponding values in the weighted map are all uniformly set to 1.

Then, the weighted class score $\boldsymbol{v}^c$ for class c is computed by averaging the element-wise product of the feature map $\boldsymbol{D}^c$ with the weighted map $\boldsymbol{V}^c$:

$$\boldsymbol{v}^c = \sigma\left(\frac{1}{|\mathcal{U}|}\sum_{i\in\mathcal{U}} \boldsymbol{D}_i^c \boldsymbol{V}_i^c\right), \tag{4.8}$$

where $\sigma(\cdot)$ is a sigmoid activation function.

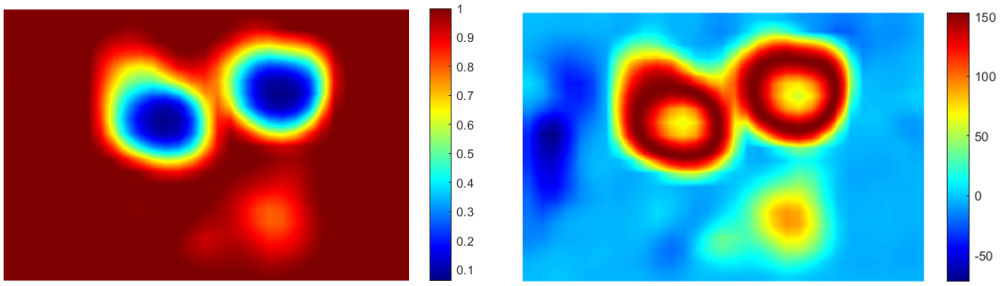
Figure 4.6.(b) shows weighted feature map, i.e., the element-wise product results of the weighted map shown in Figure 4.6.(a) with the feature map shown in Figure 4.4.(b). It is observed that, the less-discriminative regions (likes the regions corresponding to the surrounding areas of the faces and the bodies of cats in Figure 4.4.(a)) in the original feature map become more significant in the weighted feature map, while these regions now play a more crucial role in the computation of the weighted class score $\boldsymbol{v}$.

The weighted class score $\boldsymbol{v}$ and image-level label $\mathcal{Y}$ are used to compute a penalization term using the multi-label cross entropy loss function $L_{cls_v}(\boldsymbol{v}, \mathcal{Y})$ in the second branch:

$$L_{cls_v}(\boldsymbol{v}, \mathcal{Y}) = -\frac{1}{l}\left(\sum_{c\in\mathcal{Y}} \log(\boldsymbol{v}^c) - \sum_{c\in\mathcal{O}\setminus\mathcal{Y}} \log(1-\boldsymbol{v}^c)\right). \tag{4.9}$$

The first term in the loss function L_{cls_v} encourages the model to predict high scores to positive classes related to the image, which encourage the values in the less-discriminative regions of feature map to be further increased. The second term globally decrease the values in the feature map for other negative classes which are not related to the image.

Training with this branch, the classification network is forced to mine more valuable information from the less-discriminative regions to make the prediction, and hence the activation regions in the class-wise feature map are enlarged during training for the present classes.



(a) (b)

Figure 4.6: Illustration of (a) a weighted map and (b) the element-wise product of the weighted map with corresponding class-wise feature for the example image shown in Figure 4.4.

For the proposed training method with two additional branches, the total loss function L is defined as:

$$L = L_{cls_s}(\boldsymbol{s}, \mathcal{Y}) + \beta L_{rs}(\boldsymbol{r}, \mathcal{Y}) + \gamma L_{cls_v}(\boldsymbol{v}, \mathcal{Y}), \tag{4.10}$$

where $L_{rs}(\boldsymbol{r}, \mathcal{Y})$ and $L_{cls_v}(\boldsymbol{v}, \mathcal{Y})$ are the loss functions used in the branch 1 and branch 2, respectively. β and γ are the hyper-parameters to adjust the weights for new proposed loss functions.

4.4 Experimental results

In this section, we present the experimental settings for this chapter, including the datasets, evaluation metrics, and implementation details. Following this, we show both qualitative and quantitative experimental results for the methods proposed in Section 4.3.1 and Section 4.3.2. Based on the results, we analyze the strengths and challenges of each method.

4.4.1 Experiments settings

Datasets and evaluation metric

The well known PASCAL VOC 2012 dataset [18], which has 20 foreground classes and 1 background class, is used to carry out the experiments. The dataset is split, according to the common procedure, into train, validation and test sets, which contain 1464, 1449 and 1456 images, respectively. Additional training data are provided by the Semantic Boundary Dataset [147]. By combining the initial training set and these additional data, the augmented train set, consisting of 10582 images in total, is created. It is called augmented train set in what follows.

In experiments, we evaluate the seed masks (see Section 3.3.2) by using the mean Intersection-over-Union metric for the proposed methods. In addition to scalar results, visualization results for the enhanced CAM and the obtained seed are also provided.

Besides, the commonly used accuracy metric for multi-label classification is employed to evaluate the improvement for the classification network trained by the proposed new training methods. Specifically, the output class score $\boldsymbol{s}$ is activated by a sigmoid function, and indices where score values are larger than 0.5 are considered the predicted labels for given images. This metric is calculated as the ratio of the number of images with all labels correctly predicted to the total number of images in the dataset.

Implementation Details

The baseline CAM is obtained from a trained classification network following the training pipeline in [106], detailed in Section 3.3.1. In our experiments, we use the layers before the first fully connected layers in VGG-16 [65] as the backbone

network and adds three convolutional layers with convolutional kernels' size of 3×3 and one convolutional layer with convolutional kernels' size of 1×1. The weights of the backbone network are initialized with the VGG weights pre-trained on ImageNet [69]. The additional convolutional layers are randomly initialized. We train the classification network using the mini-batch gradient descent optimizer with a batch size of 16 images. The classification network is trained during 30 epochs. The initial learning rate is set to $1e-3$ and it is decreased by a factor of 10 at epoch 5 and 10. We apply random crop, random horizontal flipping, and random color jittering as data augmentation processes for training the classification network. We empirically set $T_{fg} = 0.2$ to obtain the seed. In the method proposed in Section 4.3.2, we set $T_i = \text{arctanh}(0.9)$ and $k = 0.8|\mathcal{U}_+|$ to obtain τ in the adaptive tanh function. For the hyper-parameters to adjust the weights for the proposed loss functions in the total loss, we set α to $1e-2$, β to 1, and γ to 1 in practice.

4.4.2 Experimental results for enhanced CAM with decreased overlapping regions

The method proposed in Section 4.3.1 aims to obtain enhanced CAM without overlapping regions, which means the activation maps of different classes in the CAM are expected to be element-wise mutually exclusive. The experiment results of the enhanced CAM, referred to as EmeCAM, are presented in this section.

Figure 4.7 presents the comparison between baseline CAM obtained by [106] and the obtained EmeCAM for images with two objects. It is observed that, the overlapping regions are significantly decreased in the EmeCAM.

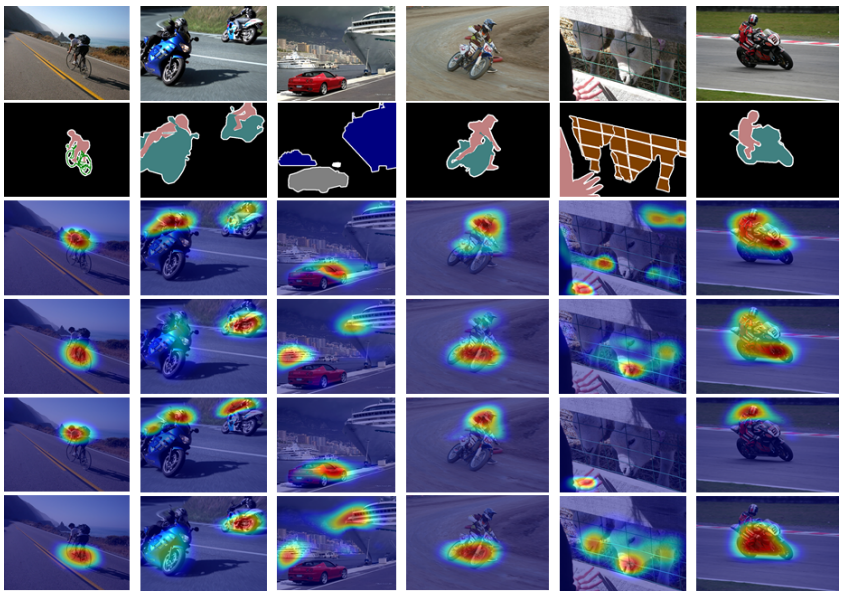


Figure 4.7: A comparison between the baseline CAM and EmeCAM for images from the PASCAL VOC 2012 train set with two classes. From top to bottom: original image, ground truth mask, the baseline CAM for the two classes, and the EmeCAM for the two classes with respect to the image.

As shown in the Figure 4.7, specifically, for images that contain both the person and motorbike classes, such as examples shown in the second, fourth, and sixth columns. The baseline CAM is not able to distinguish between these two categories, which results in substantial overlapping regions in the activation maps. The underlying reason for this limitation could be attributed to the co-occurrence of the motorbike and person classes in the most of images of the dataset, which causes the features the classification model can recognize for the motorbike class to be partially aggregated with those of the motorbike pilot. In the results obtained from EmeCAM, the overlapping regions are smaller, and the activation focuses more on the respective classes of each object. It indicates that by penalizing the existence of overlapping regions, the classification model gains a better recognition ability for these two classes.

Figure 4.8 shows the seed obtained from both the baseline CAM and the obtained EmeCAM by applying the same threshold value T_{fg}. It is observed that, our seed focuses more on the target object itself for the corresponding classes, with a reduced degree of mis-classification. However, even when the objects are adjacent in the image, the regions close to the boundary between different classes are incorrectly predicted as the background. This is likely because such region is more prone to being penalized as overlapping regions. The model tends to generate smaller responses in such adjacent areas between different objects, making it more likely to label pixels as background under the same threshold.

While the aim of EmeCAM is to eliminate the overlapping regions between activation maps, the model has also improved recognition ability for classes. This improvement can be observed in the enhanced recognition of objects in example images with a single class, as shown in Figure 4.9 and Figure 4.10. This improvement suggests that addressing inter-class overlaps not only clarifies distinctions between classes but also strengthens the model's recognition capabilities for the objects.

However, EmeCAM does not bring improvement across all classes. Due to the

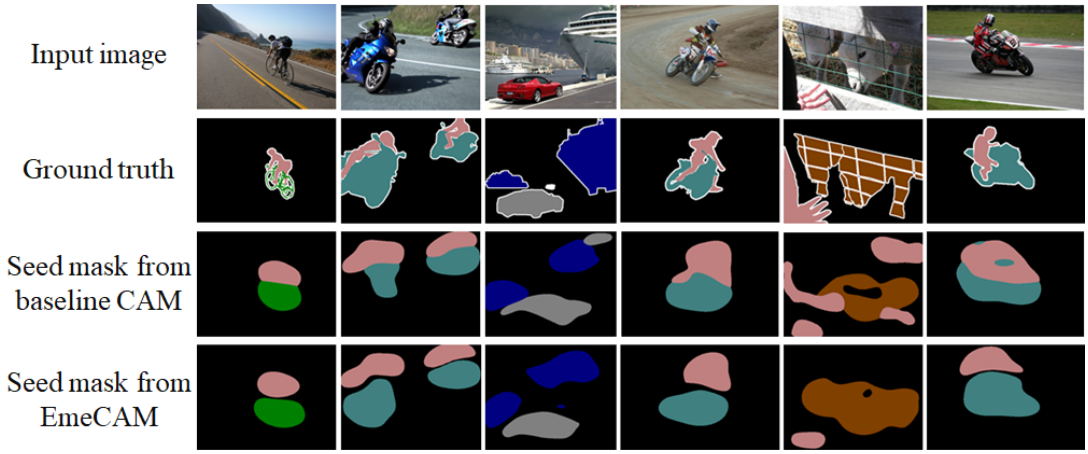


Figure 4.8: A comparison between the seed from the baseline CAM and EmeCAM by applying a fixed threshold for the images from the PASCAL VOC 2012 train set with two classes. The images are the same examples in Figure 4.7.

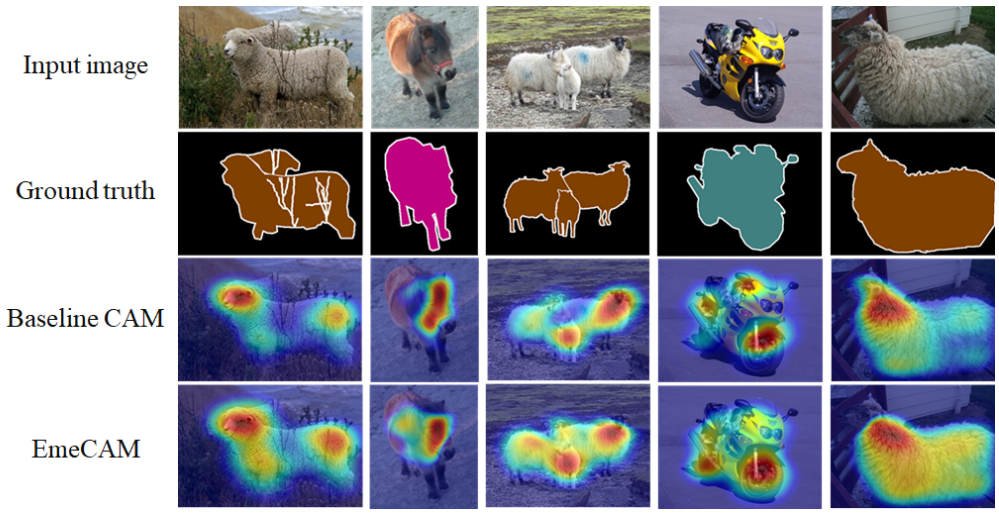


Figure 4.9: A comparison between the baseline CAM and EmeCAM for the images from the PASCAL VOC 2012 train set with single class.

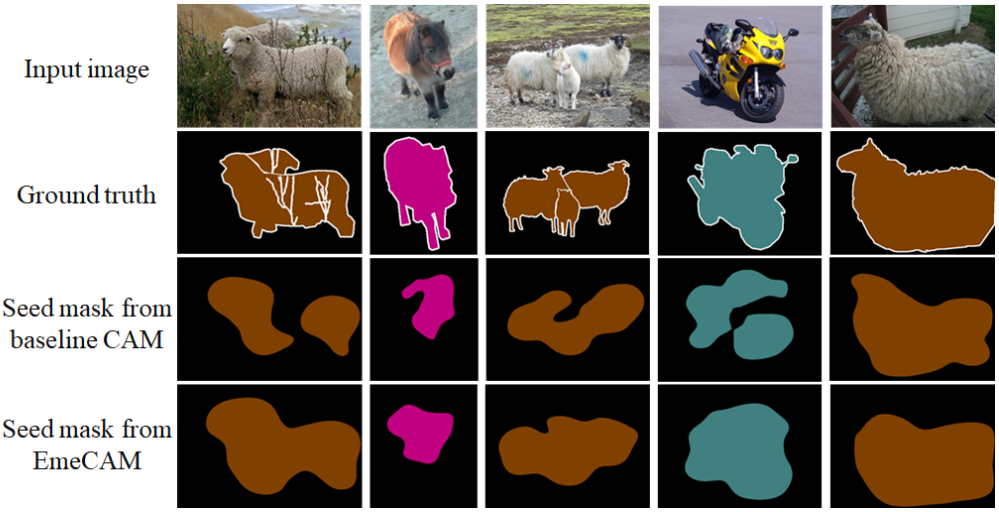


Figure 4.10: A comparison between the seed from the baseline CAM and EmeCAM by applying a fixed threshold for the images from the PASCAL VOC 2012 train set with single class. The images are the same examples in Figure 4.9.

penalization processed on overlapping regions during training, the model is forced to gain activation from other areas, which leads to a higher rate of false positive prediction for classes which are often accompanied by co-occurrence backgrounds. For example, in Figure 4.11, the first, third and fifth images illustrate this issue. Since the object boat often co-occurs with water in training images, when overlapping regions between boat and people are penalized, water (background) may be activated as boat, thus degrading the quality of CAM. Additionally, it is observed that EmeCAM causes the activation areas for some classes to shrink, which concentrate mostly on the confident regions and thereby reducing the overall recognition for the objects, as shown in the examples for the second and fourth images in Figure 4.11.

The impact of these observations have also consequences on the seeds obtained from EmeCAM, as shown in Figure 4.12. In these examples, compared to seeds ob-

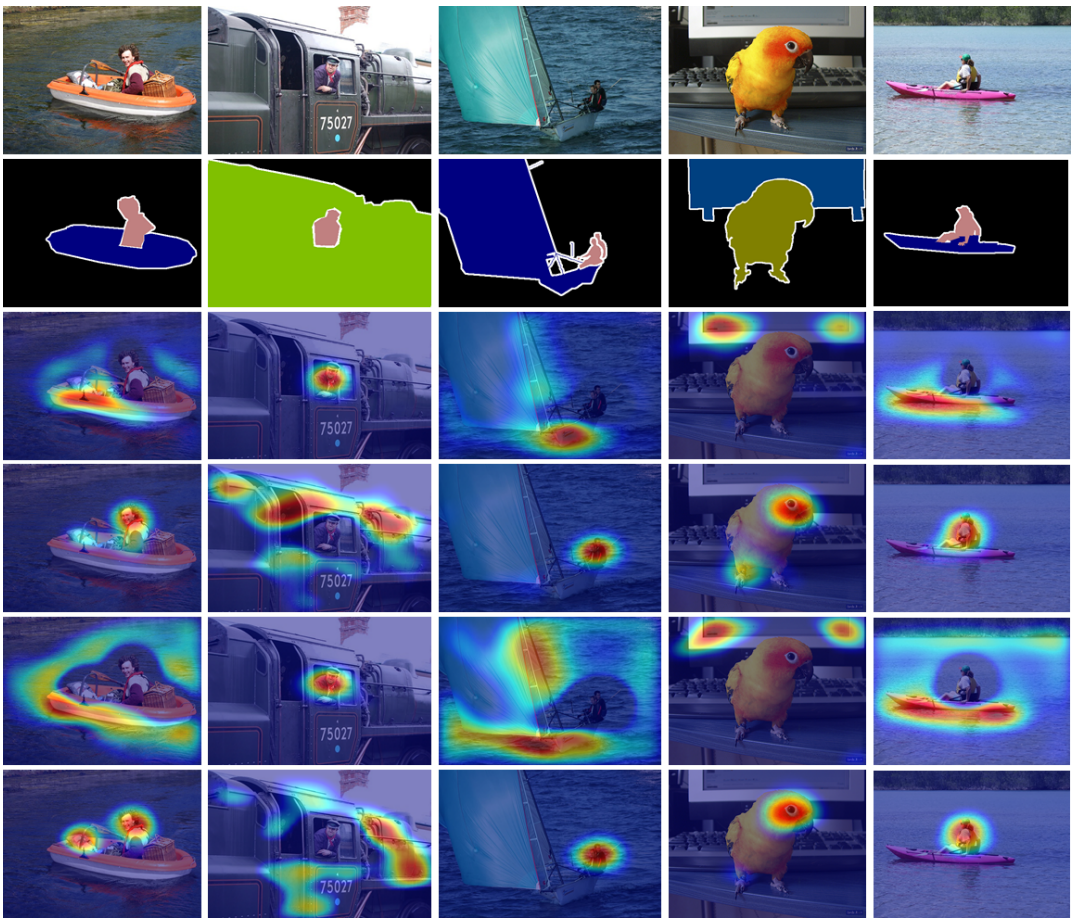


Figure 4.11: A comparison between the baseline CAM and EmeCAM for the images from the PASCAL VOC 2012 train set with two classes. From top to bottom: original image, ground truth mask, the baseline CAM, and the EmeCAM with respect to the image.

tained from the baseline method, the activation areas from seeds obtained through EmeCAM is less accurate, which causes a drop in the mIoU evaluation results.

Table 4.1 presents the comparison of quantitative results for the baseline and the proposed approach on PASCAL VOC 2012 train [18] and augmented train set [147], which reflects limitations of our method. Compared with the baseline results, the mIoU results of the seed obtained from the EmeCAM drops by 3.6% and 4.5% mIoU, with the train set and augmented train set, respectively.

We compare the classification accuracy for the network trained by the baseline

	train	augmented train
baseline	40.0	41.7
EmeCAM	36.4	37.2

Table 4.1: mIoU (%) of seed obtained from the baseline method and EmeCAM on the PASCAL VOC 2012 train and augmented train sets.

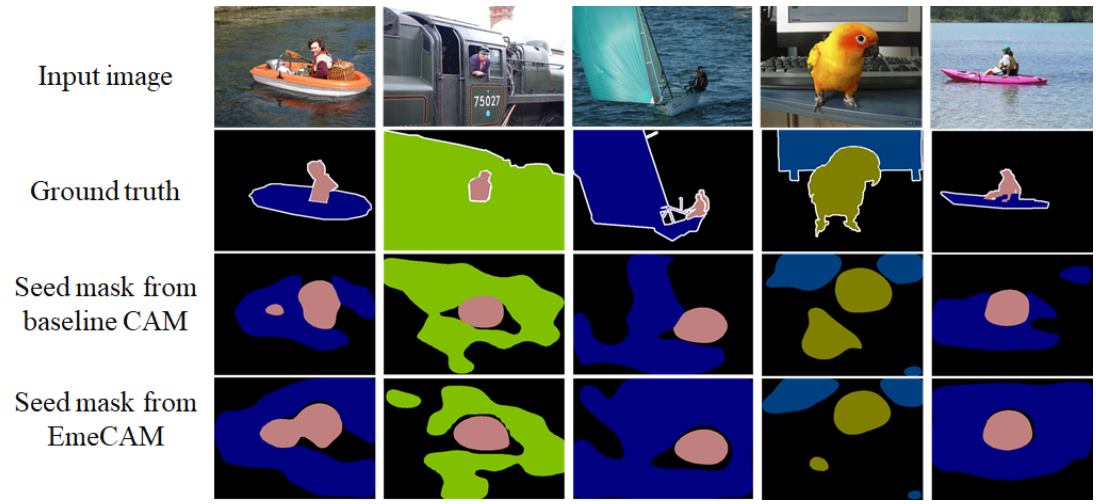


Figure 4.12: A comparison between the seed from the baseline CAM and EmeCAM by applying a fixed threshold for the images from the PASCAL VOC 2012 train set with single class. The images are the same examples in Figure 4.11.

method and the proposed method to decrease the overlapping regions on PASCAL VOC 2012 train [18] and augmented train set [147], as shown in Table 4.2. Compared with the baseline results, the classification accuracy results increases by 0.8% and 0.2%, with the train set and augmented train set, respectively. This demonstrates that the proposed method enhances feature discrimination and improves the recognition ability for certain classes. However, the overall improvement is modest.

In conclusion, experiments results prove that the EmeCAM generated from the proposed method is effective in decreasing the overlapping regions as expect. However, the seed generated from the EmeCAM are not improved in average accuracy. There is an increase in false activation in the background, leading to a decrease in accuracy. In addition, to avoid the penalization caused by overlapping regions during the training, the EmeCAM tends to focus intensively on the most confident regions, which reduces the size of the activation regions. Considering that a key issue with classic CAM is that it only focuses on the limited discriminative areas, the excessive concentration of EmeCAM on the most confident regions is an undesirable outcome. This is why we decide to not go further in the experiments and end with the fully pipeline of the WSSS. This also led us to develop the next method presented in Section 4.3.2.

	train	augmented train
baseline	72.3	75.6
EmeCAM	73.1	75.8

Table 4.2: The classification accuracy (%) of the model trained using the baseline method and the proposed method to decrease the overlapping regions (refereed to as EmeCAM in the table) on the PASCAL VOC 2012 train and augmented train sets.

4.4.3 Experimental results for enhanced CAM with enlarged activation regions

Considering the method proposed in Section 4.3.2 adds 2 branches to the standard classification pipeline, this section presents experimental results using only the first branch and using both branches. These two branches introduce new proposed loss functions (i.e., L_{rs} and L_{cls_v}) to activate less-discriminative regions in the CAM by reducing the contributions of large local values in the class-wise feature map during training.

Examples of the baseline CAM and the enhanced CAM produced by the method proposed in Section 4.3.2 are shown in Figure 4.13.

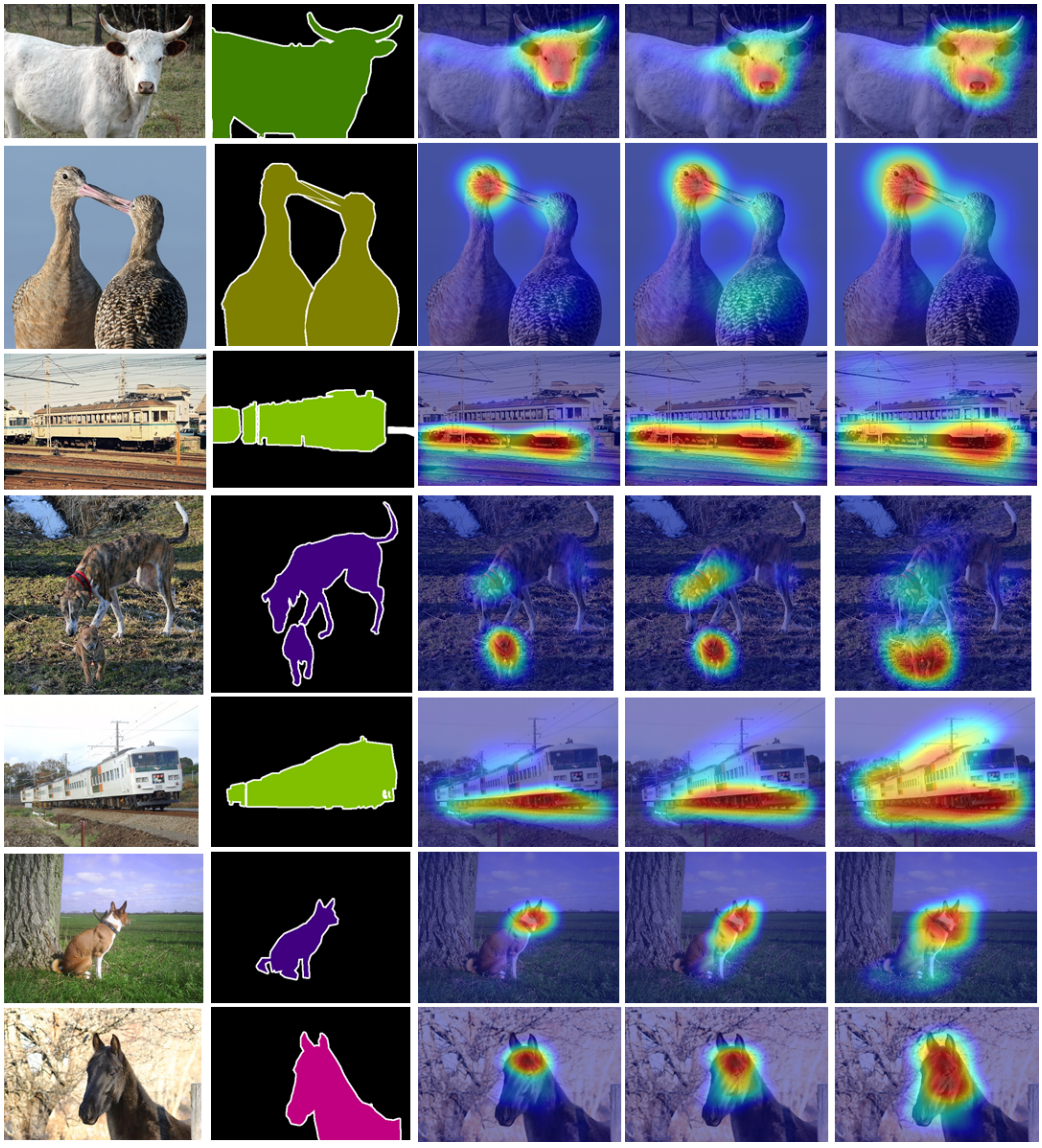


Figure 4.13: A comparison between the CAM from baseline and the proposed method for the images from the PASCAL VOC 2012 train set. From top to bottom: original image, ground truth mask, the baseline CAM, the CAM obtained where training with the first branch, and the CAM obtained where training with both two branches.

As shown in the Figure 4.13, the obtained CAM given by the proposed methods are shown in fourth and fifth rows where training with the first branch and both branches, respectively. It is observed that, the enhanced CAM covers larger parts of target objects, compared to the activation maps produced by the baseline method [106] as shown in the third row. Besides, training with both two branches can obtain the largest regions for the given images. It reveals that applying L_{cls_v} in the second branch can further lead the model to take into account more less-discriminative regions in the objects. For instance, as shown in the third column for the image with horse object, compared with the activation from the baseline CAM (in the third row), the enhanced CAM by training with first branch and both branch significantly increase the activation for the horse, with enlarged high-values regions (which are in red color). Similar improvement are observed in the other examples.

Figure 4.14 shows a comparison with the seed masks obtained from the baseline CAM and the seed masks from the obtained CAM given by the proposed method. However, this method encounters challenging issue with over-activation in the background, where significant parts of background regions are incorrectly predicted as foreground. The main reason for this effect is that both branches proposed in the method lead the model to increase activation in low-values areas in CAM.



Figure 4.14: A comparison between the obtained seed from the baseline and the proposed method by subjecting the CAM to a fixed threshold for the images from the PASCAL VOC 2012 train set. From top to bottom: original image, ground truth mask, the seed from the baseline CAM, the obtained seed where training with the first branch, and the obtained seed where training with both two branches.

These low-activation areas include not only the less-discriminative regions of the objects but also the background regions. Lacking sufficiently accurate guidance, the model inevitably increases the response in the background regions.

Therefore, as shown in Table 4.3, even though the model can generate larger regions to cover the object, their extend causes incorrect classification of the background that significantly impacts the final mIoU values. Compared to the baseline method, when using the first branch with the L_{rs} loss function, the mIoU results of seed drop 2.8% and 2.1% in the train and augmentation train sets, respectively. When training with both two branches, the larger over-activation in the background regions causes the mIoU results to drop by 5.7% and 2.8% in these two sets.

In fact, the over-activation of the background is also the main challenge for recent methods to activate less-discriminative regions [38, 39, 40, 124, 148]. Similar to our approach, as the activation of low-values regions in the CAM increases, the occurrence of false activation becomes inevitable. In these methods, a common practice to address the false positive predictions in the background is to use saliency as the cues to define the background (see Chapter 3). Specifically, the values of saliency map for a given image are in the range of [0, 1]. Recent methods assign all locations where saliency values are lower than a fixed threshold T_{sal} to the background labels when obtaining seed from the CAM, regardless of whether the CAM values are higher than T_{fg} or not. In this case, since the regions with low saliency values often correspond to the background, even if the obtained CAM has over-activation in these background areas, those regions are not labeled as objects.

In practice, we also evaluate the performance of seed using saliency as background cues. We use the saliency produced from the method in [149], and set $T_{sal} = 0.06$ as the same value in the related methods [38, 39, 124]. It is evident that with the assistance of saliency, the accuracy of the seed is significantly enhanced, showing clear boundaries between objects and the background, as shown in Figure 4.15.

However, we argue that, simply using saliency to provide background information covers the drawbacks of these methods. The occurrence of over-activation fundamentally indicates that the classification network can not accurately identify target features of objects during training. The desired increase in activation should come from the less-discriminative regions in objects (e.g., the body of the animals), rather than increasing incorrect activation in the background. Besides, as presented in Chapter 3, the use of saliency usually requires extra supervision beyond image-level labels and training additional saliency detection models, which

baseline	L_{rs}	L_{cls_v}	train	augmented train
✓			40.0	41.7
✓	✓		37.2	39.6
✓	✓	✓	34.3	38.9

Table 4.3: mIoU (%) of the seed obtained by model trained using the baseline method and considering ablation with the proposed method to enlarge activation regions on the PASCAL VOC 2012 train and augmented train sets.



Figure 4.15: A comparison between the seed obtained from baseline and the proposed method when using saliency as the background cues for the images from the PASCAL VOC 2012 train set. From top to bottom: original image, ground truth mask, the seed from the baseline CAM, the obtained seed where training with the first branch, and the obtained seed where training with both two branches.

increase the level of supervision in the whole task, and cost more time and computation.

The classification accuracy for the network trained using the baseline method and the proposed method to enlarge activation regions is shown in Table 4.4. Compared to the baseline results, adding only branch 1 and using L_{rs} for classification training already improves accuracy by 1.5% on the train set. However, when both branches are added, the classification performance significantly improves, with increases of 5.8% and 5.3% on the train set and augmented train set, respectively. It proves that, through the proposed method of enlarging activation regions, the network's response to the labeled classes in the training images is significantly enhanced.

In summary, experiments results show that the method proposed in Section 4.3.2 works effectively in enlarging the activation regions in the CAM. The over-activation issue inevitably increase the false positive prediction in the background regions, which causes a drop in the accuracy of the obtained seed. Considering we did not get the expected improvement for the seed masks from the obtained CAM, we cannot expect that using these masks to train the segmentation model will bring significant improvements. Therefore, we do not complete the last steps in the WSSS training pipeline, as presented in Chapter 3. While using saliency data can mitigate such mislabeling, it covers the inherent limitations of the methods and

baseline	L_{rs}	L_{cls_v}	train	augmented train
✓			72.3	75.6
✓	✓		73.8	75.4
✓	✓	✓	78.2	80.8

Table 4.4: Classification accuracy (%) of the model trained using the baseline method and considering ablation in the proposed method to enlarge activation regions on the PASCAL VOC 2012 train and augmented train sets.

is not cost-free. Therefore, identifying strategies to activate non-discriminative regions without misclassifying background areas as objects remains an important area to explore for the future development of WSSS.

4.5 Conclusion

In this chapter, we propose two new methods to modify the training of the classification network, in order to obtain enhanced CAM. Specifically, in the first method, we obtain enhanced CAM which is expected to exclude the overlapping regions where multiple significant activation values appear to different classes. A new loss function is proposed to penalize the occurrence of the overlapping regions by orthogonalizing the vectors reshaped from activation maps for different classes. To effectively increase the activation in the less-discriminative regions for objects in the images, the second method is proposed with two additional branches for training the classification model, which aim at suppressing significant contributions of the high response values in the class-wise feature map for the training.

Experiment results show that, our methods can achieve those targets as expected. However, both two methods also face new challenges, which are different or even opposite. In the first method, the penalization of overlapping regions causes the activation values decrease in the object regions, which leads the regions of objects mislabeled as background. In the second method, an amount of pixels are incorrectly labeled as objects since the over-activation appears in the background regions. As a result, we choose not to complete the remaining steps in the WSSS training pipeline, which involves obtaining pseudo-masks from the CAM and training a segmentation network, as presented in Chapter 3. This decision is made because the accuracy of the seed derived from the obtained CAM does not improve as expected, making the completion of further steps in the pipeline less beneficial.

Although the methods proposed in this chapter do not yield particularly ideal results from a numerical standpoint, several insights observed in the experiments: Firstly, it is observed that, relying solely on low-resolution class-wise feature features generated from the the last convolutional layer, CAM is suffered from blurry recognition of objects. Secondly, there is a lack of effective modeling and utilization of background information. Thus methods which aim at activating less-discriminative areas inevitably increase the incorrect activation in background re-

gions. Thirdly, since the training of the classification networks is a global process, the generation of CAM lacks consideration for image-specific features. Even for images that intuitively seem simple to segment, such as those with clear foreground-background differences, CAM does not demonstrate better segmentation performance.

Therefore, in the next chapter, a new image-specific approach is introduced, which aims at obtaining improved pseudo-masks from CAM, by using both deep class-wise feature with sufficient semantics and shallow features in high resolutions. It is worth mentioning that, in the method proposed in this chapter, using saliency data to determine the boundaries between objects and background has significantly improved the performance of the seed in practice. However, using such auxiliary data usually requires extra supervision and more computation cost. Therefore, the approach proposed in the next chapter is designed to rely solely on image-level labels without using saliency and other auxiliary data, which sticks to the goal of our work.

Chapter 5

A multi-feature image-specific pixel classification approach for pseudo-masks improvement

Contents

5.1 Introduction 85
5.2 Motivation 86
5.3 Proposed method 87
5.3.1 The generation of seed 88
5.3.2 Sample selection for pixel classifier training 89
5.3.3 Self-enhanced SVM 91
5.4 Shallow feature selection and method's performance limit 91
5.4.1 Experimental settings 92
5.4.2 Experiment results 94
5.5 Experiments 95
5.5.1 Experimental settings 95
5.5.2 Choice of the shallow feature and the iteration number in the self-enhanced SVM 96
5.5.3 Ablation studies 97
5.5.4 Comparison with state-of-the-arts 102
5.6 Conclusion 104

5.1 Introduction

As mentioned in Chapter 3, the segmentation performance of WSSS heavily depends on the quality of pseudo-masks. The pseudo-masks are produced by a two-step process: the generation of CAM, followed by the inference of pseudo-masks

from CAM. In the previous chapter, we introduced methods to obtain enhanced CAM. In this chapter, we mainly focus on the second step, which aims at improving pseudo-masks from CAM. In Chapter 3, we presented typical expansion methods [25, 41, 42] and refinement methods [28, 29, 116]. The expansion methods assume that, to infer labels for all pixels in the image, discriminative regions in the CAM contains enough reliable information. The refinement methods are commonly used as post-processes to further improve the accuracy of the pseudo-masks in recent WSSS approaches. In this chapter, a new expansion method based on a multi-feature image-specific pixel classification approach is proposed. By integrating refinement processes to optimize the outputs of the expansion method, accurate pseudo-masks are finally produced. Specifically, Section 5.2 gives the motivation of the proposed approach, Section 5.3 presents the proposed method in detail. In Section 5.4, we implement a study on the shallow feature selection and demonstrate the potential of the proposed method by evaluating its upper performance limit. Section 5.5 provides the experimental settings and substantial results for the proposed expansion method. Section 5.6 gives a conclusion of this chapter.

5.2 Motivation

As mentioned in Chapter 3, new methods have been proposed to obtain improved pseudo-masks from the CAM [25, 26, 27, 28, 29, 41, 116]. Among these methods, we observe that two limitations are not adequately addressed: Firstly, most methods typically rely on either low-resolution deep features with rich semantics or on the original image, which is in high resolution but with limited semantics. Low-resolution features can result in smoothed boundaries and missing details, while insufficient semantic features can lead to noisy and inaccurate predictions in pseudo-masks. Secondly, specific features of each image that should be taken into account are not fully explored when producing pseudo-masks.

Therefore, we introduce an expansion method based on Multi-feature Image-specific Pixel classification approach (MIP). The target for the proposed method is to expand the confident and reliable location information provided by CAM into non-confident regions and infer the corresponding pixel labels. Unlike methods that do not fully take into account the particularities of each image, the proposed approach introduces a pixel classifier which training is specific for each image in the training dataset. The proposed approach uses information given by CAM, seed and shallow features. The shallow features generated by the early convolutional layers of the classification network provide image neighborhood information. These early layers have more semantics than the corresponding image region itself, while maintaining a much higher level of details than the deeper layers. Since CAM and seed are generated from deep features, our approach also benefits from rich semantic information.

Figure 5.1 gives the detailed pipeline of the whole process. The MIP expansion method is applied once the CAM has been generated. Note that it can be used after any approaches designed to improve classification model and to obtain enhanced CAM in the WSSS framework. The MIP requires training an image-specific

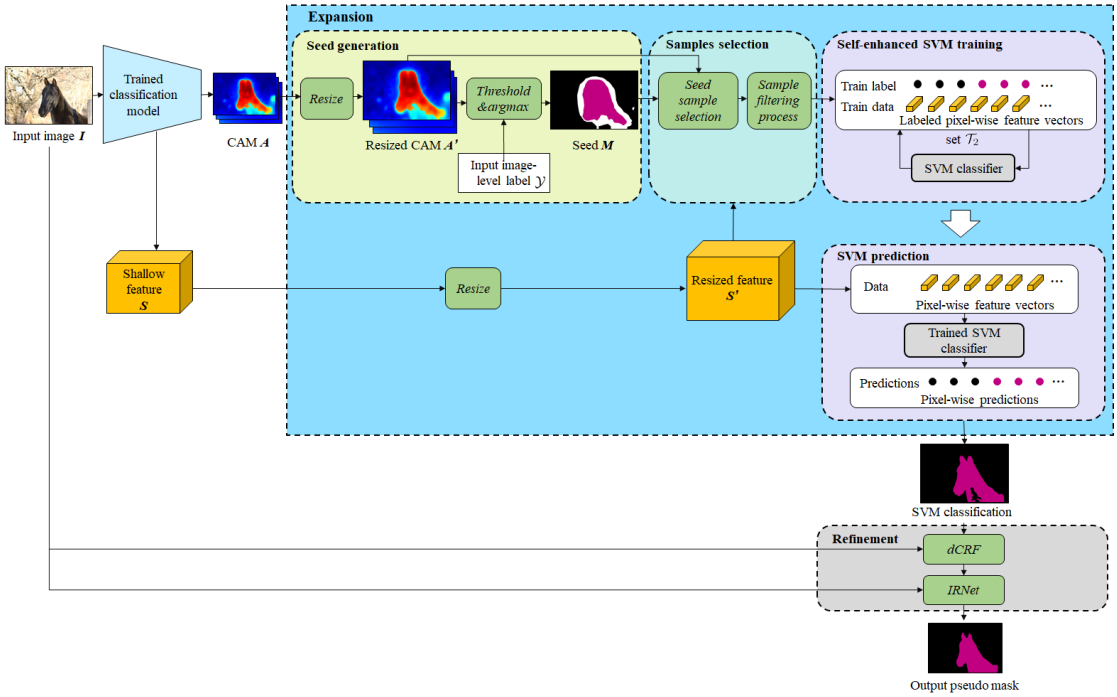


Figure 5.1: Detailed pipelines of the overall framework of the proposed method.

classifier. In practice, a Support Vector Machine (SVM) has been trained for each image. To do so, a training set is needed for each image and is defined after a sample selection process. Specifically, image selected samples are drawn from the shallow features with prior probability estimation deduced from the CAM and the seed masks. The seed masks are obtained by thresholding the CAM by two fixed thresholds, for foreground and background respectively. The assumption is that, the CAM higher values designate the most reliable samples, and the seed their labels for the targeted object, while the lower CAM values can provide more reliable indicators to describe background areas. Then, we propose a self-enhanced training strategy for SVM. The main idea is to sequentially update the labels of the samples according to previous SVM predictions and then train again the SVM classifier. Note that the training of the SVM is an off-line process to generate pseudo-masks in the annotation phase, which has no impact in terms of computational time when inferring the segmentation results for the test images. Following the common practices, a refinement process, including dCRF [116] and IRNet [29], can also be applied to further enhance the output of the expansion method to get the pseudo-masks. Finally, all images and their pseudo-masks are used to train the supervised semantic segmentation model.

5.3 Proposed method

In this section, the new expansion method based on multi-feature image-specific pixel classification is introduced. It involves three steps. Firstly, the proposed approach begins with generating seed from the CAM. Then, a sample selection process, that uses multi-feature: CAM, seed and shallow features, is proposed. It

consists in a sample selection and an additional process to exclude samples which are too similar to pixels labeled as background. At the end of these 2 operations, a train set for a pixel classifier is defined. In this study, we consider SVM as the pixel classifier. Finally, to improve the label quality, we propose a self-enhanced SVM training process that consists to iteratively correct the labels of the pixels in the SVM training set according to the SVM classification results (trained using the previous labels). The predictions of the last trained SVM provide the pixel-wise classification mask, which can be directly used as pseudo-masks or as input of the refinement process for obtaining the further improved pseudo-masks. These three steps are detailed in the next subsections.

5.3.1 The generation of seed

As shown in Figure 5.1, the first step in the proposed method begins with generating seed from CAM. Since the proposed expansion method aims at generating improved pseudo-masks from the CAM, it can use the outputs from any methods focused on obtaining enhancing CAM, without changing the baseline method.

In Chapter 3 and Chapter 4, the seed masks are generated by applying a fixed threshold and an argmax process to the resized CAM. In practice, each location in the seed masks is assigned to either an object label or the background. However, the accuracy of the masks is highly sensitive to the fixed threshold. If this threshold is too high, then the samples will only be representative of the most discriminating part of the object and not of the whole object. If it is too low, the risk of selecting samples that do not belong to the object's class increases. Unlike this method of seed generation, following [41, 113], the proposed expansion method chooses to use two thresholds to divide the labels on the seed masks into three groups: background, objects, and uncertain regions. This method ensures that the objects labels in the generated seeds are the regions with high values in the CAM, and those of background have low CAM values for all categories, meaning they produce a less ambiguous response, and thus they are assumed to be more reliable.

Specifically, to generate the seed, a bilinear interpolation process is firstly used to resize the CAM $\boldsymbol{A}$ to $\boldsymbol{A'}$, with the spatial size of $\boldsymbol{I}$. Then, the seed $\boldsymbol{M}$, which is the prediction with the same size as the image $\boldsymbol{I}$, is obtained by thresholding $\boldsymbol{A'}$, the resized CAM, with T_{fg} and T_{bg} the foreground and background thresholds, respectively, and an argmax process:

$$\boldsymbol{M}_i = \begin{cases} 0 & \text{if } \max\limits_{c\in\mathcal{Y}}(\boldsymbol{A'}_i^c) < T_{bg}, \\ \arg\max\limits_{c\in\mathcal{Y}}(\boldsymbol{A'}_i^c) & \text{if } \max\limits_{c\in\mathcal{Y}}(\boldsymbol{A'}_i^c) > T_{fg}, \\ -1 & \text{otherwise,} \end{cases} \tag{5.1}$$

where $\boldsymbol{A'}_i^c$ is the pixel value at the location index i, with $i \in \mathcal{U}$ the set of all location indexes in $\boldsymbol{A'}^c$ of the resized CAM $\boldsymbol{A'}$ for class c. $\boldsymbol{M}_i$ is the label prediction at location i. The label $\boldsymbol{M}_i$ is larger than 0 for foreground, equal to 0 for background,

and -1 when the label is too uncertain. The latter occurs mainly near object boundaries.

5.3.2 Sample selection for pixel classifier training

As mentioned, CAM does not highlight the entirety of the objects in the image, but only the discriminating regions. In addition, it defines imprecise and blurred boundaries between foreground and background. Consequently, based on CAM, the initial seed $\boldsymbol{M}$ is not very accurate and is too noisy to be used as a pseudo-mask. However, even if CAM and seed are not perfect, they still provide valuable information that can be used to select a reduced set of reliable labeled pixels. This reduced set can be used to train a pixel classifier, like support vector machine, to improve the accuracy of pixel labels.

Let $\boldsymbol{S}$ be the output of a layer of the backbone network (i.e., a shallow feature) and $\boldsymbol{S'}$ its resized version to the original image size. We define the complete labeled dataset $\mathcal{T}_0 = \{(\boldsymbol{S'}_i, \boldsymbol{M}_i) | i \in \mathcal{U}\}$, where $\boldsymbol{S'}_i$ is the pixel-wise feature vector from the resized shallow feature $\boldsymbol{S'}$, and $\boldsymbol{M}_i$ is pixel-wise inferred label from the seed value $\boldsymbol{M}$ at the location index i. Since the labels in $\mathcal{T}_0$ are provided by the seed $\boldsymbol{M}$, they are not entirely accurate. The proposed approach aims to select a reduced training set from $\mathcal{T}_0$ that is as label-accurate as possible to train a pixel classifier, and this classifier will then be used to infer pixel-wise labels across the whole image using the data $\boldsymbol{S'}$ from $\mathcal{T}_0$. The selection of the training set is a 2-step process described in the next two subsections.

Seed samples selection

A first subset $\mathcal{T}_1$ of $\mathcal{T}_0$ is defined by selecting samples from $\mathcal{T}_0$. Different rules for background and foreground selection are applied.

In experiments, we have observed that the precision of background prediction given by seed $\boldsymbol{M}$ is larger than 90%. Thus, for the background, n training samples are drawn using uniform random sampling from pixels whose seed values are 0.

Labeling foreground from seed seems to be less accurate by far. To select foreground samples with more reliable labels, the idea is to focus on the most discriminant regions of the CAM. To achieve this, weighted random sampling of the foreground is implemented to give more weights to pixels with large CAM values. The weight $\boldsymbol{P}_i^c$ at location i is determined according to:

$$\boldsymbol{P}_i^c = \frac{\boldsymbol{A'}_i^c - T_{fg}}{1 - T_{fg}}. \tag{5.2}$$

n samples are drawn for each class. Thus, $\mathcal{T}_1$ contains $(|\mathcal{Y}| + 1)n$ samples defined as pairs $(\boldsymbol{S'}_k, \boldsymbol{M}_k)$, at selected location $k \in \mathcal{K} \subset \mathcal{U}$.

Train set sample filtering

A more detailed analysis of $\mathcal{T}_1$ reveals that, in some images, a significant number of the samples are incorrectly labeled, mainly due to background labels being attributed to the foreground samples. Thus, an additional filtering operation is added to reduce the number of such errors. It consists in removing from $\mathcal{T}_1$ the samples labeled as foreground which are too similar to background.

Since the background is relatively well labeled in $\boldsymbol{M}$ and has little specific semantic, it can be represented by its empirical mean $\boldsymbol{S'_{bg}}$:

$$\boldsymbol{S'_{bg}} = \frac{1}{n} \sum_{k \in \mathcal{K}} \boldsymbol{S'}_k \mathbb{1}_{(\boldsymbol{M}_k = 0)}, \tag{5.3}$$

where $\mathbb{1}_{(\cdot)}$ is an indicator function, which equals 1 when the condition is satisfied and 0 otherwise. In this case, $\boldsymbol{S'_{bg}}$ serves as a prototype of background for comparison.

To measure the similarity $\boldsymbol{w}_k$ between a sample from the foreground (i.e., $\boldsymbol{M}_k > 0$) and the background, a simple cos-similarity between the sample's data $\boldsymbol{S'}_k$ and $\boldsymbol{S'_{bg}}$ is used:

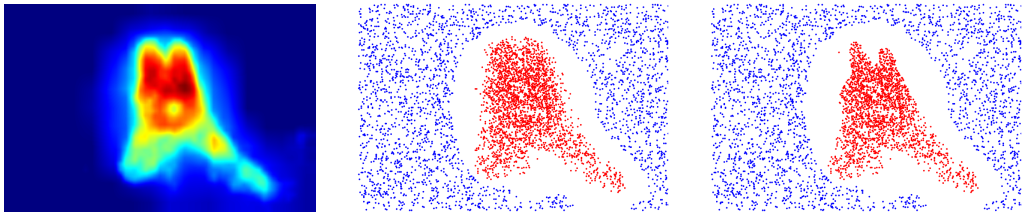
$$\boldsymbol{w}_k = \frac{\boldsymbol{S'^{\top}_{bg}}\, \boldsymbol{S'}_k}{||\boldsymbol{S'_{bg}}||\ ||\boldsymbol{S'}_k||}, \tag{5.4}$$

where $\boldsymbol{S'^{\top}_{bg}}$ is the transpose of $\boldsymbol{S'_{bg}}$.

When $\boldsymbol{w}_k > T_s$ (a threshold to be fixed), foreground sample at location k shares too much similarity with background prototype, which might correspond to an incorrect prediction and is excluded from the dataset.

As the result of the whole process, a subset $\mathcal{T}_2 = \{(\boldsymbol{S'}_m, \boldsymbol{M}_m) | m \in \mathcal{M}\}$ of $\mathcal{T}_1$ is obtained, where $\mathcal{M} \subset \mathcal{K}$. This set $\mathcal{T}_2$, whose size is smaller than $(|\mathcal{Y}| + 1)n$, is then used to train the SVM classifier.

Figure 5.2 gives an example to illustrate the 2-step selection process. Figure 5.2.(a) is the resized activation map for class horse for the input image shown in Figure 5.1. Figure 5.2.(b) visualize the location set $\mathcal{K}$ for the first selected subset $\mathcal{T}_1$, where the red points represent the locations with horse labels, and the blue points indicate the background. It is observed that the density of selected points correlates with the values in the corresponding areas of the activation map. Figure 5.2.(c) visualize the location set $\mathcal{M}$ for the second selected subset $\mathcal{T}_2$. By using train set sample filtering process, locations in background regions with incorrect labels are excluded from the dataset. The detailed ablation study for the selection process is given in Section 5.5.3.



(a) (b) (c)

Figure 5.2: An example to illustrate the 2-step selection process with (a) the activation map for class horse for the input image shown in Figure 5.1, the visualized (b) location set $\mathcal{K}$ for the training set $\mathcal{T}_1$ and (c) location set $\mathcal{M}$ for the training set $\mathcal{T}_2$.

5.3.3 Self-enhanced SVM

It is obvious that the performance of each SVM classifier is related to the accuracy of the train labels. During our experiments, we found that the labels in $\mathcal{T}_2$ are still not accurate enough. However, we also observed that after training, inferring the $\boldsymbol{S}'_m \in \mathcal{T}_2$ can return more accurate labels, noted $\boldsymbol{M}_m^{\boldsymbol{SVM}}$.

Based on these observations, we propose a self-enhanced SVM training procedure, illustrated in Figure 5.1. The main idea is to update the labels in $\mathcal{T}_2$ using the output labels $\boldsymbol{M}_m^{\boldsymbol{SVM}}$ inferred by an SVM. Thus, we obtain a new $\mathcal{T}_2 = \{(\boldsymbol{S}'_m, \boldsymbol{M}_m^{\boldsymbol{SVM}}) | m \in \mathcal{M}\}$ used to train a new SVM classifier. The process is repeated several times, defining the self-enhanced SVM procedure.

Finally, after t repetitions, all the pixel-wise feature vectors of $\boldsymbol{S}'$ are classified thank to the last trained SVM. The predictions provide the SVM classification mask, which can be used as pseudo-masks or as input of the refinement process. We carry out an experiment to study the impact of selecting different t values on the results, which is detailed in Section 5.5.2.

5.4 Shallow feature selection and method's performance limit

In this section, we propose to evaluate the potential of the proposed method by evaluating its performance limit. Indeed, based on the description of the proposed method (Section 5.3), it is clear that the sample selection process is the critical step of the method. Thus, before detailing the method's performance through experiments, we propose to evaluate its upper bound by using the ground truth to define a free of error training set, which is used to train a SVM for labeling all image pixels.

The overall framework of these studies is presented in Figure 5.3. Compared with the proposed method, presented in Figure 5.1, we remove the seed generation step, and replace the sample selection process by a uniform random sampling based on

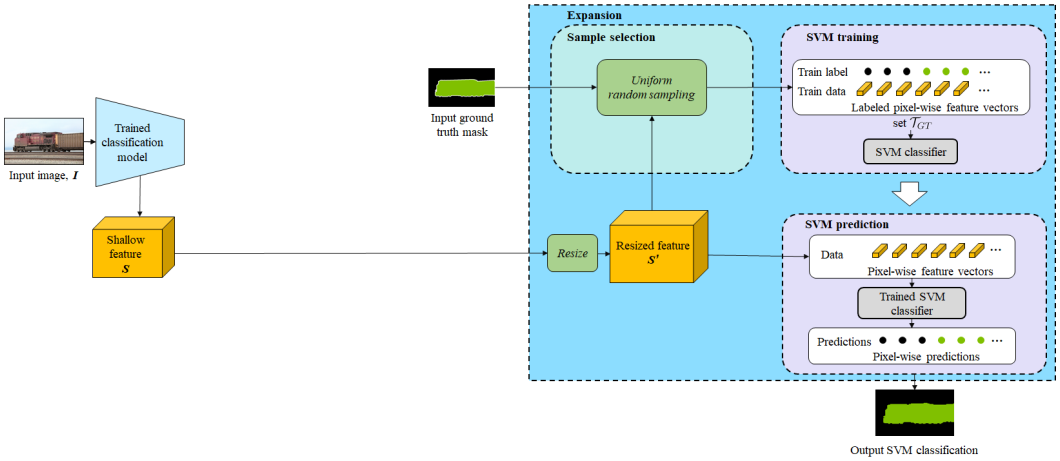


Figure 5.3: Detailed pipelines of the framework to evaluate the upper bound of the proposed method using ground truth masks.

the ground truth. The resulting training set $\mathcal{T}_{GT} = \{(\boldsymbol{S}'_j, \boldsymbol{L}_j) | j \in \mathcal{J}\}$, with $\mathcal{J} \subset \mathcal{U}$, made of n_{GT} pairs of shallow feature vectors and labels from the ground truth mask $\boldsymbol{L}$, used to train the SVM, is thus perfectly labeled. It is then no longer necessary to use the self-enhanced SVM training process. The refinement process is also removed since the goal is to evaluate the pixel classification performance at the end of the expansion.

Besides, since we use shallow features in order to balance resolution and semantics, we also need to evaluate which shallow features provide the best results when using the ground truth. A comparative study has been carried out that also provides insights for the choice of the feature used in the expansion method proposed in Section 5.3.

5.4.1 Experimental settings

Dataset and evaluation metrics

As in the previous chapter, the well known PASCAL VOC 2012 dataset [18], which has 20 foreground classes and 1 background class, is used to carry out the experiments. All the results in this section have been evaluated on the PASCAL VOC 2012 train set, which contains 1464 images in total.

Two metrics are used to evaluate the obtained SVM classification masks, which are the Intersection-over-Union per class and the mean Intersection-over-Union (mIoU).

Implementation Details

As mentioned in Section 5.3.1, the proposed method can be applied following any approach aimed at obtaining enhanced CAM. Considering one of the key targets of the study in this section is to evaluate the impact of using shallow features in the MIP method, while methods proposed in Chapter 4 mainly focus on the

deep feature generated from the last convolutional layers of the backbone network, we need to chose a baseline method that optimizes features from different layers. Analyzing literature, we found that Self-supervised Image-specific Prototype Exploration (SIPE) method [107] is ideal. It integrates and optimizes different features produced from various layers of the network in the classification training process, for obtaining enhanced CAM, as explained in Section 3.4.1. Therefore, we choose SIPE as baseline method.

ResNet-50 [66], which is detailed in Chapter 2, is adopted as the backbone network in the implement of SIPE. The architecture of the backbone network adopted from ResNet-50 is described in Table 5.1. SIPE is trained using the augmented train set of PASCAL VOC 2012 dataset (10582 images) with only image-level labels.

The outputs of each layers (i.e., the outputs of con1, conv2_x, ..., conv5_x) of ResNet-50 (i.e., the shallow features) are resized to the original image size. They are noted F_1, F_2, F_3, F_4, and F_5. The scaling factor goes from 2 to 32. These features are z-score normalized to prevent certain parts of feature from dominating the training process due to their larger magnitude.

For each image of the train set of PASCAL VOC 2012 (1464 images), we determine $\mathcal{T}_{GT}$ (the SVM training set) from the shallow feature $\boldsymbol{S'}$ which can be F_i, $i \in \{1, 2, 3, 4, 5\}$, or the original image itself (considered as a shallow feature, called F_0, in this case). The number of training samples has been fixed to $n_{GT} = 2500$ for each class, including the background. Considering the fact that the resolution of images in the PASCAL VOC 2012 train set is generally around 350×500, and there are averagely 1.4 foreground classes in each image [18], the number of pixels

Layer name	Feature name	Output size	Configurations
conv1	F_1	$224 \times 224 \times 64$	conv 7×7, 64
conv2_x	F_2	$112 \times 112 \times 256$	max pooling $\begin{bmatrix}\text{conv } 1 \times 1, 64\\ \text{conv } 3 \times 3, 64\\ \text{conv } 1 \times 1, 256\end{bmatrix} \times 3$
conv3_x	F_3	$56 \times 56 \times 512$	$\begin{bmatrix}\text{conv } 1 \times 1, 128\\ \text{conv } 3 \times 3, 128\\ \text{conv } 1 \times 1, 512\end{bmatrix} \times 4$
conv4_x	F_4	$28 \times 28 \times 1024$	$\begin{bmatrix}\text{conv } 1 \times 1, 256\\ \text{conv } 3 \times 3, 256\\ \text{conv } 1 \times 1, 1024\end{bmatrix} \times 6$
conv5_x	F_5	$14 \times 14 \times 2048$	$\begin{bmatrix}\text{conv } 1 \times 1, 512\\ \text{conv } 3 \times 3, 512\\ \text{conv } 1 \times 1, 2048\end{bmatrix} \times 3$

Table 5.1: Architecture of the backbone network adopted from ResNet-50. Building blocks are shown in brackets, with the numbers of blocks stacked. Down-sampling is performed by each layer with a stride of 2. Input image is 448×448.

selected as training samples corresponds to only a small fraction (less than 5%) of the image pixels. They are used to train the SVM. Finally, all image pixels are classified using their pixel-wise shallow feature vectors to get the SVM classification mask.

5.4.2 Experiment results

Figure 5.4 shows the SVM predictions when using the ground truth and the different features F_i, for $i \in \{0, 1, 2, 3, 4, 5\}$. When using F_0 (i.e., the original images) as the training data for the SVM, the output results have relatively detailed object contours but contains lots of false positive predictions in background regions, leading to a relatively imprecise prediction. On the other hand, the predicted segmentation boundaries are blurred and less accurate when using F_5 features as training data. Features F_0 have few semantic but have the best resolution, while resolution of F_5 is low, but F_5 embeds lots of context and semantic information. The SVM results show clearly these two extremes. Choosing intermediate shallow features leads to balanced results. As shown from column 4 to column 6 (correspond to using feature F_1 to F_3) in Figure 5.4, the compromise between resolution and semantics leads to improved predictions and objects boundaries are sharp. This an illustration of the well known bias-variance tradeoff.

Table 5.2 presents the IoU evaluated for each class, and the mIoU on the PASCAL VOC 2012 train set [18]. Numerical results confirm the visual analysis discussed above: F_1, F_2 and F_3 show better segmentation performance for all classes. Using

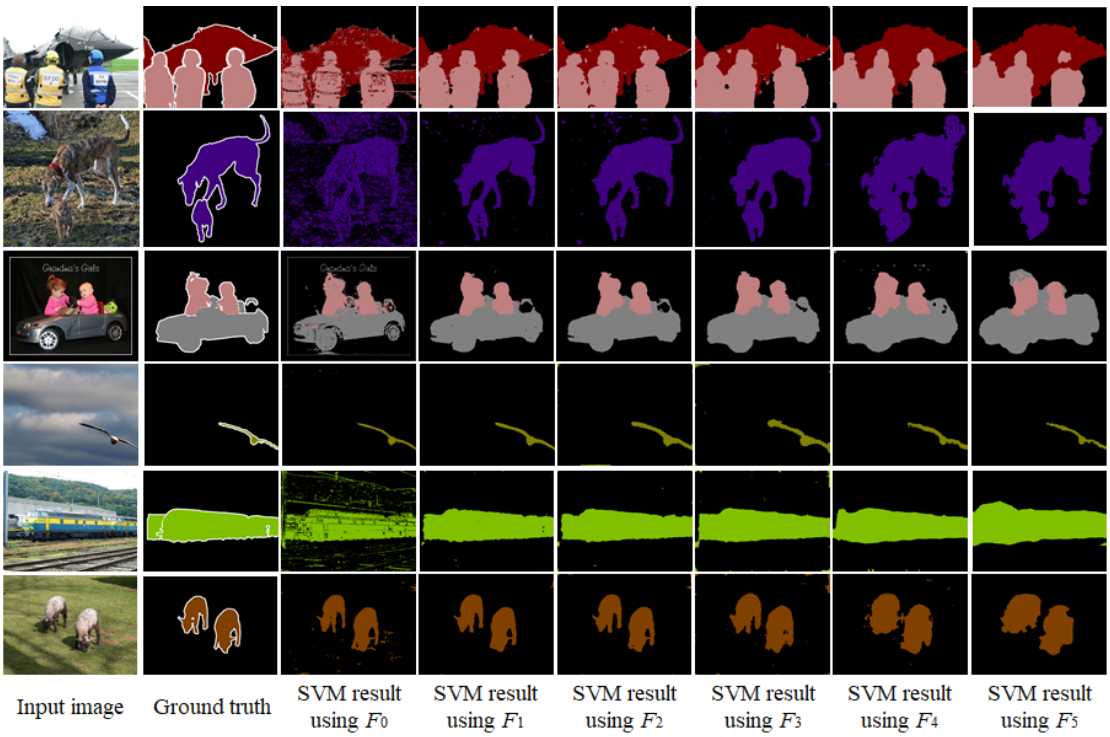


Figure 5.4: SVM classification masks on the PASCAL VOC 2012 train set.

Feature	bkg	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	motor	person	plant	sheep	sofa	train	tv	mIoU
img	86.98	68.79	35.91	64.40	57.83	58.97	68.77	62.19	77.31	58.65	73.76	69.01	71.25	73.22	57.99	68.02	61.07	70.89	74.65	64.80	60.96	65.97
F_1	**98.98**	**96.70**	**89.97**	**96.95**	**96.39**	95.61	**96.80**	**96.74**	**98.32**	**95.76**	**97.77**	97.74	**97.82**	**97.75**	**95.22**	**96.87**	**96.17**	**97.63**	97.82	**96.70**	96.13	**96.67**
F_2	98.65	92.87	85.61	95.38	93.90	95.34	95.66	95.63	98.16	94.89	96.77	96.76	97.23	96.86	93.93	95.85	95.02	97.22	97.47	95.80	95.19	95.44
F_3	98.58	87.62	84.83	93.02	91.31	**96.36**	96.74	95.63	97.91	95.57	95.73	**97.82**	96.44	96.31	93.88	96.27	95.31	96.53	**98.29**	96.52	**97.50**	95.15
F_4	96.98	78.48	75.53	86.10	84.00	93.96	94.35	91.75	95.39	91.04	87.35	96.74	90.63	88.29	89.19	93.85	91.03	88.22	96.43	93.71	96.69	90.46
F_5	90.19	55.38	66.41	61.51	60.54	77.25	80.48	75.81	82.78	80.74	72.03	91.36	72.52	71.02	73.94	85.84	81.31	72.71	82.62	73.93	86.00	75.92

Table 5.2: Comparison of segmentation mIoU (%) scores using different features on the PASCAL VOC 2012 train set. The highest values are highlighted by bold. bkg represents the background.

F_1 feature gives the best performance in most categories, which suggests that when ground truth is available, the pixel classifier trained using high-resolution with local semantics achieves the best compromise between resolution and semantics.

The expansion method, when using ground truth, gives an upper bound limit of 96.67% (mIoU), which is almost perfect. It shows the great performance potential of the proposed expansion method, if an efficient sampling process can be defined.

5.5 Experiments

In this section, we start by describing the experimental settings: datasets, evaluation metrics, baseline and implementation details. Then, taking into account the results in Section 5.4, we carry out experiments to determine the shallow feature to be selected and the number of repetitions of the self-enhanced SVM process to get the best results. We also conduct ablation studies to evaluate the improvement made by different enhancement processes. Finally, we compare our methods with recent similar WSSS methods that use only image-level labels.

5.5.1 Experimental settings

Datasets and evaluation metrics

All experiments are done using the original PASCAL VOC 2012 dataset [18] and its augmented version [147] (described in Section 5.4).

We use precision to evaluate the quality of the label selected during the sample selection step. The Intersection-over-Union per class and the mean Intersection-over-Union are used to evaluate the segmentation performance. The segmentation results on the PASCAL VOC 2012 test set are evaluated from the official evaluation server.

Baseline

In the experiments, as already stated in Section 5.4, SIPE [107] is considered as the baseline method. By empowering the feature representation, the enhanced CAM produced by SIPE yields accurate recognition for objects in images. Compared with the methods proposed in Chapter 4, using SIPE as the baseline enables

higher label accuracy in our designed training set, resulting in better experimental outcomes. Besides, since SIPE is used to provide shallow features in the study detailed in Section 5.4, its experimental results also offer valuable insights for our selection of shallow features. The proposed expansion method is then used after the generation of enhanced CAM from SIPE (see Figure 5.1).

Implementation Details

Following the default setting of SIPE, ResNet-50 [66] is adopted as the backbone network, as in Section 5.4, to get the CAM. The shallow feature $\boldsymbol{S'}$ is one of the F_i, $i \in \{1, 2, 3\}$, extracted from the layers of ResNet-50 (see Table 5.1). Indeed, the experiments in section 5.4 show that, F_0 has not enough semantic information, F_4 and F_5 are low-resolution, and F_1, F_2 and F_3 gives similar results, combining sufficient semantics with high-resolution. Therefore, we evaluate the expansion method with those three shallow features, to determine which one performs the best.

In the sample selection step, the size of $\mathcal{T}_1$ is $n = 2500$. We empirically set $T_{fg} = 0.2$ and $T_{bg} = 0.05$ to obtain the initial seed, and set the similarity threshold $T_s = 0.2$ to exclude foreground candidate samples which are too similar to background prototype, providing $\mathcal{T}_2$.

The refinement post-processing, to further improve the pseudo-masks (i.e., the output of the SVM classifier), is a tow steps sequence based on dCRF [116] and IRNet [29] (see Figure 5.1). First, the original image and the output of the SVM classifier are used as the inputs of the dCRF, leading to enhanced labels $\boldsymbol{M^{dCRF}}$. Then, the original image and $\boldsymbol{M^{dCRF}}$ are used as the inputs of the IRNet to get the output pseudo-masks.

The pairs formed with the images and the pseudo-masks (that can be either the SVM calssification masks or the output of the refinement process), are used to train the fully supervised segmentation model (see Figure 5.3). We use DeepLabV2 [14] with ResNet-101 [66] backbone as the fully supervised semantic segmentation models, which is trained by the pseudo-masks. In addition, for a fair comparison, we also train the segmentation model with ResNet-38 [70] backbone with the default setting as [119]. The segmentation models are trained using the pseudo-masks. As already noted, once trained, these models are evaluated on the PASCAL VOC test set and the evaluation results are given by the official server.

5.5.2 Choice of the shallow feature and the iteration number in the self-enhanced SVM

We evaluate the expansion method with F_1, F_2 and F_3 to determine which one performs the best, since using these three features all achieve more than 95% mIoU results in the study proposed in Section 5.4. Besides, in the expansion method, a self-enhanced training process is used. We then need to determine the appropriate number of iterations.

Table 5.3 shows the mIoU results on the SVM classification masks, obtained on the PASCAL VOC 2012 train set, according to the selected sallow feature and the number of iterations. They show that using F_2 with 3 repetitions works the best. This solution is therefore selected for the following experiments.

We can also note that, although using the F_1 feature gives the best results when ground truth masks are available, see Table 5.2, it does not perform so well when pixel labels are inferred. One possible explanation for the drop in performance is that the less semantic the attributes, the greater their variability, and the greater the impact of mislabeling samples during learning. Since F_1 attribute contains the least semantic, the samples to be classified are the most dispersed, and the decision function learned is less regular and more sensitive to labeling errors than the ones defined using deeper features (F_2 or F_3).

5.5.3 Ablation studies

We conduct 4 ablation studies to assess the impact of the proposed approach on the PASCAL VOC 2012 train and augmented train set.

Train set sample filtering

We evaluate the performance of the train set sample filtering process applied in the sample selection step. Table 5.4 provides the precision of each class: background and objects, with or without the application of this process on the PASCAL VOC 2012 train set.

The results in the first line in Table 5.4 show that the original precision of label prediction for background is much higher than the ones for the objects. Moreover, it is already very high (90%), which suggests that the background predictions in the initial seed are reliable and can be used with confidence to create a background prototype. Therefore, the process is only applied to samples which are labeled as objects. Those samples which are too similar to the background prototype are excluded. Thus, the process has no impact on background labels as can be seen in column 1 of Table 5.4. Furthermore, the efficiency of this filtering process is observed for all object classes, comparing precision with or without application of the complementary process. The mean precision for all objects classes gains 3.6% by applying the fitering process.

repetition	1	2	3	4	5	6	7
F_1	54.8	54.5	55.1	55.8	**56.3**	55.6	54.3
F_2	59.6	60.5	**61.2**	60.0	59.3	59.1	58.9
F_3	58.2	**59.2**	59.1	58.7	57.5	58.3	58.3

Table 5.3: mIoU (%) of the SVM classification mask using F_1, F_2 or F_3 as shallow feature and different number of iterations in the self-enhanced SVM process. The best results for each feature are highlighted in bold.

Process	bkg	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	motor	person	plant	sheep	sofa	train	tv	mean
w/o SFP	90.3	43.5	25.2	54.1	48.7	39.8	81.5	57.0	78.4	33.4	71.4	65.4	69.0	64.7	64.4	50.7	43.0	67.4	69.0	59.6	55.5	55.9
with SFP	90.3	48.9	29.0	59.6	52.5	42.7	82.2	59.5	82.3	35.1	75.2	65.6	72.9	70.5	67.3	56.3	44.2	72.6	71.3	61.0	57.4	59.5
Gain	-	5.4	3.8	5.6	4.0	2.9	0.6	2.4	3.9	1.7	3.8	0.2	3.9	5.7	2.9	5.6	1.2	5.1	2.3	1.4	1.9	3.6

Table 5.4: Precision (%) of sample selection for categories without and with train set Sample Filtering Process (SFP) on the PASCAL VOC 2012 train set. bkg presents the background. mean presents the mean precision for all objects classes.

Expansion method

The effect of each process in the proposed MIP expansion method is now evaluated. To do so, we compute the mIoU on the pseudo-masks, obtained when the refinement step is removed. The baseline corresponds the SIPE method without the refinement step.

According to the first and second lines in Table 5.5, introducing the SVM based expansion method, using only the seed sample selection and without the self-enhanced process, leads to slight improvements compare to the baseline. As mentioned before, this can be explained by the incorrect label rate in the baseline seed.

As already shown, the addition of the train set sample filtering process improves the precision of selected samples. This has an impact on the SVM result (third line in Table 5.5), which shows an enhancement of the mIoU by 1.7% and 0.3% on the train and augmented train sets, respectively. The improved quantitative results of SVM by using train set sample filtering process can be observed in Figure 5.5. Without using filtering process, the results of SVM are close to the seed from baseline. When samples with incorrect labels in the training set are excluded through the filtering process, the results from the SVM show enhanced detail and clearer boundaries for the objects.

These results also show that the SVM provides better classification results. That is why, we introduced the self-enhanced training strategy. Thanks to this process, the SVM train label's quality is globally improved as the sequences of self-enhanced training strategy goes on. Incorrect labels of samples in the learning set is reduced and, as a result, misclassifications resulting from imperfect training are also re-

baseline	SSS	SFP	SeT	train	augmented train
✓				58.6	56.8
✓	✓			59.0	57.1
✓	✓	✓		60.7	57.4
✓	✓	✓	✓	**61.2**	**58.9**

Table 5.5: mIoU (%) considering ablation in our MIP expansion method, using the PASCAL VOC 2012 train and augmented train sets. SSS = *Seed Sample Selection.* SFP = *train set Sample Filtering Process.* SeT = *Self-enhanced Training. The best results are highlighted in bold.*

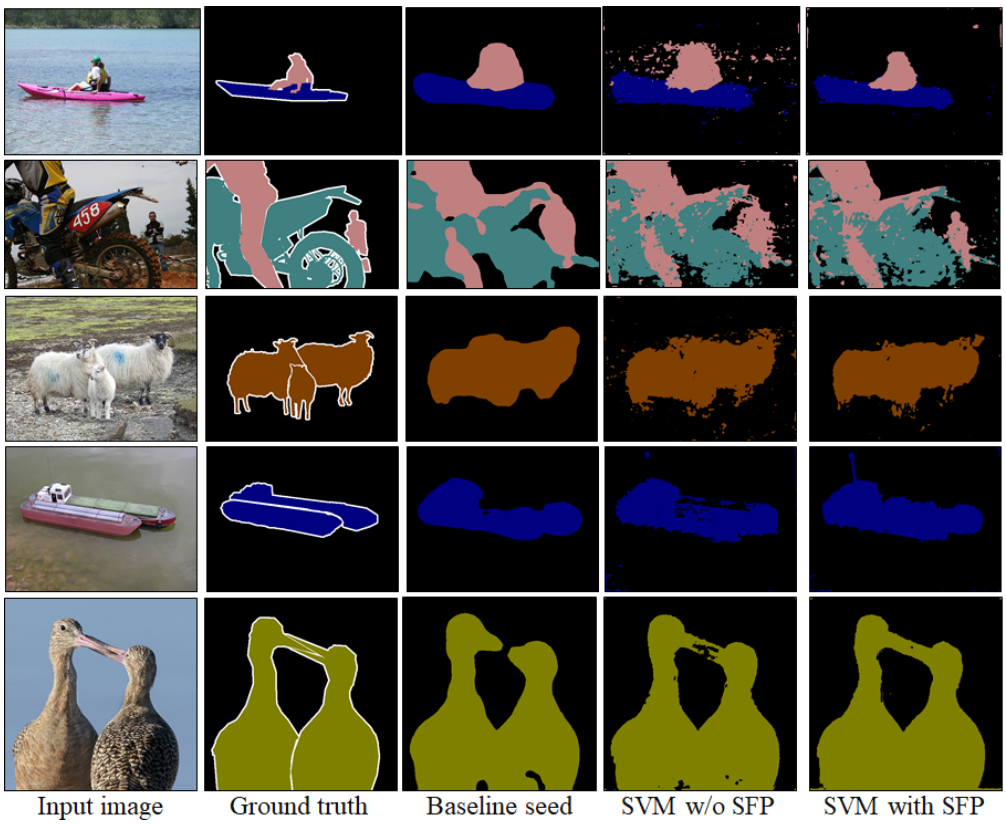


Figure 5.5: Qualitative results of ablation study without and with train set sample filtering process on the PASCAL VOC 2012 train set.

duced. The final mIoU (see fourth line in Table 5.5) reach 61.2% and 58.9% in the train and augmented train sets, respectively. It is a significant improvement in comparison to the baseline seed (2.6% on the train set).

Refinement process

Finally, we evaluate the mIoU of pseudo-masks improved by different conventional post processes [29, 116]. A fair comparison between our method, the baseline method [107], and other WSSS methods [36, 37, 40, 114, 125] applying the same post processes is shown in the Table 5.6.

It is observed that, our results after dCRF post-process exceed the other state-of-the-art methods. After applying IRNet, the difference reduces with some methods. The performance gains achieved by IRNet vary very significantly depending on the method (between 3.4% and 12.6%), which suggests that the characteristics of the masks also vary very significantly depending on the method, which does not really seem obvious at first analysis. Comparatively, our relative performance improvement falls within the average gain (6.2%). While it may not lead to the best results, it is really close (in the range of statistical fluctuation) and surpasses most of other approaches.

Figure 5.6 shows the qualitative results from the baseline method and our method with dCRF process [116] for images in the train set of PASCAL VOC 2012 dataset. The dCRF refinement process is able to remove noise produced by isolated pixels,

Method	Backbone	dCRF	Mask
AdvCAM [114]	ResNet-50	62.1	69.9
RIB [40]	ResNet-50	62.9	**70.6**
MuSCLe [37]	EfficientNet	64.6	66.8
SIPE [107]	ResNet-50	64.7	69.3
ICD [36]	VGG16	62.2	-
Iter_dCRF [125]	ResNet-50	62.7	-
MIP	ResNet-50	**66.2**	70.3

Table 5.6: mIoU (%) of pseudo-masks with different refinement process on the PASCAL VOC 2012 train set. The best results are shown in bold. Mask means the refined pseudo-masks by both dCRF and IRNet post processes.

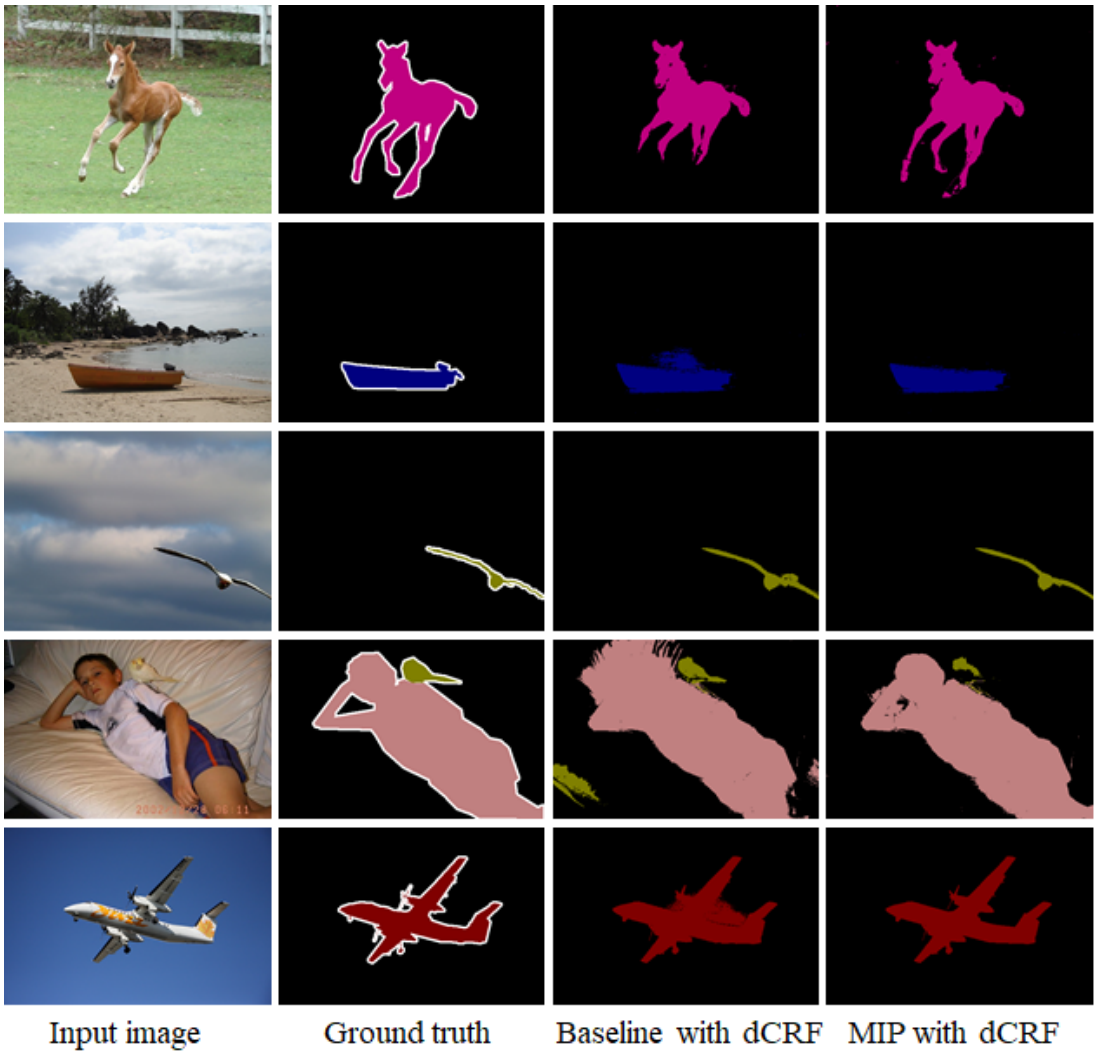


Figure 5.6: Qualitative results of baseline and the proposed expansion method with dCRF process on images from the PASCAL VOC 2012 train set.

since dCRF takes the surrounded pixels' prediction into consideration. It is observed that, under the same conditions of processing with dCRF, our results are more accurate compared to the baseline method. As illustrated in first column, compared with the baseline method, the recognition of the horse object is more complete in the proposed method. Additionally, regions previously mislabeled as background in the baseline method can now be correctly recognized as objects, as seen in the rest examples.

Figure 5.7 presents the qualitative results from the baseline method and our method using the IRNet process [29] for images in the train set of the PASCAL VOC 2012 dataset. The results show that the accuracy of pseudo-masks processed with IRNet has significantly improved. They not only recognize most regions of the objects but also possess clear objects boundaries. Compared to the baseline, since the proposed method uses high-resolution shallow features, it recovers more target details as shown in the second and fifth lines.

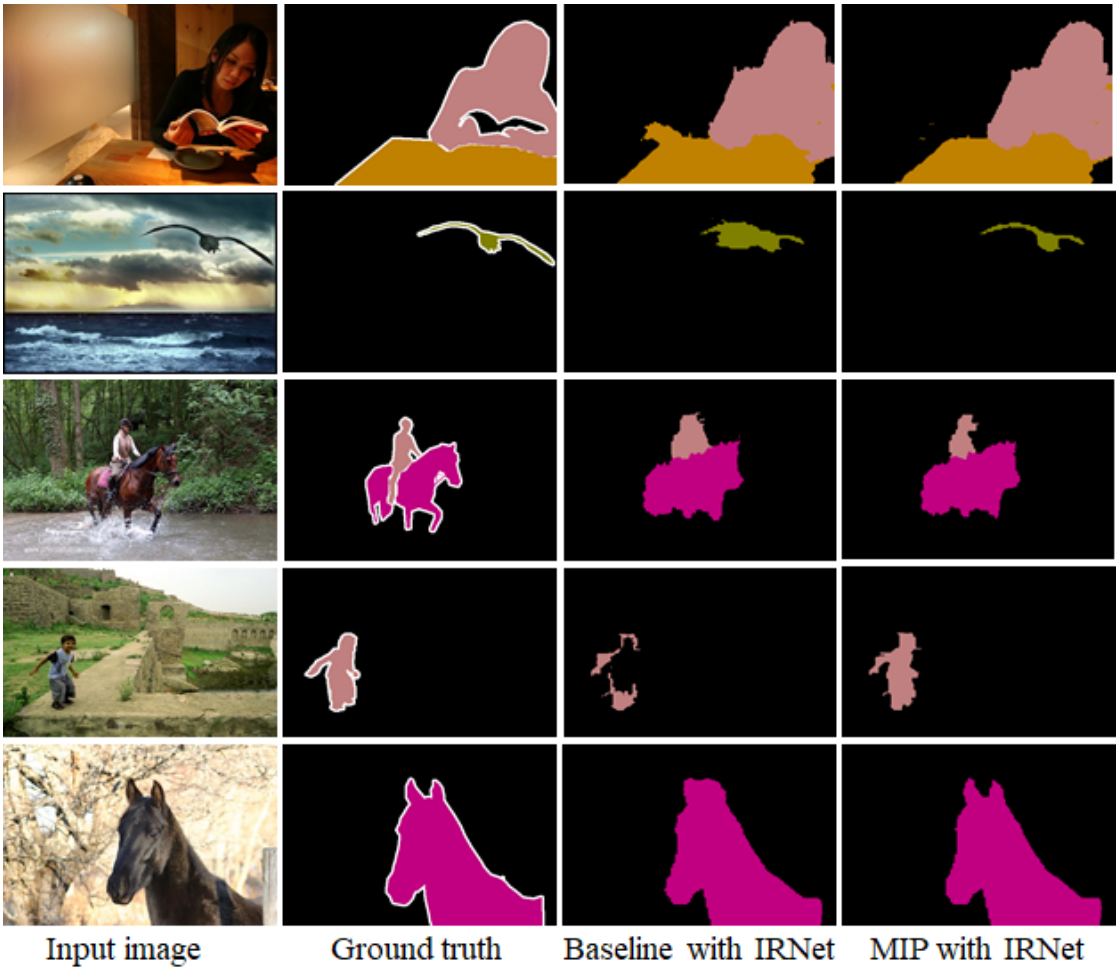


Figure 5.7: Qualitative results of baseline and the proposed expansion method with IRNet process on images from the PASCAL VOC 2012 train set.

5.5.4 Comparison with state-of-the-arts

The final comparison is made by the segmentation performance of the semantic segmentation network trained by the pseudo-masks. Table 5.7 and Table 5.8 show the quantitative comparison on the PASCAL VOC 2012 validation and test set between MIP, with the SIPE baseline, and state-of-the-art methods based on segmentation models with 2 different backbone networks ResNet-38 and ResNet-101. Details of these two networks are presented in Chapter 2.

Compared with the baseline method [107], MIP shows significant improvements in both validation and test set. More broadly, the MIP based method achieved the best performance on the validation set, no matter the backbone. While we did not achieve the best results on the test set, with ResNet-38 the performance is in the range of statistical fluctuation of the best one. With ResNet-101, we are among the best in the validation set. The relatively close experimental results between the validation and test sets indicate the consistency of our segmentation performance on different images. Indeed, we should expect close results between the validation and the test sets, since both are not used during training.

Method	Val	Test
SIPE [107]	68.2	69.5
ADELE [150]	68.6	68.9
TransCAM [151]	69.3	69.6
ASDT [152]	**69.7**	70.1
URN [115]	69.4	**70.6**
MIP	**69.7**	70.4[1]

Table 5.7: mIoU (%) of segmentation results on the PASCAL VOC 2012 validation and test set. All methods in table take ResNet-38 as backbone network and only use image-level label as supervision.

It is observed that, both methods in SPML (Semi-supervised Pixel-wise Metric Learning) [153] and OPC (Online Pseudo-masks Correcting) [154] methods achieved over 71% mIoU on the test set. Specifically, SPML formulates weakly supervised segmentation as a semi-supervised metric learning problem. It uses a contrastive learning approach to map each pixel into a point in the feature space, where pixels of the same (different) classes are mapped to the similar (distinctive) features. Contrastive learning has been popularly applied in recent WSSS methods [37, 155, 156, 157], since it effectively enhances the distinction between pixel-wise embeddings of different classes for improving segmentation accuracy. Unlike the methods discussed in this thesis on improving the accuracy of pseudo-masks, OPC focuses more on the means to appropriately train the segmentation model given existing pseudo-masks. OPC introduces an Exponential Moving Average (EMA) model [158] as the teacher model of the segmentation model (i.e., the student model) trained by the pseudo-masks. The output of the teacher model can be

[1] `http://host.robots.ox.ac.uk:8080/anonymous/GWLPKC.html`

Method	Val	Test
Iter_dCRF [125]	63.5	63.2
AdvCAM [114]	68.1	68.0
ReCAM [159]	68.5	68.4
ASDT [152]	68.5	68.4
PPL [160]	67.8	69.6
SIPE [107]	68.8	69.7
AMR [161]	68.8	69.1
VWL-L [145]	69.2	69.2
AMN [122]	69.5	69.6
W-OoD [162]	69.8	69.9
ESOL [121]	69.9	69.3
SPML [153]	69.5	71.6
OPC [154]	70.5	**71.8**
MIP	**70.6**	70.8[2]

Table 5.8: mIoU (%) of segmentation results on the PASCAL VOC 2012 validation and test set. All methods in table take ResNet-101 as backbone network and only use image-level label as supervision.

used to correct pseudo-masks online, which finally improve the segmentation performance of the student model. This suggests that, considering MIP has already achieved relatively accurate pseudo-masks, using a training strategy for the segmentation model that is more suitable to WSSS can obtain better segmentation results.

As mentioned before, some recent WSSS methods with image-level label also use saliency [113, 131, 156, 163], superpixel [36], other extra supervision [164], transformer [145, 165], CNN with increased depth [26, 27] and advanced version of segmentation network [166] to improve their final segmentation results. In order to make a fair comparison, results shown in the Table 5.7 and Table 5.8 are methods that only use image-level labels and the same segmentation model with the same backbone network. However, the best published results in conferences or journals reach for example 73.6% for the method PPC (Pixel-to-Prototype Contrast) [156] that uses saliency, or 73.5% for the method BECO (Boundary-Enhanced CO-training) [165] based on transformer, on the test set. As discussed in Chapter 3 and Chapter 4, saliency can provide useful background cues, but the quality of saliency itself also determines the final results. Additionally, using saliency introduces extra supervision and increases computational costs. Recently, transformers [167, 89, 168] have become increasingly popular in the image processing domain, and more transformer-based methods [151, 169, 170, 171, 172, 173] are emerging in WSSS. This represents a new research direction worth attention.

Figure 5.8 shows qualitative segmentation results from the validation set. As shown in the provided examples in Figure 5.8.(a), our method demonstrates ex-

[2] `http://host.robots.ox.ac.uk:8080/anonymous/F9GXN1.html`

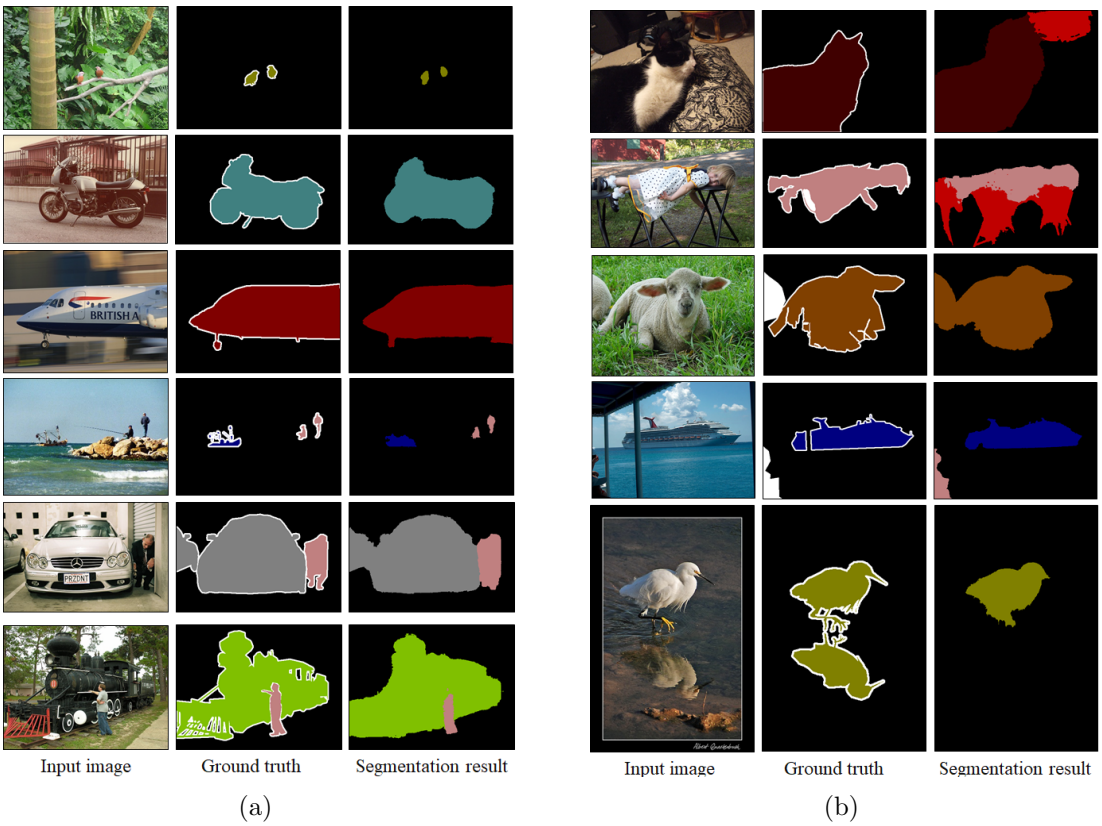


Figure 5.8: Qualitative segmentation results on the PASCAL VOC 2012 validation set (a) with different sizes of objects and (b) when the ground truth is not total accurate. The white regions labeled in ground truth are used to label the boundaries and to mask out ambiguous, heavily occluded or too small objects.

cellent detail recovery capabilities for objects of various sizes while maintaining clear boundaries. Generally, our approach performs well in images with relatively uniform background features. Even in the presence of complex backgrounds, our method produces robust segmentation results. Besides, as shown in Figure 5.8.(b), the ground truth for the training images are not entirely correct (the shadow of bird in the last example is wrongly labeled as the bird) or completely annotated (the first two examples miss the annotations of chair, and the white regions mean that the annotator can not give confident labels in such regions) in the dataset. It is observed that experiments results from our method can correctly segment these challenging regions. These results suggest that our approach is capable of effectively exploiting shallow features to recover object details.

5.6 Conclusion

One key issue for weakly supervised semantic segmentation methods, giving only image-level labels, is to generate accurate pseudo-masks. To tackle this problem, we propose a new expansion process based on image-specific pixel classifier using multi-feature (CAM, seed and shallow feature) to obtain accurate pseudo-masks

with a good balance between resolution and semantics.

For each image, CAM, seed and shallow feature are used in a sample selection process, composed of seed sample selection and train set sample filtering processes. The samples (couples of data and labels) serve as train set for a SVM classifier. To improve the overall performance, a self-enhanced process is also proposed that consists in sequentially updating the labels in the train set according to the SVM prediction (trained on the train set with the previous labels).

The critical part of the proposed MIP appears to be the sample selection. We found in the study that when the ground truth is known, that is, the source of sample selection is perfect, the mIoU of SVM output can achieve more than 96%. It reveals that the proposed method can achieve high performance if an efficient sample selection method can be designed.

Compared with the baseline and recent WSSS methods, our approach demonstrates a significant improvement in the improved pseudo-masks, which recovers more target details and provides clearer boundaries for the objects. Our experiments illustrate the value of acquiring reliable data from imperfect CAM, and prove that shallow feature with high resolution is effective in the segmentation. The results also highlight the advantages of the image-specific pixel classifier for the WSSS task, since it can effectively use the visual features of each image to improve the accuracy of predicted labels. By concentrating on the information within a single image, this approach avoids the propagation and interference of incorrect annotations from other images, which is a common issue found in traditional global training methods.

Through comparative analysis with state-of-the-art WSSS methods, we discover that using advanced versions of segmentation networks or transformer-based approaches can improve the final segmentation results. Furthermore, considering the fact that it is difficult for pseudo-masks to achieve the same accuracy as ground truth, exploring appropriate training strategies to train the segmentation model by the improved pseudo-masks is also a worthwhile research direction.

Chapter 6

Conclusions and prospects

Contents

6.1 Conclusions 107

6.2 Prospects 109

6.1 Conclusions

In this thesis, we have studied methods to improve the pseudo-masks for weakly supervised semantic segmentation. We have introduced the neural networks and convolutional neural networks, with a specific emphasis on their roles in image processing, in Chapter 2. Typical CNN-based models for image classification and segmentation have been presented, since they are commonly used in recent weakly supervised semantic segmentation methods. Related works have been analyzed in Chapter 3. The motivations of semantic segmentation tasks under different levels of supervision (i.e., fully, un-, semi- and weakly supervised semantic segmentation) have been presented. In weakly supervised semantic segmentation, the primary challenge is to accurately infer location information from weak supervision, such as image-level labels, labeled points, bounding boxes and scribbles, all of which only provide limited spatial details of objects in the images compared to pixel-level labels. Among all weak supervision, the image-level labels are the easiest one to annotate but also provide the least location information. In the thesis, we have focused on this framework. An analysis of the training pipeline of WSSS with image level labels reveals firstly that the accuracy of pseudo-masks, which are served as the training targets to train a segmentation model, has strong impact on the final segmentation performance, and secondly that the generation of pseudo-masks can be divided into 2 steps: obtaining Class Activation Map (CAM) and inferring pseudo-masks from the CAM. To improve these two steps, new methods have been proposed. The conclusions for the proposed methods are summarized as follows.

New classification network training methods to enhance CAM

In Chapter 4, we observe that the limitations of CAM for semantic segmentation can be traced back to the fact that the primary objective of the classification network, which provides features to generate the CAM, is to classify the target objects in the images, without considering the properties of the CAM itself. To enforce selected properties of the CAM such as exclusion of overlapping regions (where multiple significant activation values appear to different classes) or full coverage of objects, two methods are introduced by adding new objectives to the training of the classification model while enhancing desired CAM properties.

The first method proposes a new loss function to penalize overlapping regions by orthogonalizing the vectors reshaped from the activation maps for different classes. Experimental results show that the overlapping regions are significantly reduced, and the enhanced CAM focuses more on the confident regions of the target objects compared with the baseline method. The recognition capability of the trained classification model is shown to improve for certain classes. However, it is also observed that the activated areas in the obtained CAM are globally reduced, that avoiding the appearance of overlapping regions, which leads to smaller object regions in the CAM. This can be explained by the fact that a natural solution to reduce the overlapping regions between classes is to shrink the activated area as much as possible. These activated areas cover a small proportion of the objects which is not fully aligned with objective to segment objects.

In response to previous results, the second method aims at extending the object regions with significant response in the CAM. This means to increase the activation values in regions with less specific object related information. To do so, two additional branches with proposed objectives are introduced in the training pipeline of the classification network. The first branch outputs a response score averaged from the class-wise response map produced by applying an adaptive tanh activation function on the class-wise feature map. The second branch gives the weighted class score by using a new weighted pooling method to give less weight to large values in the class-wise map. Both scores are used to compute a new proposed loss functions that reduces contributions of the local large values of the class-wise feature map to the training. Experimental results show that, these modifications lead to larger activated regions, but also to over-activation issue, with many background areas being incorrectly activated in the CAM. The primary cause of this over-activation is that, areas with low activation include not only the less-discriminative regions within the objects, but also the background regions. Thus, lacking sufficiently accurate guidance, the model inevitably increases the response in the background regions.

The study of these methods has inspired us to acquire reliable information from the imperfect CAM. We also found that, the deep feature generated from the last convolutional layer of the backbone classification network can only provide limited low-resolution information when producing the CAM results, which leads us to explore the use of high-resolution features to recover the details and accurate boundaries of targets. Additionally, it also inspired us with the importance of analyzing the specificity of each image, thereby leveraging image-specific features to enhance the accuracy of label inference.

New expansion method based on a multi-features image-specific pixel classification approach

In Chapter 5, we focus on post-processing CAM to generate improved pseudo-masks. We analyze two key limitations of related methods: (1) Most methods rely on either low-resolution deep features with rich semantics or the original high-resolution image with limited semantics. (2) The specific features of each image are not fully explored when producing pseudo-masks. To address these issues, we propose a new expansion approach to generate improved pseudo-masks based on an image-specific pixel classifier using multi-feature to balance resolution and semantics.

For each image, CAM, seed, and shallow features are used in a sample selection process. The selected samples (pairs of pixel-wise feature and labels) are used to train an SVM classifier. A self-enhanced process is also proposed to reduce the impact of the mislabeled samples. It consists in sequentially updating the labels in the train set according to the SVM prediction (trained on the train set with the previous labels).

A study is conducted to select shallow features and test method upper bounds, which demonstrates the significant performance potential of this approach. In the study, when the ground truth is known, that is, the source of sample selection is perfect, the mIoU of SVM output can achieve more than 96%. The results reveal that, the proposed method can achieve high performance if an efficient sample selection method can be designed. It also proves the effectiveness of using high-resolution features with sufficient semantics in image segmentation.

When using only image-level labels, the output of the proposed expansion method increases 2.6% mIoU compared with the results from the baseline method in the same dataset. The segmentation networks trained by the improved pseudo-masks also outperform most of recent WSSS methods in both validation and test set. The experimental results show that our method demonstrates excellent detail recovery capabilities for objects of various sizes, while maintaining clear boundaries. This approach highlight the importance of obtaining reliable data from imperfect CAM, and prove that high-resolution shallow features effectively improve segmentation details. The image-specific pixel-wise classifier targets the distinctive features within each image, demonstrating its effectiveness in inferring pixel-level labels for a given image. As a result, the obtained improved pseudo-masks can be effectively used to train a segmentation model.

6.2 Prospects

With the conclusion of the thesis, interesting prospects can be proposed for further work:

Any CNN-based models can be adapted as the backbone network for the methods proposed in this thesis. Future work could involve evaluating the performance of these methods on various networks, such as the high-resolution network designed

to enhance detail analysis [174]. It can provide a further understanding of the strengths and challenges of the proposed methods.

The research in Chapter 4 reveals that the over-activation of CAM in the background regions often results in a high number of false object predictions. Developing methods (without using auxiliary data) to correct these misclassified label in background remains an important topic to explore for related approaches in WSSS. Taking inspiration from the sample filtering process used in Chapter 5, one possible solution could be to introduce a background prototype obtained by calculating the mean of pixel-wise semantic features in the background regions and label those pixels, that share a strong similarity with the prototype as background.

A possible extension of the proposed approach in Chapter 5 would be to exploit several shallow features, and not just one, during the sampling phase, in order to improve it. In Chapter 5, we separately use features generated from different layers of the backbone network for the sample's data in the training set. Indeed, we also did experiments by simply concatenating selected features, i.e., concatenating more than one features along the channel axis. However, the output results from the classifier trained by the concatenated features are close to the results from the classifier trained by the feature which has the most number of channels among the selected features. The effective methods to merge several features beyond the simple use of concatenation, especially taking into account the differences in feature depths, remains an open question.

Considering the fact that the accuracy of pseudo-masks inevitably has a gap compared with the ground truth, designing a robust classifier that can cope with a proportion of incorrect label is a worth exploring research direction and an open challenge.

Furthermore, another interesting research direction is combining WSSS with popular Transformer-based foundation models. Some studies have shown that using Transformer-based models instead of CNN-based models to generate CAM using only image-level labels can lead to accurate results [37, 151]. Since these models are trained on large amounts of data, they commonly have strong generalization capabilities. An important area of study is how to fine-tune these pre-trained models specifically for WSSS tasks on particular datasets.

Appendix A

Résumé en français

A.1 Introduction générale

La segmentation sémantique est une tâche essentielle en vision par ordinateur. Elle vise à attribuer à chaque pixel d'une image une classe spécifique [1, 2]. Elle est largement appliquée en imagerie médicale [3, 4], en conduite autonome [5, 6, 7] ou encore en télédétection [8, 9]. Ces dernières années, les approches basées sur l'apprentissage profond, en particulier les réseaux de neurones convolutionnels (CNN pour Convolutional Neural Networks) [10, 11, 12, 13, 14, 15, 16], ont démontré de très bonnes performances en segmentation grâce à leur capacité d'apprentissage de représentations complexes.

Cependant, une des principales faiblesses des méthodes de segmentation supervisées, comme celles basées sur les CNN, est qu'elles nécessitent une grande quantité de données entièrement annotées, c'est-à-dire des images avec des étiquettes pour chaque pixel (vérité terrain). Ce type d'annotation est difficile à obtenir car elle demande une très grande précision et nécessite donc beaucoup de temps. Dans certains domaines spécifiques, comme l'imagerie médicale, le manque de données annotées est courant. De plus, l'annotation manuelle au niveau des pixels nécessite des connaissances d'expert, ce qui augmente le coût de l'annotation.

L'entraînement de modèles de segmentation à l'aide de données annotées au niveau des pixels est connu sous le nom de segmentation sémantique supervisée (FSSS pour Fully Supervised Semantic Segmentation). Parmi les approches alternatives (voir figure A.1), la segmentation sémantique non supervisée (USS pour Unsupervised Semantic Segmentation) vise à classer les pixels des images sans utiliser de vérité terrain. Cependant, les performances de ce type d'approche sont notablement inférieures aux méthodes FSSS en raison de la difficulté des modèles de trouver des similarités dans les images sans être guidé par des données étiquetées. Une façon d'améliorer les résultats tout en allégeant la phase d'annotation est de n'étiqueter qu'une partie de la base d'apprentissage. Ce type d'approche, appelée segmentation sémantique semi-supervisée (SSSS pour Semi-Supervised Semantic Segmentation) exploite conjointement des données annotées et des données non annotées. Généralement la proportion de données annotées par rapport aux données

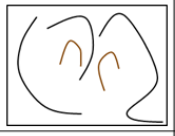
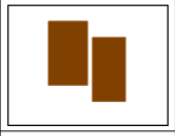
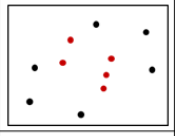
Data	Full supervision	Weak supervision			
Image	Pixel-level label	Scribbles	Bounding boxes	Labeled points	Image-level label
					
Annotation time per image (seconds):	240.0	34.9 (per scribble)	33.8 (per box)	22.1 (per point)	20.0

Figure A.1: Illustration des types d'annotations en segmentation faiblement supervisée, en comparaison avec la segmentation supervisée.

non annotées va jouer sur les performances des modèles de segmentation. Une autre approche permettant de réduire fortement le coût des annotations est d'associer à chaque image des étiquettes plus haut niveau. Ces approches, appelées segmentation sémantique faiblement supervisée (WSSS pour Weakly Supervised Semantic Segmentation), peuvent être décomposées en plusieurs groupes en fonction du type d'annotation. Il est possible d'utiliser des étiquettes au niveau de l'image [20, 21, 22, 23, 24, 25, 26, 27, 28, 29] qui indiquent les objets (ou classes) présents dans l'image, des points [30, 31] qui indiquent la classes de certains pixels ou zones dans l'image, des traits qui surlignent la présence des classes dans une zone de l'image [32, 33] ou encore des boîtes englobantes [34, 35], qui entourent les zones dans lesquelles se trouve une classe. C'est l'annotation au niveau de l'image qui est bien sûr la plus simple et la plus rapide à réaliser, mais elle ne fournit aucune information de localisation des classes dans l'image. C'est cette dernière que nous considérons dans ce travail.

Comme illustré par la Figure A.2, les méthodes récentes pour la WSSS, avec des étiquettes au niveau de l'image, suivent généralement un processus d'entraînement en deux étapes. La première étape consiste à générer des pseudo masques. Ceux-ci jouent un rôle crucial puisqu'ils servent à inférer des étiquettes pour chaque pixel de l'image, étiquettes qui servent de vérité terrain pour entrainer le modèle de segmentation lors de la deuxième étape. De fait, la qualité de ces pseudo masques influe directement les performances de la segmentation. Bien que la WSSS soit très populaire et qu'elle fasse l'objet de nombreux travaux, la qualité des pseudo masques présente encore un écart significatif par rapport à la vérité terrain. Par conséquent, l'objectif principal de cette thèse est d'explorer des approches permettant d'améliorer la précision des pseudo masques, ce qui est également le principal axe de recherche de la littérature récente en WSSS.

La génération de pseudo masques comprend elle aussi deux étapes. La première étape consiste à obtenir les cartes d'activation par classe (CAM pour Class Activation Map), qui sont données par une matrice tridimensionnelle, qu'on appellera CAM par la suite, produite à partir de la couche convolutionnelle finale d'un modèle CNN de classification. Ce dernier a été entraîné à partir d'une base de données constituée de couples image et étiquettes au niveau image (objets présents dans l'image). Chaque carte met en évidence les régions de l'image associées par le réseau à une classe donnée. A noter que ces cartes sont de taille inférieure à

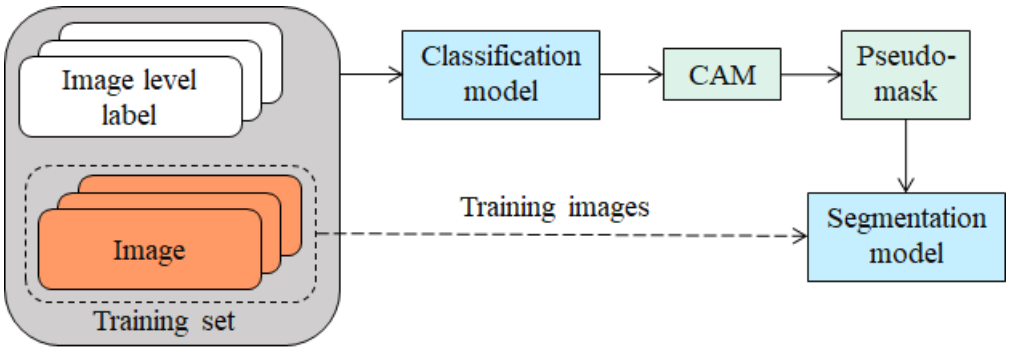


Figure A.2: Illustration du pipeline d'entraînement en deux étapes pour la segmentation sémantique faiblement supervisée avec des étiquettes au niveau de l'image.

la taille des images d'entrée, en fonction de l'architecture du modèle de classification (CNN), et que le nombre de cartes (la profondeur ou le nombre de canaux du CAM) est égal au nombre de classes dans la base de données. La deuxième étape consiste à générer des pseudo masques à partir des CAM. Dans sa version la plus basique, cela consiste à redimensionner le CAM à la taille spatiale de l'image d'entrée, puis à seuiller le CAM redimensionné avec un seuil fixé par l'utilisateur (i.e. seuiller chaque carte) et enfin à générer le pseudo masque en attribuant à chaque pixel du pseudo masque la classe correspondant à la valeur maximale des cartes pour ce pixel dans le CAM. Pour un pixel donné du pseudo masque, si toutes les valeurs d'activation correspondantes dans le CAM sont nulles alors on attribue la classe 'fond' au pixel.

L'objectif de ce travail est d'améliorer la segmentation finale d'images dans un contexte WSSS en considérant deux questions :

1. Comment obtenir un CAM de meilleure qualité qui permettra de générer un meilleur pseudo masque, peu importe la méthode de génération des pseudo masques ?
2. Comment produire des pseudo masques de meilleure qualité à partir des CAM existants ?

En ce qui concerne de première question, les études récentes [21, 36, 37, 38, 39, 40] montrent que les problèmes du CAM sont liés au fait qu'il se base sur un réseau de neurone utilisé pour une tâche de classification au niveau de l'image et non de segmentation.

La première conséquence de cet apprentissage par classe est que le réseau tend à se concentrer uniquement sur les informations les plus discriminatives pour la classification, ce qui est insuffisant pour fournir des localisations précises d'objets pour la tâche de segmentation. Afin d'essayer de pallier ce problème nous proposons deux méthodes dans la suite de ce travail. La première consiste à ajouter une contrainte lors de l'entrainement du modèle de classification afin d'éviter les régions de chevauchement entre différente cartes du CAM. L'idée est que si une zone spatiale produit une réponse significative sur une carte du CAM (i.e. un canal du CAM ou encore une classe) elle ne devrait avoir de valeurs significatives

dans aucune autre carte du CAM; par hypothèse un pixel ne peut appartenir qu'à une classe.

La seconde conséquence de l'apprentissage par classe est que, pour produire une réponse significative pour cette classe, il suffit que la carte du CAM associé à cette classe prennent des valeurs significatives sur une ou plusieurs parties de l'objet apprises comme spécifique de cette classe d'objet par rapport aux autres classes de la base. La réponse du réseau n'a pas besoin de couvrir la totalité de la région occupée par l'objet en question conduisant à la génération d'un pseudo masque qui ne rend pas compte de l'ensemble de l'objet à segmenter. Cette couverture partielle va dégrader la génération du pseudo masque qui ne rend pas compte de l'ensemble de l'objet et, par conséquent, dégrader l'apprentissage du modèle de segmentation. La deuxième approche développée cherche à pallier ce problème en pondérant les valeurs élevées du CAM afin de forcer le réseau à prendre en compte d'autres régions de l'objet pour le discriminer des autres classes. Cette pondération est appliquée pendant la phase d'apprentissage du CNN.

Pour la deuxième question, le but est d'améliorer la qualité des pseudo masques générés à partir des CAM produits par le réseau. Cette étape de traitement des CAM consiste généralement à améliorer la qualité des frontières des objets ou à pallier le fait que le CAM ne couvre pas l'intégralité de l'objet (comme expliqué précédemment). Nous rappelons que les pseudo masques sont ensuite utilisés comme vérité terrain pour entrainer le modèle de segmentation. Obtenir un modèle de segmentation de bonne qualité nécessite donc d'obtenir des pseudo masques de très bonne qualité. Il existe dans la littérature un certain nombre de traitements développés pour améliorer les pseudo masques. Certaines consistent, pour un objet donné, à étendre l'information de localisation fiable fournie par le CAM à d'autres régions invisibles pour le CAM et à inférer les étiquettes des pixels correspondants [25, 41, 42]. L'approche que nous proposons s'inscrit dans cette voie. L'idée est d'utiliser un classificateur auxiliaire pour chaque image de la base d'entrainement afin de classifier les pixels de l'image pour améliorer les pseudo masques. Cette approche, exploite plusieurs éléments d'information provenant notamment du CAM et des couches internes du CNN. L'objectif est d'améliorer la résolution des pseudo masques en exploitant des informations internes au réseau dont la résolution est meilleure que celle du CAM et de définir une description de chaque classe spécifique à chaque image et donc plus précise et plus à même de capter les régions se distingant du fond mais invisibles pour le CAM.

Le manuscrit en anglais se compose d'une introduction, correspondant au chapitre 1. Le chapitre 2 introduit les éléments théoriques utiles à ce travail. Le chapitre 3 propose un état de l'art sur la segmentation sémantique et plus particulièrement WSSS. Le chapitre 4 introduit les méthodes permettant d'améliorer le CAM et le chapitre 5 introduit la méthode permettant d'améliorer les pseudo masques. Cette dernière correspond à la principale contribution de ce travail. Le chapitre 6 conclu le travail et propose des perspectives. La suite de ce résumé étendu en français synthétise le contenu des chapitres 4 et 5, qui présentent les contributions de ce travail. Une conclusion et des perspectives sont finalement proposées au chapitre 6.



Figure A.3: Exemple d'images contenant deux classes (première ligne). La deuxième ligne correspond à la vérité terrain. Les troisième et quatrième lignes montrent les cartes du CAM pour chacune des classes présentent dans l'image, illustrant le fait que les objets ne sont pas forcément entièrement mis en évidence par ces cartes. La dernière ligne correspond aux pseudos masques obtenus à partir des CAM (à comparer avec la deuxième ligne). Le modèle de classification utilisé est ResNet-50 [66].

A.2 Méthodes d'amélioration des CAM

La génération d'un CAM pour chaque image de la base d'entrainement constitue la première étape (voir Figure A.2) du processus d'entraînement pour la segmentation sémantique faiblement supervisée (WSSS). Il permet ensuite de générer pour chaque image un pseudo masque utilisé comme vérité terrain pour entrainer le modèle de segmentation. Toutefois, comme illustré à la Figure A.3, le CAM ne met en évidence que des parties limitées des objets, ce qui s'avère insuffisant pour générer des pseudo masques de bonne qualité (voir Figure A.3, dernière ligne). Cela impacte directement l'apprentissage du modèle de segmentation et par conséquent, ses performances qui sont médiocres.

Les limitations du CAM découlent du fait qu'il est généré à partir de la dernière couche d'un réseau de neurone convolutionnel (typiquement ResNet-50 [66] comme pour les résultats de la Figure A.2) dont la tâche est de classifier les objets dans les images à partir de la seule connaissance des étiquettes au niveau de l'image (i.e. les objets/Classes présents dans chaque image). Pour obtenir des CAM de meilleure qualité (i.e. meilleure localisation des zones discriminantes de chaque objet dans l'image) nous proposons dans ce chapitre deux approches qui consistent à mieux contrôler l'apprentissage du classifieur en ajoutant des termes à la fonction coût minimisée.

A.2.1 Génération des CAM et des pseudo masques

Méthode classique d'obtention des CAM

La méthode générale pour obtenir le CAM $\boldsymbol{A}$ consiste dans un premier temps à entrainer un classifieur qui prend en entrée des images et les étiquettes (les classes des objets présents dans chaque image) associées. L'entrainement du classifieur suit la processus schématisé à la Figure A.4. La dernière couche d'un réseau de neurones convolutionnel (par exemple ResNet-50) fourni les cartes de caractéristiques $\boldsymbol{F} \in \mathcal{R}^{h \times w \times n}$ pour une image d'entrée $\boldsymbol{I} \in \mathcal{R}^{H \times W \times 3}$. La taille de $\boldsymbol{F}$ dépend de l'architecture du CNN et des dimensions de l'image d'entrée. Nous appliquons alors une nouvelle étape de convolution afin d'obtenir la matrice $\boldsymbol{D} \in \mathcal{R}^{h \times w \times l}$, dont la profondeur (le nombre de canaux) correspond au nombre de classes l de la base d'entrainement. Afin d'obtenir les scores $\boldsymbol{s}^c$ associés à chaque classe c, on applique un regroupement moyen global (GAP) pour chaque canal de $\boldsymbol{D}$ puis une fonction sigmoïde pour des raisons d'échelles et d'interprétation du résultat.

Soit $\mathcal{O} = \{1, 2, ..., l\}$ l'ensemble des indices-labels de classe, numérotés de 1 à l, présents dans la base d'apprentissage. Pour une image donnée $\boldsymbol{I}$, il est possible d'observer plusieurs classes. Soit $\mathcal{Y}$ l'ensemble des labels associés, tel que $\mathcal{Y} \subset \mathcal{O}$. La fonction coût à minimiser est définie par

$$L_{cls_s}(\boldsymbol{s}, \mathcal{Y}) = -\frac{1}{l}\left(\sum_{c \in \mathcal{Y}} \log(\boldsymbol{s}^c) - \sum_{c \in \mathcal{O} \setminus \mathcal{Y}} \log(1 - \boldsymbol{s}^c)\right). \tag{A.1}$$

Une fois l'apprentissage réalisé, le CAM $\boldsymbol{A}$ est obtenu en appliquant une fonction d'activation (ici la ReLU) à $\boldsymbol{D}$ puis une normalisation. Ainsi pour un pixel d'indice i (lecture lexicographique de la matrice) et une classe c nous avons

$$\boldsymbol{A}_i^c = \frac{\mathrm{ReLU}(\boldsymbol{D}_i^c)}{\max(\boldsymbol{D}_i^c) + \epsilon}, \tag{A.2}$$

où ϵ est une valeur très petite permettant d'éviter des divisions par zéro. Notons que le CAM $\boldsymbol{A}$ est donc une matrice tri-dimensionnelle de même dimension que $\boldsymbol{D}$.

Méthode classique de génération des pseudo masques à partir des CAM

Conséquence de l'architecture des CNN, le CAM a une résolution spatiale réduite par rapport à l'image initiale. Pour obtenir le pseudo masque il faut donc dans un premier temps redimensionner $\boldsymbol{A}$. Pour cela nous appliquons une interpolation bilinéaire qui conduit au CAM $\boldsymbol{A'} \in \mathcal{R}^{H \times W \times l}$. La méthode classique pour obtenir un pseudo masque bidimensionnel $\boldsymbol{M}$ à partir de $\boldsymbol{A'}$ consiste à appliquer

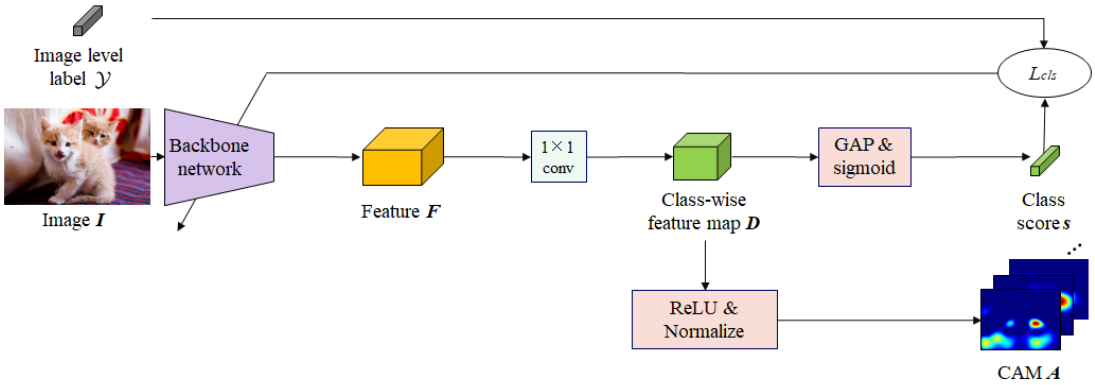


Figure A.4: Processus d'entrainement du classifieur permettant de générer le CAM.

un seuillage et une fonction argmax (suivant la troisième dimension de $\boldsymbol{A'}$) :

$$\boldsymbol{M}_i = \begin{cases} \underset{c \in \mathcal{Y}}{\text{argmax}}(\boldsymbol{A}_i'^c) & \text{si } \underset{c \in \mathcal{Y}}{\max}(\boldsymbol{A}_i'^c) > T_{fg}, \\ 0 & \text{sinon.} \end{cases} \tag{A.3}$$

$\boldsymbol{A}_i'^c$ est la valeur du CAM redimensionné pour la classe c et i l'indice de position du pixel. $\boldsymbol{M}_i$ est la classe du pixel i. T_{fg} est un seuil à fixer. Par convention, l'arrière-plan (i.e le fond de l'image ou encore tout pixel dont la classe n'est pas dans $\mathcal{O}$) est codé comme étant la classe 0.

A.2.2 CAM excluant les zones de recouvrement

Lors de nos expérimentations portant sur les méthodes WSSS de la littérature, nous avons noté que les CAM faisaient régulièrement apparaître des zones de recouvrement entre classes ou avec la classe 'fond' (i.e. entre différents canaux du CAM), ce qui peut être source d'ambiguïté. Idéalement, on pourrait penser que le CAM devrait présenter des valeurs fortes uniquement dans les zones spatiales de la carte correspondant à la classe de l'objet et des valeurs très faibles dans les mêmes zones des autres cartes.

La méthode proposée dans cette section s'appuie sur ce constat et cherche donc à limiter ces zones de recouvrement. En résumé, l'idée est de forcer chaque pixel de l'image à n'appartenir qu'à une seule classe. Pour cela nous proposons de modifier la fonction coût utilisée pour entrainer le classifieur. La procédure est schématisée à la Figure A.5.

Considérons qu'il existe l classes dans la base d'apprentissage, soit $\mathcal{O} = \{1, 2, ..., l\}$ l'ensemble de ces classes, numérotées de 1 à l. Pour une image $\boldsymbol{I}$ donnée, il est possible d'avoir plusieurs classes présentent. Soit $\mathcal{Y}$, tel que $\mathcal{Y} \in \mathcal{O}$ et $|\mathcal{Y}| > 1$, l'ensemble de ces classes. Pour évaluer si il existe des recouvrement entre les cartes du CAM $\boldsymbol{A}$ prises deux à deux (nous rappelons qu'une carte du CAM correspond à une classe donnée) et leur occurence, nous calculons la matrice $\boldsymbol{U}$ dont chaque

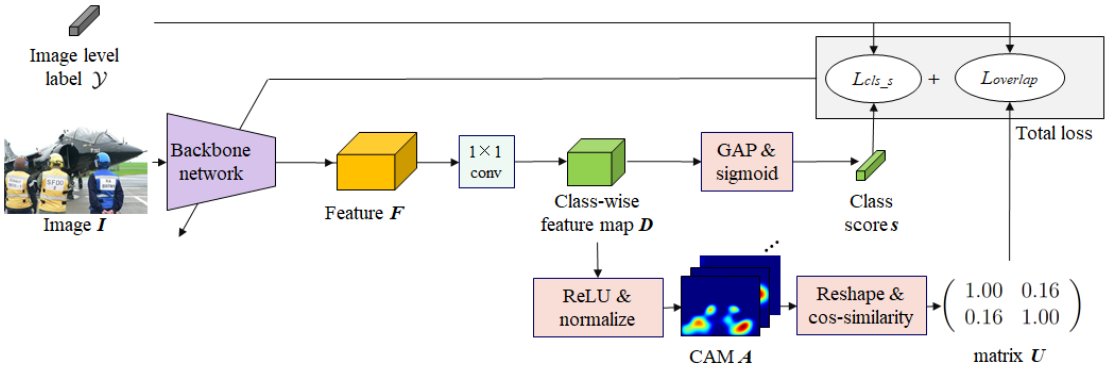


Figure A.5: Procédure d'entrainement du classifieur permettant d'obtenir des CAM excluant les zones de recouvrement.

élément est donné par

$$\boldsymbol{U}_{xy} = \frac{\boldsymbol{e}^{x\top}\,\boldsymbol{e}^{y}}{||\boldsymbol{e}^{x}||\;||\boldsymbol{e}^{y}|| + \epsilon}, \tag{A.4}$$

où $\boldsymbol{e}^{x}$ et $\boldsymbol{e}^{y}$ sont des vecteurs unidimensionnels obtenus en transformant les cartes d'activation $\boldsymbol{A}^{x}$ et $\boldsymbol{A}^{y}$ pour les classes x et y, avec $x, y \in \mathcal{Y}$, qui sont des matrices en vecteurs. La valeur de $\boldsymbol{U}_{xy}$ est en fait la cosimilarité entre $\boldsymbol{e}^{x}$ et $\boldsymbol{e}^{y}$. ϵ est une petite valeur pour éviter que le dénominateur soit nul.

Pour pénaliser les régions de chevauchement, une nouvelle fonction de perte $L_{overlap}$ est donc proposée :

$$L_{overlap}(\boldsymbol{U}, \mathcal{Y}) = \frac{1}{2} \sum_{x \in \mathcal{Y}} \sum_{y \in \mathcal{Y} | y \neq x} \boldsymbol{U}_{xy}. \tag{A.5}$$

$L_{overlap}$ permet de minimiser la valeur de $\boldsymbol{U}_{xy}$ afin d'orthogonaliser les vecteurs $\boldsymbol{e}^{x}$ et $\boldsymbol{e}^{y}$. Cette orthogonalisation force les cartes d'activation correspondantes à s'exclure mutuellement et donc à réduire les régions de chevauchement entre classes dans le CAM.

La fonction coût totale combine le terme $L_{overlap}$ avec la fonction coût initiale de la tâche de classification L_{cls_s} :

$$L = L_{cls_s}(\boldsymbol{s}, \mathcal{Y}) + \alpha L_{overlap}(\tilde{\boldsymbol{U}}, \mathcal{Y}), \tag{A.6}$$

où α est un hyperparamètre à fixer.

A.2.3 CAM obtenu à partir d'un classifieur entrainé avec différentes contraintes

La deuxième méthode a pour objectif d'améliorer l'activation des régions les moins discriminantes des objets dans les images. Pour cela, nous proposons de modifier

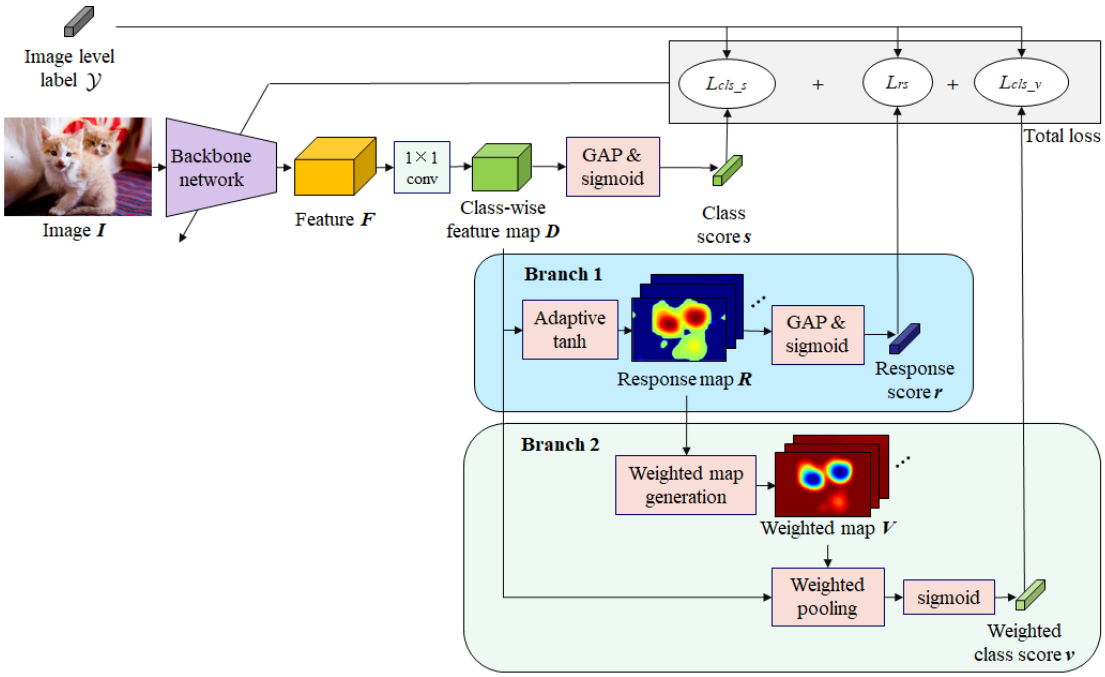


Figure A.6: Schéma de l'entraînement du modèle de classification avec des termes visant à pondérer les cartes d'activation.

le processus d'entrainement du classifieur en ajoutant deux thermes à la fonction coût, comme illustré à la Figure A.6.

La premier terme ('Branch 1' dans la Figure A.6) est obtenu en suivant deux étapes. La première consiste à calculer la matrice $\boldsymbol{R}$ à partir de $\boldsymbol{D}$:

$$\boldsymbol{R}_n^i = \begin{cases} \tanh(\tau \boldsymbol{D}_i^c) & \text{si } \boldsymbol{D}_i^c >= 0, \\ \tanh(\boldsymbol{D}_i^c) & \text{sinon.} \end{cases} \tag{A.7}$$

où τ est un paramètre permettant de rééchelonner les valeurs positives.

Puis, pour obtenir le score $\boldsymbol{r}^c$ pour la classe c on applique un GAP et une fonction sigmoïde à $\boldsymbol{R}^c$:

$$\boldsymbol{r}^c = f(\mathrm{GAP}(\boldsymbol{R}^c)), \tag{A.8}$$

où $f(\cdot)$ représente la fonction sigmoïde.

La fonction coût est

$$L_{rs} = -\frac{1}{l}\left(\sum_{c\in\mathcal{Y}} \boldsymbol{r}^c - \sum_{c\in\mathcal{O}\setminus\mathcal{Y}} \boldsymbol{r}^c\right). \tag{A.9}$$

L'opposé de la fonction coût $-L_{rs}$ mesure le contraste entre la réponse moyenne du réseau aux classes de l'image et sa réponse aux autres classes. La première étape de normalisation de $\boldsymbol{D}$ par une tangente hyperbolique permet de limiter l'impact des régions très discriminantes de l'image dont le score se trouve plafonnée. Le plafonnement entrave l'augmentation de la mesure de contraste dans ces régions et conduit l'optimisation à augmenter ce contraste en cherchant à augmenter la

contribution d'autres régions moins discriminantes et en réduisant la contribution des classes non présentes dans l'image.

Pour le second terme ajouté à la fonction coût, nous souhaitons obtenir des cartes pondérées. Pour cela nous commençons par déterminer $\boldsymbol{V}$. Pour une classe c donnée et un pixel i

$$\boldsymbol{V}_i^c = \min(1 - \boldsymbol{R}_i^c, 1). \tag{A.10}$$

Le score $\boldsymbol{v}$ pour une classe c est

$$\boldsymbol{v}^c = \frac{1}{|\mathcal{U}|} \sum_{i \in \mathcal{U}} \boldsymbol{D}_i^c \boldsymbol{V}_i^c. \tag{A.11}$$

La fonction coût correspondante est

$$L_{cls_v}(\boldsymbol{v}, \mathcal{Y}) = -\frac{1}{l} \left(\sum_{c \in \mathcal{Y}} \log(\boldsymbol{v}^c) - \sum_{c \in \mathcal{O} \setminus \mathcal{Y}} \log(1 - \boldsymbol{v}^c) \right). \tag{A.12}$$

Le premier terme de L_{cls_v} encourage le modèle à prédire des valeurs plus élevés, pour les valeurs positives, dans les zones des objets qui n'étaient pas discriminantes avec le CAM classique. Le deuxième terme tend à réduire les valeurs négatives.

Finalement, la fonction coût considérée dans cette méthode est

$$L = L_{cls_s}(\boldsymbol{s}, \mathcal{Y}) + \beta L_{rs}(\boldsymbol{r}, \mathcal{Y}) + \gamma L_{cls_v}(\boldsymbol{v}, \mathcal{Y}), \tag{A.13}$$

où β et γ sont des hyperparamètres à fixer.

A.2.4 Résultats expérimentaux

Pour mener les expérimentations et comparer les différentes approches nous avons utilisé le jeu de données PASCAL VOC 2012 [18], classiquement utilisé dans la littérature en WSSS (mais aussi pour d'autres problématiques).

Résultats pour la première approche

Les résultats obtenus avec la première approche sont illustrés par la Figure A.7. La première ligne correspond aux images originales, nous avons pris ici des images contenant deux classes afin de simplifier l'affichage des résultats. La seconde ligne donne la vérité terrain associée à chaque image, les troisième et quatrième ligne correspondent aux résultats du CAM classique pour chaque classe, et les lignes 5 et 6 fournissent les résultats pour la première méthode. On peut noter que cette méthode permet effectivement de réduire les zones de recouvrements et améliore la mise en évidence des zones d'intérêt.

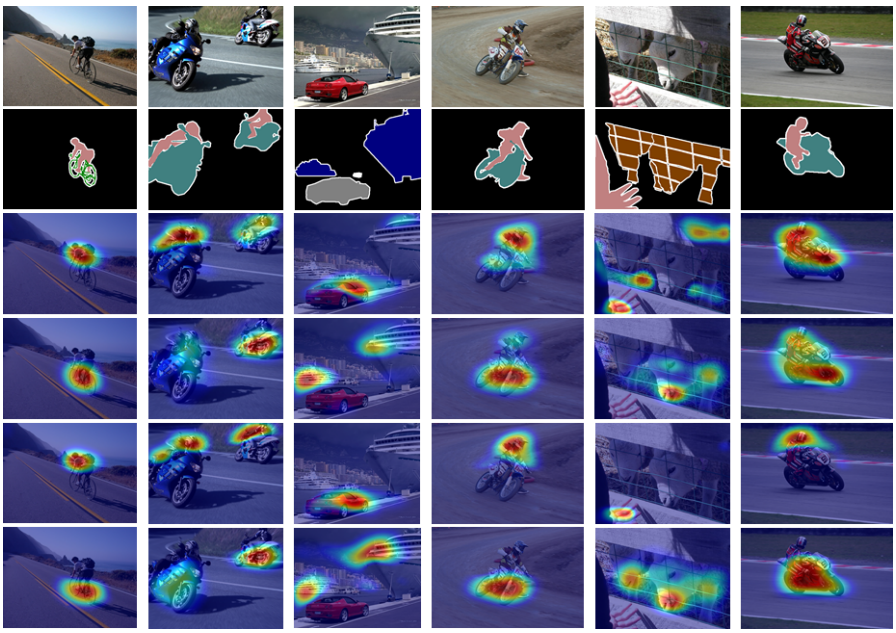


Figure A.7: Comparaison entre le CAM de base et CAM amélioré pour des images du jeu de données d'entraînement PASCAL VOC 2012 avec deux classes. De haut en bas : image originale, masque de vérité terrain, CAM de base pour les deux classes, et CAM amélioré pour les deux classes respectives.

Toutefois, le CAM obtenu avec la première méthode n'apporte pas d'amélioration pour toutes les classes; la pénalisation des régions de chevauchement pendant l'entraînement oblige le modèle à activer d'autres zones liées au fond de l'image (qui ont des caractéristiques communes à plusieurs images de la base de donnée), entraînant un taux plus élevé de faux positifs.

La figure A.8 montre les pseudo masques obtenus à partir des CAM classiques et des CAM de la première méthode. Visuellement certains résultats paraissent meilleurs et d'autres non. Mais ils font aussi appraître que pour des objets adjacents ou superposés la limite entre les deux objets est classée comme classe 0 (arrière-plan). Cela va avoir un impact souvent négatif sur les performances, comme on peut le voir dans le tableau A.1. Notons qu'un point positif de cette méthode est que les classes présentes dans l'image sont mieux détectées dans le CAM : il existe généralement des activations significatives uniquement dans les CAM correspondants aux classes présentes dans l'image (nous ne proposons pas d'exemple de résultat dans ce résumé).

Du fait de ces résultats nous n'avons pas poursuivi les expérimentations anticipant des résultats de segmentation moins bons qu'avec la méthode classique.

Résultats pour la deuxième approche

La Figure A.9 montre que les CAM obtenus par la méthode d'entraînement avec des scores liés à la réponse couvre de plus grandes parties des objets cibles par rapport aux cartes d'activation des CAM classiques. De plus, l'entraînement

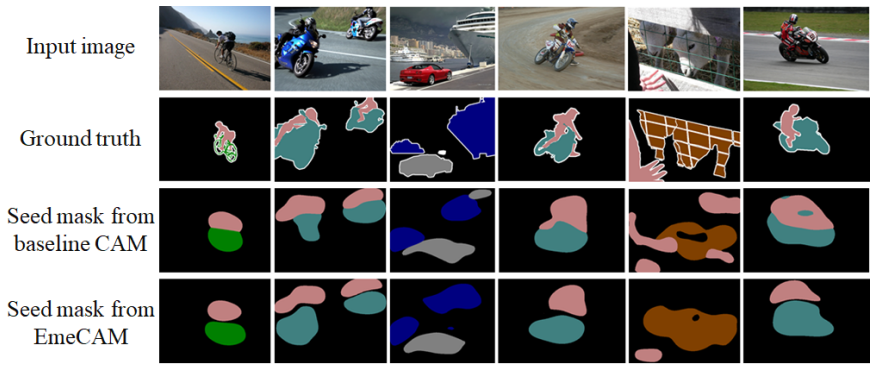


Figure A.8: Comparison entre les pseudo masques obtenus avec la méthode de base de génération des pseudo masque (ligne 3) et notre méthode (ligne 4). Les images sont les même que dans la Figure A.7.

	train	augmented train
BaseCAM	40.0	41.7
EmeCAM	36.4	37.2

Table A.1: mIoU (%) calculé sur l'ensemble de la base d'entrainement de PASCAL VOC pour les pseudo masques obtenus à partir du CAM classique (noté BaseCAM) et de notre CAM (noté EmeCAM).

avec les deux branches permet d'obtenir les plus grandes régions pour les images données, indiquant que l'utilisation de la seconde branche amène le modèle à explorer davantage les régions les moins discriminantes des objets. Cependant, cette méthode provoque des problèmes d'activation de zones du fond qui sont fréquemment prédites comme étant membre de la classe adjacente.

Comme le montre la figure A.10, les pseudo masques générés à partir des CAM obtenus avec la deuxième méthode couvrent davantage les objets. Cependant cela entraîne aussi une dégradation des prédictions des zones d'arrière-plan (classe 0). En effet, il est assigné aux zones adjacentes d'un objet la classe de celui-ci. Comme précédemment, cette conséquence de la méthode conduit à une dégration des résultats, comme illustré au tableau A.2.

Ainsi, comme pour la première méthode, il ne nous a pas semblé utile de poursuivir les expérimentations et segmenter les images, sachant que les résultats seraient moins bons qu'avec la méthode classique.

Conclusion

L'ensemble des résultats de cette partie montrent que les méthodes peuvent traiter efficacement les limites du CAM comme prévu, par exemple en diminuant les régions de chevauchement ou en augmentant l'activation du CAM dans les régions moins discriminantes. Cependant, ces améliorations sont contre balancées par des effets opposés conséquences des objectifs des méthodes.

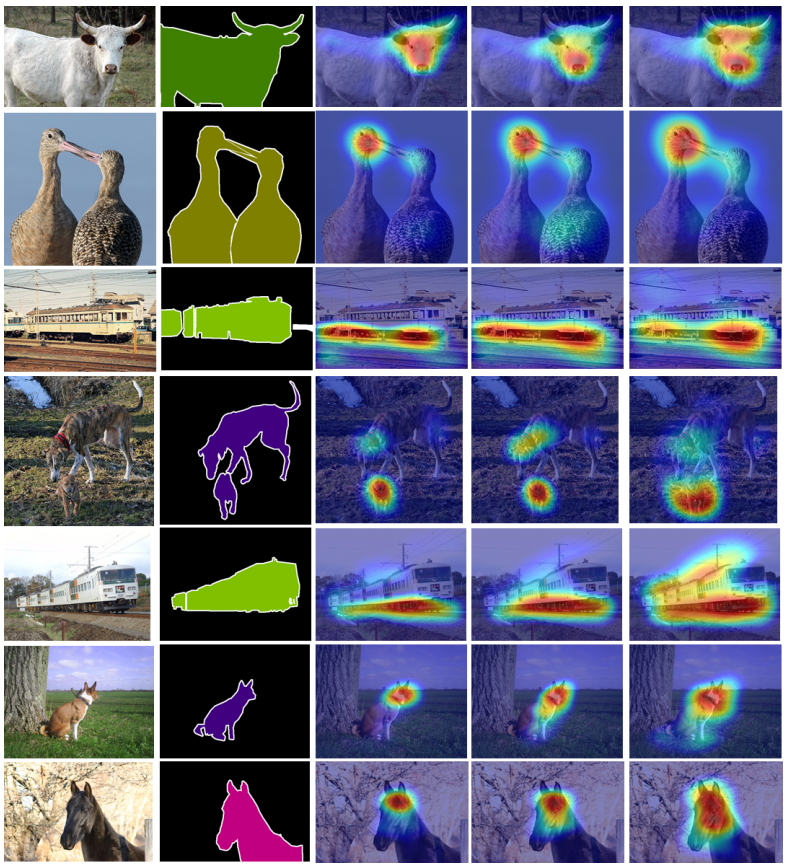


Figure A.9: Comparaison du CAM classique et de la méthode proposée pour des images du jeu de données d'entraînement PASCAL VOC 2012. De haut en bas : image originale, masque de vérité terrain, CAM de base, CAM obtenu avec la méthode d'entraînement utilisant L_{rs}, et CAM obtenu en utilisant l'entraînement avec L_{rs} et L_{cls_v}.

Pour la première méthode, la pénalisation des régions de chevauchement entraîne une diminution des valeurs d'activation notamment dans les régions adjacentes entre deux objets, conduisant à une affectation de ces régions à l'arrière-plan. Pour la seconde méthode, les objets sont mieux couverts mais apparaît une suractivation dans les régions d'arrière-plan dégradant la prédiction de cette dernière. En dépit de la capacité de contrôle gagnée par ces méthodes sur le CAM, le paramétrage d'un équilibre optimal entre les deux méthodes semble très difficile à obtenir et le gain moyen trop hypothétique.

A.3 Méthode d'amélioration des pseudo masques

Dans les méthodes WSSS récentes, l'obtention de pseudo masques de bonne qualité à partir du CAM implique généralement deux étapes : l'expansion (expansion en anglais) et l'affinage (refinement en anglais), comme le montre la figure A.11. Ces méthodes n'ayant pas d'impact sur l'étape de génération du CAM elles peuvent être appliquées après n'importe quelles approches de détermination des CAM.

baseline	L_{rs}	L_{cls_v}	train	augmented train
✓			40.0	41.7
✓	✓		37.2	39.6
✓	✓	✓	34.3	38.9

Table A.2: mIoU (%) des pseudo masques en fonction des éléments considérés. La base de donnée est la base d'entrainement augmené de PASCAL VOC 2012.

Les méthodes dans [25, 41, 42] proposent des post-traitements qui partent du principe que les régions discriminantes dans la CAM d'une image contiennent des informations fiables qui peuvent être utilisées pour déduire les classes des zones non discriminantes. Ces méthodes sont appelées méthodes d'expansion, car elles permettent à la région étiquetée avec confiance de s'étendre au fur et à mesure que le processus se poursuit. Les méthodes [28, 29, 116] sont considérées comme des processus d'affinage dans les approches WSSS récentes, car elles sont proposées pour améliorer la discrimination des limites entre les objets présents dans l'image et l'arrière-plan.

Dans ce chapitre, une nouvelle méthode de génération des pseudo masques est proposée. Celle-ci exploite un classifieur dont les paramètres sont spécifiques à chaque image de la base d'entrainement et qui prend en comptes différentes sources afin de pallier le manque de résolution du CAM tout en conservant l'apport sémantique de ce dernier.

A.3.1 Méthode d'expansion proposée

Comme illustré dans la Figure A.12, la méthode proposée comprend trois étapes. A partir du CAM nous générons une 'graine' (appelé seed par la suite), suivant une approche équivalente à la génération classique des pseudo masques. Ensuite, un processus de sélection d'échantillons, utilisant plusieurs caractéristiques issues du CAM, du seed et des couches du CNNN, est mis en œuvre. Ce processus fourni un ensemble d'entraînement pour un classificateur qui servira à inférer les labels de l'ensemble des pixels de l'image (à partir de différentes sources). Dans cette étude, nous considérons un classifieur basé sur les machines à vecteurs de support (SVM : Support Vector Machines). Afin d'améliorer la performance globale du SVM, nous proposons également processus itératif permettant de corriger itérativement les étiquettes des pixels dans l'ensemble d'entraînement du SVM en fonction des résultats de classification du SVM (entraîné en utilisant les étiquettes précédentes). Les prédictions du SVM ainsi entraîné fournissent le masque de classification pixel par pixel, qui peut être directement utilisé comme pseudo masques ou comme entrée du processus de raffinement pour améliorer encore ces pseudo masques.

Génération du seed

Comme précédemment indiqué et illustré Figure A.12, la première étape consiste à générer le seed $\boldsymbol{M}$ à partir du CAM. Comme pour la génération classique des



Figure A.10: Comparison entre les pseudo masques obtenus à partir du CAM classique ou de la deuxième méthode. La première ligne correspond aux image originales, la deuxième ligne est la vérité terrain, la troisième ligne correspond aux pseudo masques pour le CAM classique, la quatrième ligne correspond aux pseudo masque en ne considérant que la première branche de la deuxième approche (i.e. L_{rs}), et la dernière ligne correspond aux pseudo masques obtenus avec la deuxième méthode (incluant L_{rs} et L_{cls_v}).

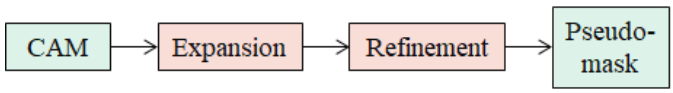


Figure A.11: Génération de pseudo masques de bonnes qualité à partir des CAM via l'utilisation de deux étapes de post-traiements.

pseudo masque, cela consiste à appliquer une interpolation bilinéaire afin de redimensionner le CAM $\boldsymbol{A}$ et obtenir $\boldsymbol{A'} \in \mathcal{R}^{H \times W \times l}$. Pour un pixel d'indice i, le seed est alors

$$\boldsymbol{M}_i = \begin{cases} 0 & \text{si } \max\limits_{c \in \mathcal{Y}}(\boldsymbol{A'}_i^c) < T_{bg}, \\ \arg\max\limits_{c \in \mathcal{Y}}(\boldsymbol{A'}_i^c) & \text{si } \max\limits_{c \in \mathcal{Y}}(\boldsymbol{A'}_i^c) > T_{fg}, \\ -1 & \text{sinon}, \end{cases} \tag{A.14}$$

où $\boldsymbol{A'}_i^c$ est la valeur du pixel d'indice i, avec $i \in \mathcal{U}$ l'ensemble des indices de position (lecture lexicographique), du CAM redimensionné $\boldsymbol{A'}$. Notons que $\boldsymbol{M}_i$ est égale à 0 pour le pixel i dont la classe prédite est l'arrière-plan, -1 lorsque la classe est incertaine et la valeur de la classe sinon.

Sélection des échantillons pour l'ensemble d'entrainement du SVM

Dans une première étape nous devons définir les données qui seront utilisées pour inférer le masque de classification servant soit directement de pseudo masque soit d'entrée au processus d'affinage, qui fournit en sortie le pseudo masque.

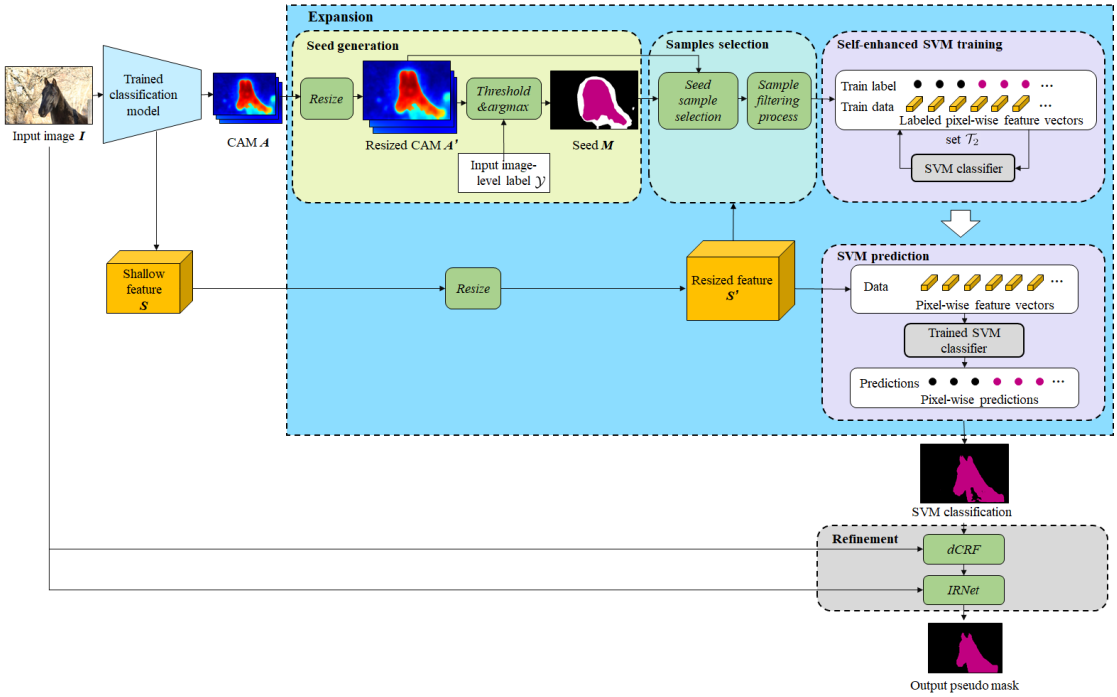


Figure A.12: Schéma détaillant les étapes de la méthode proposée.

L'ensemble de la base de données est $\mathcal{T}_0 = \{(\boldsymbol{S}'_i, \boldsymbol{M}_i)|i \in \mathcal{U}\}$, où $\boldsymbol{S'}$ est la version de la sortie d'une couche $\boldsymbol{S}$ du CNN redimensionnée aux dimensions spatiales de l'image originale et $\boldsymbol{S}'_i$ est le vecteur contenant les valeurs de l'ensemble des cartes contenues dans $\boldsymbol{S}$ pour le pixel d'indice i .

Il faut ensuite sélectionner des échantillons dans cet ensemble $\mathcal{T}_0$ afin d'entrainer le SVM. Ce sous-ensemble doit être représentatif des classes présentent dans l'image afin de pouvoir inférer les données $\boldsymbol{S'}$ de $\mathcal{T}_0$ et obtenir un masque de classification de bonne qualité.

Un premier sous-ensemble $\mathcal{T}_1$ de $\mathcal{T}_0$ est défini par sélection d'échantillons. Différentes règles sont appliquées pour cette sélection suivant les différentes classes. Pour l'arrière-plan, n échantillons sont tirés aléatoirement suivant une loi uniforme à partir de pixels dont les valeurs de départ sont 0 (dans $\boldsymbol{M}$). Pour les objets, un échantillonnage aléatoire pondéré est mis en œuvre pour donner plus de poids aux pixels ayant des valeurs de CAM élevées. Le poids $\boldsymbol{P}_i^c$, à l'emplacement i et pour la classe c est déterminé par

$$\boldsymbol{P}_i^c = \frac{\boldsymbol{A'}_i^c - T_{fg}}{1 - T_{fg}}. \tag{A.15}$$

On sélectionne n échantillons pour chaque classe. De fait, $\mathcal{T}_1$ contient $(|\mathcal{Y}| + 1)n$ échantillons défini par des paires $(\boldsymbol{S'}_k, \boldsymbol{M}_k)$, pour la position $k \in \mathcal{K} \subset \mathcal{U}$ sélectionnée.

Ensuite, un sous-ensemble $\mathcal{T}_2$ de $\mathcal{T}_1$ est obtenu en supprimant de $\mathcal{T}_1$ les échantillons étiquetés comme étant des objets mais qui sont trop similaires à l'arrière-plan. Étant donné que l'arrière-plan est relativement bien étiqueté dans $\boldsymbol{M}$, on définit

un prototype de l'arrière-plan, noté $\boldsymbol{S'_{bg}}$, de la façon suivante

$$\boldsymbol{S'_{bg}} = \frac{1}{n} \sum_{k \in \mathcal{K}} \boldsymbol{S'}_k \mathbb{1}_{(\boldsymbol{M}_k = 0)}, \tag{A.16}$$

où $\mathbb{1}_{(.)}$ est la fonction indicatrice, qui vaut 1 lorsque la condition est remplie et 0 dans le cas contraire.

Pour mesurer la similarité $\boldsymbol{w}_k$ entre un échantillon 'objet' (c'est-à-dire $\boldsymbol{M}_k > 0$) et l'arrière-plan, on utilise une cosimilarité entre les données de l'échantillon $\boldsymbol{S'}_k$ et le prototype $\boldsymbol{S'_{bg}}$:

$$\boldsymbol{w}_k = \frac{\boldsymbol{S'^{\top}_{bg}}\, \boldsymbol{S'}_k}{||\boldsymbol{S'_{bg}}||\ ||\boldsymbol{S'}_k||}, \tag{A.17}$$

où $\boldsymbol{S'^{\top}_{bg}}$ est la transposée de $\boldsymbol{S'_{bg}}$.

Lorsque $\boldsymbol{w}_k > T_s$ (un seuil à fixer), l'échantillon 'objet' à l'emplacement k présente trop de similarité avec le prototype d'arrière-plan, ce qui peut correspondre à une prédiction incorrecte, est exclu de l'ensemble de données.

Le résultat de l'ensemble du processus est un sous-ensemble $\mathcal{T}_2 = \{(\boldsymbol{S'}_m, \boldsymbol{M}_m) | m \in \mathcal{M}\}$, qui est utilisé pour entrainer le SVM, ceci pour chaque image de la base d'apprentissage.

SVM auto-amélioré

Une procédure itérative d'entrainement d'un SVM est proposée pour améliorer les résultats. L'idée est de mettre à jour les étiquettes dans $\mathcal{T}_2$ à l'aide des étiquettes inférées $\boldsymbol{M}_m^{\boldsymbol{SVM}}$ par le SVM, entrainé à partir des étiquettes précédentes. Nous obtenons ainsi un nouveau $\mathcal{T}_2 = \{(\boldsymbol{S'}_m, \boldsymbol{M}_m^{\boldsymbol{SVM}}) | m \in \mathcal{M}\}$ qui est utilisé pour entrainer un nouveau classificateur SVM. Le processus est répété plusieurs fois.

Après plusieurs répétitions, tous les vecteurs de caractéristiques de $\boldsymbol{S'}$ (i.e. pour chaque position i dans $\boldsymbol{S'}$) sont classifiés grâce au dernier SVM entrainé. Les prédictions fournissent le masque de classification SVM, qui peut être utilisé comme pseudo-masque ou comme entrée du processus de raffinement.

A.3.2 Sélection de la couche du CNN et limites de performance de la méthode

Dans cette section, nous proposons d'évaluer le potentiel de la méthode proposée en estimant sa limite de performance. En effet, d'après le descriptif de la méthode proposée, il est clair que le processus de sélection des échantillons est l'étape critique de cette méthode. Ainsi, avant de déterminer les performances de la méthode, nous proposons d'évaluer une borne supérieure en utilisant la vérité terrain pour définir un ensemble d'entraînement exempt d'erreurs.

Comme le montre la Figure A.13, par rapport à la méthode proposée, nous supprimons l'étape de génération du seed et remplaçons le processus de sélection

d'échantillons par un échantillonnage aléatoire uniforme basé sur la vérité terrain. L'ensemble d'entraînement du SVM obtenu, composé de paires de vecteurs de caractéristiques et des étiquettes provenant du masque de vérité terrain, est donc parfaitement étiqueté. Il n'est alors plus nécessaire d'utiliser le processus d'apprentissage itératif du SVM. Le processus de raffinement est également supprimé puisque l'objectif est d'évaluer la performance intrinsèque de la méthode, sans post-traitement.

D'autre part, la méthode utilise une des couches du CNN. Les limites de performance sont donc évaluées en fonction de la couche choisie.

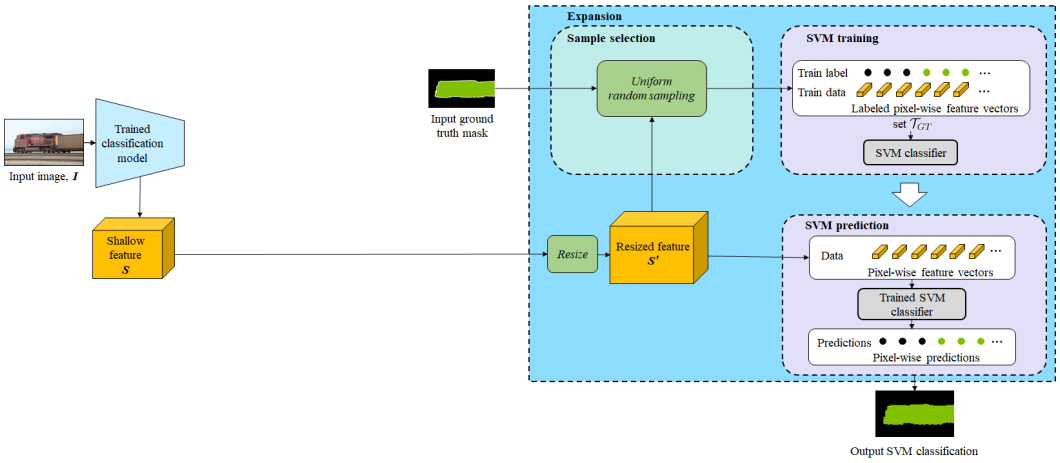


Figure A.13: Schéma du processus permettant d'évaluer la borne supérieure de la méthode proposée, en utilisant la vérité terrain.

Pour générer $\boldsymbol{S}$, dans cette étude la méthode appelée Self-supervised Image-specific Prototype Exploration (SIPE) a été choisie, car elle intègre et optimise les différentes caractéristiques produites par les diverses couches du réseau dans le processus d'apprentissage du classifieur. Le réseau sur lequel se base cette méthode est ResNet-50 [66]. Les couches du réseau sont notées F_1, F_2, F_3, F_4 et F_5. Pour tirer parti de la résolution de l'étiquetage de l'image, la sortie de chaque couche est redimensionnée à la taille de l'image originale. Dans la suite, $\boldsymbol{S'}$ de $\mathcal{T}_{GT}$ correspond donc à un F_i, $i \in \{1, 2, 3, 4, 5\}$, ou à l'image original, notée F_0.

En fonction du choix de $\boldsymbol{S'}$, le tableau A.3 présente l'IoU de chaque classe, obtenu en comparant la vérité terrain et la sortie du SVM (sortie qui fournit les pseudo masques utilisés pour l'apprentissage du réseau de segmentation), ainsi que le mIoU sur l'ensemble de données d'entraînement de PASCAL VOC 2012 [18]. On peut noter qu'en utilisant F_1, F_2 et F_3, les performances pour toutes les classes sont excellentes.

L'utilisation de F_1 donne les meilleures performances dans la plupart des catégories, ce qui suggère que lorsque la vérité terrain est disponible, notre méthode permet, avec F_1, un bon compromis entre les informations hautes résolution et la sémantique.

La méthode proposée, lorsqu'elle utilise la vérité terrain, donne une limite supérieure de 96,67% (mIoU), ce qui est presque parfait (i.e. la prédiction correspond presque

parfaitement à la vérité terrain). Cela montre le potentiel de la méthode, si un processus efficace de sélection d'un ensemble d'apprentissage pour le classifieur peut être défini.

Feature	bkg	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	motor	person	plant	sheep	sofa	train	tv	mIoU
F_0	86.98	68.79	35.91	64.40	57.83	58.97	68.77	62.19	77.31	58.65	73.76	69.01	71.25	73.22	57.99	68.02	61.07	70.89	74.65	64.80	60.96	65.97
F_1	**98.98**	**96.70**	**89.97**	**96.95**	**96.39**	95.61	**96.80**	**96.74**	**98.32**	**95.76**	**97.77**	97.74	**97.82**	**97.75**	**95.22**	**96.87**	**96.17**	**97.63**	97.82	**96.70**	96.13	**96.67**
F_2	98.65	92.87	85.61	95.38	93.90	95.34	95.66	95.63	98.16	94.89	96.77	96.76	97.23	96.86	93.93	95.85	95.02	97.22	97.47	95.80	95.19	95.44
F_3	98.58	87.62	84.83	93.02	91.31	**96.36**	96.74	95.63	97.91	95.57	95.73	**97.82**	96.44	96.31	93.88	96.27	95.31	96.53	**98.29**	96.52	**97.50**	95.15
F_4	96.98	78.48	75.53	86.10	84.00	93.96	94.35	91.75	95.39	91.04	87.35	96.74	90.63	88.29	89.19	93.85	91.03	88.22	96.43	93.71	96.69	90.46
F_5	90.19	55.38	66.41	61.51	60.54	77.25	80.48	75.81	82.78	80.74	72.03	91.36	72.52	71.02	73.94	85.84	81.31	72.71	82.62	73.93	86.00	75.92

Table A.3: Comparison entre la vérité terrain et la sortie du SVM en utilisant comme métrique le IoU pour chaque classe et le mIoU (%) global, en fonction du choix de $\boldsymbol{S'}$: F_0 (l'image originale), F_1, F_2, F_3, F_4 et F_5 (les différentes couches de ResNet-50). La base de données est l'ensemble d'entrainement de PASCAL VOC 2012. Les meilleurs résultats sont surlignés en gras. Notez que 'bkg' représente la classe arrière-plan.

A.3.3 Résultats expérimentaux

Dans un premier temps nous avons évalué les différents paramètres de la méthode proposée à savoir le choix de F_i, $i \in \{0, 1, 2, 3, 4, 5\}$, et le nombre d'itérations pour le SVM auto-amélioré. Contrairement aux résultats de la section précédente, il s'est avéré que F_2 permettait d'obtenir les meilleurs résultats, lorsque la vérité terrain n'est pas connues. D'autre part, le nombre d'itération qui conduit en moyenne aux meilleurs résultats est 3. Dans la suite, l'ensemble des résultats présentés utilisent ces valeurs de paramètres.

Nous avons aussi évalué l'impact de chaque étape de notre processus sur la qualité des pseudo masques générés. Le Tableau A.4 présente ces résultats en terme de mIoU sur la base d'entrainement PASCAL VOC 2012 et la base dite augmentée (qui contient plus de données). Ces résultats montrent clairement l'impact et l'apport de chaque étape de la méthode. Le mIoU obtenu avec l'ensemble des étapes de la méthode (quatrième ligne du tableau A.4) atteint 61,2% et 58,9% en fonction de la base. C'est une amélioration significative par rapport à la méthode de référence SIPE.

Les résultats obtenus pour le processus complet ont été déterminés. Le processus comprend la génération des pseudo masques et l'apprentissage du modèle de segmentation à partir de ces pseudo masques. Les résultats de segmentation sont présentés dans les tableaux A.5 et A.6 pour l'ensemble de validation et de test de PASCAL VOC 2012. Notre méthode, référencée MIP, est comparée à SIPE, méthodes sur lequel s'appuies notre approche, et différentes méthodes récentes de la littérature. Le modèle de segmentation final est DeeplabV2 qui utilise soit ResNet-38 [70] soit ResNet-101 [66].

Par rapport à la méthode de base SIPE, MIP montre des améliorations significatives à la fois dans l'ensemble de validation et dans l'ensemble de test.

De manière plus générale, la méthode basée sur MIP a obtenu les meilleures performances sur l'ensemble de validation, quel que soit le CNN utilisé. Bien que nous

baseline	SSS	SFP	SeT	train	augmented train
✓				58.6	56.8
✓	✓			59.0	57.1
✓	✓	✓		60.7	57.4
✓	✓	✓	✓	**61.2**	**58.9**

Table A.4: mIoU (%) en fonction des étapes considérées pour générer les pseudo masques, utilisant les ensembles d'entraînement PASCAL VOC 2012. SSS *pour Seed Sample Selection pour obtenir* $\mathcal{T}_1$. SFP *pour Sample Filtering Process pour obtenir* $\mathcal{T}_2$. SeT *pour Self-enhanced Training (SVM auto-amélioré). Les meilleurs résultats sont en gras.*

Method	Val	Test
SIPE [107]	68.2	69.5
ADELE [150]	68.6	68.9
TransCAM [151]	69.3	69.6
ASDT [152]	**69.7**	70.1
URN [115]	69.4	**70.6**
MIP	**69.7**	70.4

Table A.5: mIoU (%) obtenu en segmentation sur la base PASCAL VOC 2012 de validation et de test, en utilisant ResNet-38.

n'ayons pas obtenu les meilleurs résultats sur l'ensemble de test, avec ResNet-38, la performance se situe dans la plage de fluctuation statistique de la meilleure performance. Avec ResNet-101, nous sommes parmi les meilleurs dans l'ensemble de validation. Les résultats expérimentaux relativement proches entre les ensembles de validation et de test indiquent la cohérence de nos performances de segmentation sur différentes images. On peut noter que certaines méthodes (notamment celles qui affichent les meilleurs résultats sur la base de test) obtiennent de meilleurs résultats sur la base de test, avec une différence assez significative par rapport à la base de validation.

La Figure A.14 montre les résultats de segmentation à partir de l'ensemble de validation. Comme le montrent les exemples fournis dans la Figure A.14.(a), notre méthode démontre d'excellentes capacités de récupération de détails pour des objets de différentes tailles. De plus, comme illustré dans la Figure A.14.(b), la vérité terrain des images d'entraînement n'est pas toujours sans erreurs. Il est observé que les résultats expérimentaux de notre méthode peuvent quand même correctement segmenter ces régions.

Par rapport aux méthodes de référence et aux méthodes WSSS récentes, notre approche démontre une amélioration significative des pseudo masques, qui récupèrent plus de détails des objets et fournissent des limites plus claires.

Method	Val	Test
Iter_dCRF [125]	63.5	63.2
AdvCAM [114]	68.1	68.0
ReCAM [159]	68.5	68.4
ASDT [152]	68.5	68.4
PPL [160]	67.8	69.6
SIPE [107]	68.8	69.7
AMR [161]	68.8	69.1
VWL-L [145]	69.2	69.2
AMN [122]	69.5	69.6
W-OoD [162]	69.8	69.9
ESOL [121]	69.9	69.3
SPML [153]	69.5	71.6
OPC [154]	70.5	**71.8**
MIP	**70.6**	70.8

Table A.6: mIoU (%) obtenu en segmentation sur la base PASCAL VOC 2012 de validation et de test, en utilisant ResNet-101.

A.4 Conclusions et perspectives

A.4.1 Conclusions

Dans cette thèse, nous avons étudié des méthodes pour améliorer les pseudo masques pour la segmentation sémantique faiblement supervisée. Une analyse du pipeline d'apprentissage de WSSS avec des étiquettes au niveau de l'image révèle tout d'abord que la précision des pseudo masques, qui sont utilisés comme cibles d'apprentissage pour entrainer un modèle de segmentation, a un fort impact sur la performance de segmentation finale, et ensuite que la génération de pseudo masques peut être divisée en 2 étapes : l'obtention de la carte d'activation de classe (CAM) et la génération de pseudo masques à partir du CAM. De nouvelles méthodes ont été proposées pour améliorer ces deux étapes. Les conclusions des méthodes proposées sont résumées ci-dessous.

Méthodes d'amélioration des CAM

Dans la section A.2 (Chapitre 4 du manuscrit), nous observons que les limitations du CAM pour la segmentation sémantique peuvent être attribuées au fait que l'objectif principal du réseau de classification, qui fournit des caractéristiques pour générer le CAM, est de classer les objets cibles dans les images, sans tenir compte des propriétés du CAM elle-même. Pour renforcer certaines propriétés du CAM, telles que l'exclusion des régions de chevauchement (où plusieurs valeurs d'activation significatives apparaissent pour différentes classes) ou la couverture des objets, deux méthodes sont introduites. Le principe de ces méthodes est d'ajouter des termes à la fonction cout contrôlant l'apprentissage pour forcer les solutions, de façon plus ou moins contraignantes, à satisfaire certaines propriétés

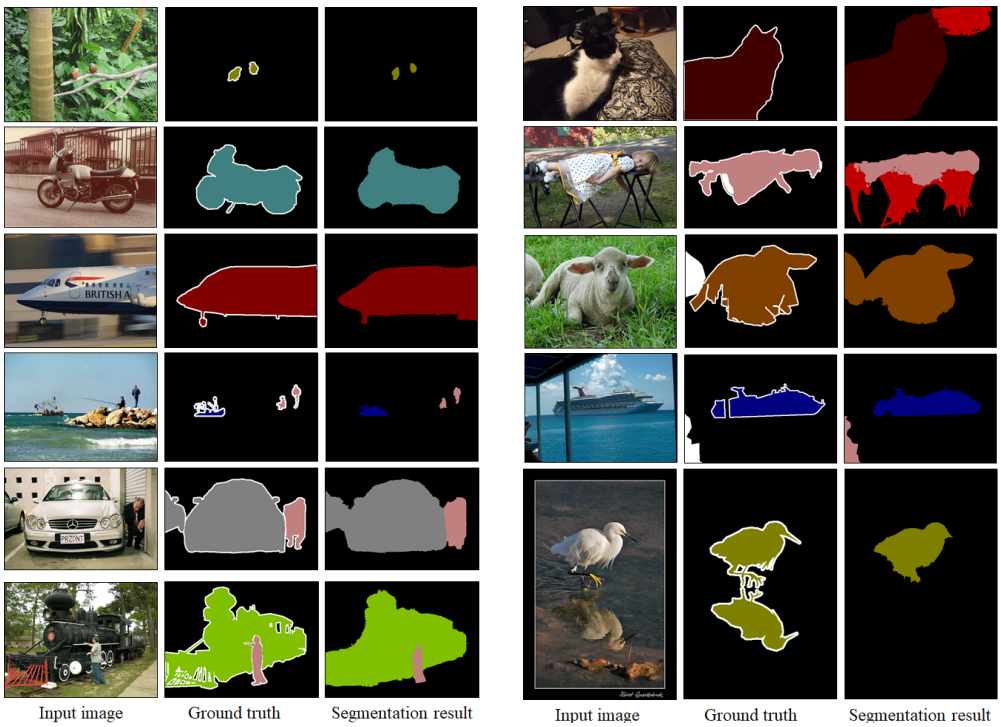


Figure A.14: Résultats de segmentation sur l'ensemble de validation PASCAL VOC 2012 (a) avec des objets de différentes tailles et (b) lorsque la vérité terrain n'est pas totalement exacte.

souhaitées pour le CAM.

La première méthode propose une nouvelle fonction de perte pour pénaliser les régions de chevauchement en cherchant à orthogonaliser les vecteurs issus des cartes d'activation des différentes classes. Les résultats expérimentaux montrent que les régions de chevauchement sont considérablement réduites et, comparé à la méthode classique, le CAM modifié se concentre davantage sur les régions sûres des objets cibles. La capacité de reconnaissance du modèle de classification s'améliore pour certaines classes. Cependant, on observe également que les zones activées dans le CAM obtenu sont globalement plus petites, ce qui limite l'apparition de régions de chevauchement, mais couvre moins bien les objets. Cela s'explique bien par le fait qu'une solution naturelle pour réduire les régions de chevauchement entre les classes consiste à réduire autant que possible les zones activées limitant par conséquent les risques de superposition. Ces zones activées couvrent donc une petite proportion des objets, ce qui n'est pas tout à fait conforme à notre objectif de segmentation.

En réponse aux résultats précédents, la deuxième méthode vise à étendre les régions d'objets produisant des réponses significatives dans le CAM. Il faut donc contraindre le réseau à produire des CAM avec des valeurs d'activation significatives dans des régions contenant des informations moins spécifiques des objets à classer. Pour ce faire, deux branches supplémentaires avec les objectifs proposés sont introduites dans le pipeline d'apprentissage du réseau de classification. La première

branche détermine un score de réponse moyenne par classe produite par le CAM normé à l'aide d'une fonction tanh dont la dynamique est adaptée. La deuxième branche donne un score à chaque classe en utilisant une nouvelle méthode de pondération du CAM qui donne moins d'importance aux grandes valeurs de la carte de la classe et d'avantage au régions plus incertaines. Les deux scores sont utilisés pour calculer une nouvelle fonction de perte qui réduit les contributions des grandes valeurs locales de la carte des caractéristiques de la classe à l'apprentissage pour forcer le réseau à donner plus d'importance à d'autres parties des objets à détecter. Les résultats expérimentaux montrent que ces modifications conduisent à des régions activées plus grandes, mais aussi à un problème de suractivation, de nombreuses zones d'arrière-plan étant activées de manière incorrecte dans le CAM. La cause principale de cette suractivation est que les zones faiblement activées comprennent non seulement les régions les moins discriminantes des objets, mais aussi les régions d'arrière-plan. Ainsi, en l'absence de contraintes suffisamment précises et spécifiques, le modèle augmente inévitablement la réponse dans les régions d'arrière-plan.

A partir de l'étude de ces méthodes, nous avons constaté que la dernière couche convolutive du réseau de classification ne peut fournir que des informations de faible résolution lors de la production des CAM, mais avec un bon niveau de sémantique. Ceci nous a amené à explorer l'utilisation d'informations issues des premières couches du réseau, ayant une meilleure résolution, afin de récupérer plus de détails. En outre, nous avons constaté la difficulté de trouver un critère d'amélioration des CAM universel tant la diversité des images est grande. Ce constat nous a conduit à valoriser la spécificité de chaque image pour tirer parti de ses caractéristiques propres et améliorer la précision de l'inférence des labels des pixels.

Méthode d'amélioration des pseudo masques

Dans la section A.3.1 (Chapitre 5 du manuscrit), le travail a été consacré aux traitements permettant d'améliorer les pseudo masques. Nous avons analysé et identifié deux limitations clés des méthodes de la littérature : (1) La plupart des méthodes exploitent soit des caractéristiques profondes produites par le réseau de classification, ces caractéristiques ont une résolution faible avec une sémantique riche, soit (2) elles exploitent l'image originale à haute résolution mais dont la sémantique locale est limitée. Les caractéristiques spécifiques de chaque image ne sont pas systématiquement exploités lors de la production des pseudo masques. Partant de cette analyse, nous proposons une nouvelle approche d'expansion pour générer des pseudo masques améliorés basés sur un classificateur de pixels spécialement entrainé pour chaque image. Ce classifieur utilise des caractéristiques produites par le réseau de classification (sortie des différentes couches) avec l'objectif de trouver un compromis entre la résolution et la sémantique.

Pour chaque image, il faut définir un ensemble d'apprentissage pour entrainer le classifieur. Le CAM, le seed et la sortie de différentes couches sont utilisées dans le processus de sélection d'échantillons proposé. Les échantillons sélectionnés (paires de caractéristiques et d'étiquettes déduite du CAM) sont utilisés pour entrainer un classificateur SVM. Un processus d'auto-amélioration est également introduit pour réduire l'impact des erreurs d'étiquetage dans l'ensemble d'apprentissage.

L'étude des performances de cette approche, en utilisant la vérité terrain pour sélectionner l'ensembles d'apprentissage du SVM, montre le potentiel de cette approche. Dans ce cas, le mIoU de la sortie du SVM peut dépasser les 96 %, ce qui est très supérieur aux résultats classiques.

Ces résultats montrent donc que cette méthode peut atteindre d'excellentes performances si la sélection de l'ensemble d'apprentissage pour le SVM comporte peu d'erreur. L'enjeu se déplace donc vers la définition d'un processus fiable de sélection de cet ensemble d'apprentissage.

Les expérimentations ont montré que le réseau de segmentation entraîné par les pseudo masques obtenus avec notre méthode sont plus performants que la plupart des méthodes WSSS récentes, tant avec l'ensemble de validation qu'avec l'ensemble de test de la base PASCAL VOC. Les résultats expérimentaux montrent que notre méthode présente d'excellentes capacités de segmentation des détails pour des objets de différentes tailles, tout en définissant des limites claires. Ces résultats soulignent l'importance d'obtenir des données fiables à partir de CAM imparfaits, et prouvent que les caractéristiques des premières couches du réseau apportent à la fois du contexte et une résolution qui améliorent efficacement les détails de la segmentation. Le classificateur de pixels, en étant défini spécifiquement pour chaque image cible les caractéristiques distinctives de cette image, sans avoir à déterminer une 'solution moyenne' moins adaptée à chaque cas particulier. La démarche entreprise parait donc pertinente pour améliorer l'apprentissage des modèles de segmentation en général.

A.4.2 Perspectives

A l'issue de ces travaux, la première remarque est que tout modèle convolutif peut être utilisé comme réseau de base des méthodes proposées dans cette thèse. Une évaluation des performances de ces méthodes sur différents réseaux pourrait être menée, notamment avec le réseau à haute résolution HRNET [174] conçu pour améliorer l'analyse des détails dans un cadre supervisé. Cette étude pourrait permettre de cerner avec plus de précision les apports et les lacunes des méthodes proposées.

Les résultats obtenus au Chapitre 4 révèlent que la suractivation du CAM dans les régions d'arrière-plan entraîne souvent un nombre élevé de fausses détections. Le développement de méthodes (sans utiliser de données auxiliaires) destinées à corriger ces pixels mal classés dans l'arrière-plan reste un sujet important à explorer pour les approches liées au WSSS. En s'inspirant du processus de filtrage des échantillons utilisé dans le Chapitre 5, une solution possible pourrait consister à introduire un prototype d'arrière-plan obtenu en calculant la moyenne des caractéristiques sémantiques par pixel dans les régions d'arrière-plan et à étiqueter comme arrière-plan les pixels qui partagent une forte similarité avec ce prototype.

Dans le Chapitre 5, nous utilisons séparément les cartes d'activation générées par les différentes couches du réseau pour caractériser les données de l'échantillon d'apprentissage. Une extension possible consisterait à exploiter simultanément

plusieurs caractéristiques peu profondes, pendant la phase d'échantillonnage, afin d'améliorer la fiabilité de l'ensemble d'apprentissage. Dans une première approche, non présenté dans ce mémoire, nous avons réalisé des expériences en concaténant simplement les caractéristiques de toutes les cartes d'activation du réseau. Toutefois, en faisant cela nous donne plus de poids aux couches ayant le plus de cartes et il s'en suit que les résultats obtenus sont très proches des résultats obtenus avec le classificateur entraîné avec les caractéristiques de la couche comptant le plus grand nombre de cartes. La fusion efficace des caractéristiques de plusieurs couches restent donc une question ouverte qu'il serait possible d'aborder comme une extension assez simple de ces travaux.

Compte tenu du fait que la précision des pseudo masques présente inévitablement un écart par rapport à la vérité de terrain, la conception d'un classificateur particulièrement robuste aux erreurs d'étiquetage est une piste de recherche intéressante et un défi ouvert dont l'investigation va au-delà de la question de la segmentation faiblement supervisée.

Par ailleurs, un autre axe de recherche intéressant consiste à combiner les WSSS avec les modèles basés sur Transformer.

Enfin certaines études récentes ont montré que la génération de CAM à partir de classifieurs d'objets conçus à l'aide de transformateurs plutôt que de CNN peut conduire à de meilleurs résultats [37, 151]. Comme ces modèles sont entrainés sur de grandes quantités de données, ils ont généralement de très bonnes capacités de généralisation. Une piste d'étude importante à explorer est l'adaptation de ces modèles pré-entraînés aux tâches WSSS pour des ensembles de données spécifiques (imagerie médicale, satellites...).

Bibliography

[1] Shijie Hao, Yuan Zhou, and Yanrong Guo. "A brief survey on semantic segmentation with deep learning". In: *Neurocomputing* 406 (2020), pp. 302–321.

[2] Alberto Garcia-Garcia et al. "A review on deep learning techniques applied to semantic segmentation". In: *arXiv preprint arXiv:1704.06857* (2017).

[3] Neeraj Sharma and Lalit M Aggarwal. "Automated medical image segmentation techniques". In: *Journal of Medical Physics* 35.1 (2010), pp. 3–14.

[4] Yan Wang et al. "Abdominal multi-organ segmentation with organ-attention networks and statistical fusion". In: *Medical Image Analysis* 55 (2019), pp. 88–102.

[5] Di Feng et al. "Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges". In: *IEEE Transactions on Intelligent Transportation Systems* 22.3 (2020), pp. 1341–1360.

[6] Khan Muhammad et al. "Vision-based semantic segmentation in scene understanding for autonomous driving: Recent achievements, challenges, and outlooks". In: *IEEE Transactions on Intelligent Transportation Systems* 23.12 (2022), pp. 22694–22715.

[7] Huihui Pan et al. "Deep dual-resolution networks for real-time and accurate semantic segmentation of traffic scenes". In: *IEEE Transactions on Intelligent Transportation Systems* 24.3 (2022), pp. 3448–3460.

[8] Xiaohui Yuan, Jianfang Shi, and Lichuan Gu. "A review of deep learning methods for semantic segmentation of remote sensing imagery". In: *Expert Systems with Applications* 169 (2021), p. 114417.

[9] Yan Liu et al. "Efficient patch-wise semantic segmentation for large-scale remote sensing images". In: *Sensors* 18.10 (2018), p. 3232.

[10] Jonathan Long, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2015, pp. 3431–3440.

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

[12] Tsung-Yi Lin et al. “Feature pyramid networks for object detection”. In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2017, pp. 2117–2125.

[13] Liang-Chieh Chen et al. “Semantic image segmentation with deep convolutional nets and fully connected crfs”. In: *arXiv preprint arXiv:1412.7062* (2014).

[14] Liang-Chieh Chen et al. “Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs”. In: *IEEE transactions on Pattern Analysis and Machine Intelligence* 40.4 (2017), pp. 834–848.

[15] Liang-Chieh Chen et al. “Rethinking atrous convolution for semantic image segmentation”. In: *arXiv preprint arXiv:1706.05587* (2017).

[16] Liang-Chieh Chen et al. “Encoder-decoder with atrous separable convolution for semantic image segmentation”. In: *Proceedings of the European Conference on Computer Vision*. 2018, pp. 801–818.

[17] Lyndon Chan, Mahdi S Hosseini, and Konstantinos N Plataniotis. “A comprehensive analysis of weakly-supervised semantic segmentation in different image domains”. In: *International Journal of Computer Vision* 129.2 (2021), pp. 361–384.

[18] Mark Everingham et al. “The pascal visual object classes (voc) challenge”. In: *International Journal of Computer Vision* 88 (2010), pp. 303–338.

[19] Gabriel J Brostow, Julien Fauqueur, and Roberto Cipolla. “Semantic object classes in video: A high-definition ground truth database”. In: *Pattern Recognition Letters* 30.2 (2009), pp. 88–97.

[20] George Papandreou et al. “Weakly-and semi-supervised learning of a deep convolutional network for semantic image segmentation”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 1742–1750.

[21] Bingfeng Zhang et al. “Reliability does matter: An end-to-end weakly supervised semantic segmentation approach”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 2020, pp. 12765–12772.

[22] Bingfeng Zhang et al. “End-to-end weakly supervised semantic segmentation with reliable region mining”. In: *Pattern Recognition* 128 (2022), p. 108663.

[23] Bingfeng Zhang. “Learning Semantic Segmentation with Weak and Few-shot Supervision”. PhD thesis. University of Liverpool, 2022.

[24] Nikita Araslanov and Stefan Roth. “Single-stage semantic segmentation from image labels”. In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2020, pp. 4253–4262.

[25] Sungpil Kho et al. “Exploiting shape cues for weakly supervised semantic segmentation”. In: *Pattern Recognition* 132 (2022), p. 108953.

[26] Yi Li et al. “Pseudo-mask matters in weakly-supervised semantic segmentation”. In: *Proceedings of the IEEE international conference on computer vision*. 2021, pp. 6964–6973.

[27] Sanghyun Jo, In-Jae Yu, and Kyungsu Kim. "MARS: Model-agnostic Biased Object Removal without Additional Supervision for Weakly-Supervised Semantic Segmentation". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2023, pp. 614–623.

[28] Jiwoon Ahn and Suha Kwak. "Learning pixel-level semantic affinity with image-level supervision for weakly supervised semantic segmentation". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2018, pp. 4981–4990.

[29] Jiwoon Ahn, Sunghyun Cho, and Suha Kwak. "Weakly supervised learning of instance segmentation with inter-pixel relations". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2019, pp. 2209–2218.

[30] Amy Bearman et al. "What's the point: Semantic segmentation with point supervision". In: *Proceedings of the European Conference on Computer Vision*. 2016, pp. 549–565.

[31] R. Austin McEver and B. S. Manjunath. "PCAMs: Weakly supervised semantic segmentation using point supervision". In: *arXiv preprint arXiv:2007.05615* (2020).

[32] Paul Vernaza and Manmohan Chandraker. "Learning random-walk label propagation for weakly-supervised semantic segmentation". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2017, pp. 7158–7166.

[33] Di Lin et al. "Scribblesup: Scribble-supervised convolutional networks for semantic segmentation". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2016, pp. 3159–3167.

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

[35] Youngmin Oh, Beomjun Kim, and Bumsub Ham. "Background-aware pooling and noise-aware loss for weakly-supervised semantic segmentation". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2021, pp. 6913–6922.

[36] Junsong Fan et al. "Learning integral objects with intra-class discriminator for weakly-supervised semantic segmentation". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2020, pp. 4283–4292.

[37] Kunhao Yuan et al. "A multi-strategy contrastive learning framework for weakly supervised semantic segmentation". In: *Pattern Recognition* 137 (2023), p. 109298.

[38] Yunchao Wei et al. "Object region mining with adversarial erasing: A simple classification to semantic segmentation approach". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2017, pp. 1568–1576.

[39] Beomyoung Kim, Sangeun Han, and Junmo Kim. "Discriminative region suppression for weakly-supervised semantic segmentation". In: *The AAAI Conference on Artificial Intelligence*. Vol. 35. 2021, pp. 1754–1761.

[40] Jungbeom Lee et al. "Reducing information bottleneck for weakly supervised semantic segmentation". In: *Advances in Neural Information Processing Systems*. Vol. 34. 2021, pp. 27408–27421.

[41] Zilong Huang et al. "Weakly-supervised semantic segmentation network with deep seeded region growing". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2018, pp. 7014–7023.

[42] Yanwen Chong et al. "Erase then grow: Generating correct class activation maps for weakly-supervised semantic segmentation". In: *Neurocomputing* 453 (2021), pp. 97–108.

[43] Donald Olding Hebb. *The organization of behavior: A neuropsychological theory*. Psychology press, 2005.

[44] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. "Multilayer feedforward networks are universal approximators". In: *Neural Networks* 2.5 (1989), pp. 359–366.

[45] Phil Picton. *Introduction to Neural Networks*. Springer, 1994.

[46] Zhengyou Zhang. "Feature-based Facial Expression Recognition: Experiments with a Multi-layer Perceptron". PhD thesis. INRIA, 1998.

[47] Yann LeCun et al. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[48] Dan Claudiu Cireşan et al. "Deep, big, simple neural nets for handwritten digit recognition". In: *Neural computation* 22.12 (2010), pp. 3207–3220.

[49] George Cybenko. "Approximation by superpositions of a sigmoidal function". In: *Mathematics of control, signals and systems* 2.4 (1989), pp. 303–314.

[50] Barry L. Kalman and Stan C. Kwasny. "Why tanh: choosing a sigmoidal function". In: *Proceedings of the International Joint Conference on Neural Networks*. Vol. 4. 1992, pp. 578–581.

[51] Peng-Tao Jiang et al. "Layercam: Exploring hierarchical class activation maps for localization". In: *IEEE Transactions on Image Processing* 30 (2021), pp. 5875–5888.

[52] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. "Deep sparse rectifier neural networks". In: *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings. 2011, pp. 315–323.

[53] Bolei Zhou et al. "Learning deep features for discriminative localization". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2016, pp. 2921–2929.

[54] John S Bridle. "Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition". In: *Neurocomputing: Algorithms, architectures and applications*. Springer, 1990, pp. 227–236.

[55] Sebastian Ruder. "An overview of gradient descent optimization algorithms". In: *arXiv preprint arXiv:1609.04747* (2016).

[56] Juan Terven et al. "Loss functions and metrics in deep learning". In: *arXiv preprint arXiv:2307.02694* (2023).

[57] Erich L Lehmann and George Casella. *Theory of point estimation*. Springer Science & Business Media, 2006.

[58] Rod Goodman, John W Miller, and Padhraic Smyth. "Objective functions for neural network classifier design". In: *Proceedings of the IEEE International Symposium on Information Theory*. 1991, pp. 87–87.

[59] Elizbar A Nadaraya. "On estimating regression". In: *Theory of Probability & Its Applications* 9.1 (1964), pp. 141–142.

[60] Amanda Clare and Ross D King. "Knowledge discovery in multi-label phenotype data". In: *European conference on principles of data mining and knowledge discovery*. Springer. 2001, pp. 42–53.

[61] Herbert Robbins and Sutton Monro. "A stochastic approximation method". In: *The Annals of Mathematical Statistics* (1951), pp. 400–407.

[62] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[63] Peng Gao et al. "Learning reinforced attentional representation for end-to-end visual tracking". In: *Information Sciences* 517 (2020), pp. 52–67.

[64] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in Neural Information Processing Systems* 25 (2012).

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

[66] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2016, pp. 770–778.

[67] Michael Nielsen. *Neural networks and deep learning*. Vol. 25. Determination press San Francisco, CA, USA, 2015.

[68] Alexander Kolesnikov and Christoph H Lampert. "Seed, expand and constrain: Three principles for weakly-supervised image segmentation". In: *Proceedings of the European Conference on Computer Vision*. 2016, pp. 695–711.

[69] Jia Deng et al. "Imagenet: A large-scale hierarchical image database". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2009, pp. 248–255.

[70] Zifeng Wu, Chunhua Shen, and Anton Van Den Hengel. "Wider or deeper: Revisiting the resnet model for visual recognition". In: *Pattern Recognition* 90 (2019), pp. 119–133.

[71] Floris van Beers et al. “Deep neural networks with intersection over union loss for binary image segmentation”. In: *Proceedings of the International Conference on Pattern Recognition Applications and Methods*. 2019, pp. 438–445.

[72] Chien-Hao Tseng et al. “Person retrieval in video surveillance using deep learning–based instance segmentation”. In: *Journal of Sensors* 2021 (2021), pp. 1–12.

[73] Hassan Abu Alhaija et al. “Augmented reality meets deep learning for car instance segmentation in urban scenes”. In: *British Machine Vision Conference*. Vol. 1. 2017.

[74] Wenchao Gu, Shuang Bai, and Lingxing Kong. “A review on 2D instance segmentation based on deep neural networks”. In: *Image and Vision Computing* 120 (2022), p. 104401.

[75] Pedro F Felzenszwalb and Daniel P Huttenlocher. “Efficient graph-based image segmentation”. In: *International Journal of Computer Vision* 59 (2004), pp. 167–181.

[76] P Daniel Ratna Raju and G Neelima. “Image segmentation by using histogram thresholding”. In: *International Journal of Computer Science Engineering and Technology* 2.1 (2012), pp. 776–779.

[77] Guy Barrett Coleman and Harry C Andrews. “Image segmentation by clustering”. In: *Proceedings of the IEEE* 67.5 (1979), pp. 773–785.

[78] David G Lowe. “Object recognition from local scale-invariant features”. In: *Proceedings of the IEEE International Conference on Computer Vision*. Vol. 2. 1999, pp. 1150–1157.

[79] Navneet Dalal and Bill Triggs. “Histograms of oriented gradients for human detection”. In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. Vol. 1. 2005, pp. 886–893.

[80] K Bhargavi and S Jyothi. “A survey on threshold based segmentation technique in image processing”. In: *International Journal of Innovative Research and Development* 3.12 (2014), pp. 234–239.

[81] Todd R Reed and JM Hans Dubuf. “A review of recent texture segmentation and feature extraction techniques”. In: *CVGIP: Image understanding* 57.3 (1993), pp. 359–372.

[82] Alberto Garcia-Garcia et al. “A review on deep learning techniques applied to semantic segmentation”. In: *arXiv preprint arXiv:1704.06857* (2017).

[83] Mucong Ye et al. “Enhanced feature pyramid network for semantic segmentation”. In: *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE. 2021, pp. 3209–3216.

[84] Marco Toldo et al. “Unsupervised domain adaptation in semantic segmentation: a review”. In: *Technologies* 8.2 (2020), p. 35.

[85] Kehan Li et al. “ACSeg: Adaptive Conceptualization for Unsupervised Semantic Segmentation”. In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2023, pp. 7162–7172.

[86] Jang Hyun Cho et al. "Picie: Unsupervised semantic segmentation using invariance and equivariance in clustering". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2021, pp. 16794–16804.

[87] Mark Hamilton et al. "Unsupervised semantic segmentation by distilling feature correspondences". In: *arXiv preprint arXiv:2203.08414* (2022).

[88] Robert Harb and Patrick Knöbelreiter. "Infoseg: Unsupervised semantic image segmentation with mutual information maximization". In: *DAGM German Conference on Pattern Recognition*. Springer. 2021, pp. 18–32.

[89] Alexey Dosovitskiy et al. "An image is worth 16x16 words: Transformers for image recognition at scale". In: *arXiv preprint arXiv:2010.11929* (2020).

[90] Mathilde Caron et al. "Emerging properties in self-supervised vision transformers". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2021, pp. 9650–9660.

[91] James MacQueen et al. "Some methods for classification and analysis of multivariate observations". In: *Proceedings of the Berkeley Symposium on Mathematical Statistics and Probability*. Vol. 1. 1967, pp. 281–297.

[92] Kamal Taha. "Semi-supervised and un-supervised clustering: A review and experimental evaluation". In: *Information Systems* 114 (2023), p. 102178.

[93] Peixia Li et al. "Semi-supervised semantic segmentation under label noise via diverse learning groups". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2023, pp. 1229–1238.

[94] Shuo Li et al. "CFCG: Semi-Supervised Semantic Segmentation via Cross-Fusion and Contour Guidance Supervision". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2023, pp. 16348–16358.

[95] Chen Liang et al. "Logic-induced diagnostic reasoning for semi-supervised semantic segmentation". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2023, pp. 16197–16208.

[96] Jie Ma et al. "Enhanced soft label for semi-supervised semantic segmentation". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2023, pp. 1185–1195.

[97] Lihe Yang et al. "Revisiting weak-to-strong consistency in semi-supervised semantic segmentation". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2023, pp. 7236–7246.

[98] Haonan Wang et al. "AllSpark: Reborn Labeled Features from Unlabeled in Transformer for Semi-Supervised Semantic Segmentation". In: *arXiv preprint arXiv:2403.01818* (2024).

[99] Rihuan Ke et al. "A three-stage self-training framework for semi-supervised semantic segmentation". In: *IEEE Transactions on Image Processing* 31 (2022), pp. 1805–1815.

[100] Ashish Vaswani et al. "Attention is all you need". In: *Advances in Neural Information Processing Systems* 30 (2017).

[101] Tsung-Yi Lin et al. "Microsoft coco: Common objects in context". In: *Proceedings of the European Conference on Computer Vision*. 2014, pp. 740–755.

[102] Catherine Wah et al. *The caltech-ucsd birds-200-2011 dataset*. Tech. rep. CNS-TR-2011-001. California Institute of Technology, 2011.

[103] Yu-Ting Chang et al. "Weakly-supervised semantic segmentation via sub-category exploration". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2020, pp. 8991–9000.

[104] Yang Li et al. "Weakly supervised semantic segmentation by iterative superpixel-CRF refinement with initial clues guiding". In: *Neurocomputing* 391 (2020), pp. 25–41.

[105] Tianyi Zhang et al. "Splitting vs. merging: Mining object regions with discrepancy and intersection loss for weakly supervised semantic segmentation". In: *Proceedings of the European Conference on Computer Vision*. 2020, pp. 663–679.

[106] Xiaolin Zhang et al. "Adversarial complementary learning for weakly supervised object localization". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2018, pp. 1325–1334.

[107] Qi Chen et al. "Self-supervised image-specific prototype exploration for weakly supervised semantic segmentation". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2022, pp. 4288–4298.

[108] Sanghyun Jo and In-Jae Yu. "Puzzle-cam: Improved localization via matching partial and full features". In: *Proceedings of the IEEE International Conference on Image Processing*. IEEE. 2021, pp. 639–643.

[109] Weixuan Sun, Jing Zhang, and Nick Barnes. "Inferring the class conditional response map for weakly supervised semantic segmentation". In: *Proceedings of the IEEE Winter conference on Applications of Computer Vision*. 2022, pp. 2878–2887.

[110] Jungbeom Lee et al. "Ficklenet: Weakly and semi-supervised semantic image segmentation using stochastic inference". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2019, pp. 5267–5276.

[111] Dong Zhang et al. "Causal intervention for weakly-supervised semantic segmentation". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 655–666.

[112] Tao Chen et al. "Spatial Structure Constraints for Weakly Supervised Semantic Segmentation". In: *IEEE Transactions on Image Processing* (2024).

[113] Jing Li, Junsong Fan, and Zhaoxiang Zhang. "Towards noiseless object contours for weakly supervised semantic segmentation". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2022, pp. 16856–16865.

[114] Jungbeom Lee et al. "Anti-adversarially manipulated attributions for weakly supervised semantic segmentation and object localization". In: *IEEE transactions on Pattern Analysis and Machine Intelligence* (2022).

[115] Yi Li et al. "Uncertainty estimation via response scaling for pseudo-mask noise mitigation in weakly-supervised semantic segmentation". In: *The AAAI Conference on Artificial Intelligence*. Vol. 36. 2022, pp. 1447–1455.

[116] Philipp Krähenbühl and Vladlen Koltun. "Efficient inference in fully connected crfs with gaussian edge potentials". In: *Advances in Neural Information Processing Systems*. Vol. 24. 2011.

[117] Ying-Hwey Nai et al. "Comparison of metrics for the evaluation of medical segmentations using prostate MRI dataset". In: *Computers in Biology and Medicine* 134 (2021), p. 104497.

[118] Jinwang Wang et al. "A normalized Gaussian Wasserstein distance for tiny object detection". In: *arXiv preprint arXiv:2110.13389* (2021).

[119] Yude Wang et al. "Self-supervised equivariant attention mechanism for weakly supervised semantic segmentation". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2020, pp. 12275–12284.

[120] Liyi Chen et al. "Weakly supervised semantic segmentation with boundary exploration". In: *Proceedings of the European Conference on Computer Vision*. 2020, pp. 347–362.

[121] Jinlong Li et al. "Expansion and shrinkage of localization for weakly-supervised semantic segmentation". In: *Advances in Neural Information Processing Systems*. Vol. 35. 2022, pp. 16037–16051.

[122] Minhyun Lee, Dongseob Kim, and Hyunjung Shim. "Threshold matters in wsss: Manipulating the activation for the robust and accurate segmentation model against thresholds". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2022, pp. 4330–4339.

[123] Junsong Fan, Zhaoxiang Zhang, and Tieniu Tan. "Employing multi-estimations for weakly-supervised semantic segmentation". In: *Proceedings of the European Conference on Computer Vision*. 2020, pp. 332–348.

[124] Yunchao Wei et al. "Revisiting dilated convolution: A simple approach for weakly-and semi-supervised semantic segmentation". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2018, pp. 7268–7277.

[125] Yujie Li, Jiaxing Sun, and Yun Li. "Weakly-Supervised Semantic Segmentation Network With Iterative dCRF". In: *IEEE Transactions on Intelligent Transportation Systems* 23.12 (2022), pp. 25419–25426.

[126] Xiang Wang et al. "Weakly-supervised semantic segmentation by iterative affinity learning". In: *International Journal of Computer Vision* 128 (2020), pp. 1736–1749.

[127] Wataru Shimoda and Keiji Yanai. "Self-supervised difference detection for weakly-supervised semantic segmentation". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 5208–5217.

[128] Sanghyun Jo et al. "DHR: Dual Features-Driven Hierarchical Rebalancing in Inter-and Intra-Class Regions for Weakly-Supervised Semantic Segmentation". In: *arXiv preprint arXiv:2404.00380* (2024).

[129] Suha Kwak, Seunghoon Hong, and Bohyung Han. "Weakly supervised semantic segmentation using superpixel pooling network". In: *Proceedings of the AAAI conference on Artificial Intelligence*. Vol. 31. 2017.

[130] Tong Wu et al. "Embedded discriminative attention mechanism for weakly supervised semantic segmentation". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2021, pp. 16765–16774.

[131] Seungho Lee et al. "Railroad is not a train: Saliency as pseudo-pixel supervision for weakly supervised semantic segmentation". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2021, pp. 5495–5505.

[132] Lian Xu et al. "Leveraging auxiliary tasks with affinity learning for weakly supervised semantic segmentation". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2021, pp. 6984–6993.

[133] Sheng Yi et al. "Weakly-supervised semantic segmentation with superpixel guided local and global consistency". In: *Pattern Recognition* 124 (2022), p. 108504.

[134] Yang Li et al. "Weakly supervised semantic segmentation by iterative superpixel-CRF refinement with initial clues guiding". In: *Neurocomputing* 391 (2020), pp. 25–41.

[135] Ting Zhao and Xiangqian Wu. "Pyramid feature attention network for saliency detection". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2019, pp. 3085–3094.

[136] Radhakrishna Achanta et al. "SLIC superpixels compared to state-of-the-art superpixel methods". In: *IEEE transactions on Pattern Analysis and Machine Intelligence* 34.11 (2012), pp. 2274–2282.

[137] Yuzhu Ji et al. "CNN-based encoder-decoder networks for salient object detection: A comprehensive review and recent advances". In: *Information Sciences* 546 (2021), pp. 835–857.

[138] Guangshuai Gao et al. "Co-saliency detection with co-attention fully convolutional network". In: *IEEE Transactions on Circuits and Systems for Video Technology* 31.3 (2020), pp. 877–889.

[139] Tanveer Hussain et al. "Pyramidal attention for saliency detection". In: *2022 IEEE conference on Computer Vision and Pattern Recognition Workshops*. IEEE. 2022, pp. 2877–2887.

[140] Jing Zhang et al. "Deep unsupervised saliency detection: A multiple noisy labeling perspective". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2018, pp. 9029–9038.

[141] Jing Zhang et al. "Deep unsupervised saliency detection: A multiple noisy labeling perspective". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2018, pp. 9029–9038.

[142] Gyungin Shin, Samuel Albanie, and Weidi Xie. "Unsupervised salient object detection with spectral cluster voting". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2022, pp. 3971–3980.

[143] Kunpeng Li et al. "Tell me where to look: Guided attention inference network". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2018, pp. 9215–9223.

[144] Qibin Hou et al. "Self-erasing network for integral object attention". In: *Advances in Neural Information Processing Systems*. Vol. 31. 2018.

[145] Lixiang Ru et al. "Learning Affinity from Attention: End-to-end Weakly-supervised Semantic Segmentation with Transformers". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2022, pp. 16846–16855.

[146] Venkatarama Krishnan. *Probability and Random Processes*. John Wiley & Sons, 2015.

[147] Bharath Hariharan et al. "Semantic contours from inverse detectors". In: *International Conference on Computer Vision*. IEEE. 2011, pp. 991–998.

[148] Peng-Tao Jiang et al. "Integral object mining via online attention accumulation". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 2070–2079.

[149] Jiang-Jiang Liu et al. "A simple pooling-based design for real-time salient object detection". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2019, pp. 3917–3926.

[150] Sheng Liu et al. "Adaptive early-learning correction for segmentation from noisy annotations". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2022, pp. 2606–2616.

[151] Ruiwen Li et al. "TransCAM: Transformer attention-based cam refinement for weakly supervised semantic segmentation". In: *Journal of Visual Communication and Image Representation* 92 (2023), p. 103800.

[152] Dingwen Zhang et al. "Weakly supervised semantic segmentation via alternative self-dual teaching". In: *arXiv preprint arXiv:2112.09459* (2021).

[153] Tsung-Wei Ke, Jyh-Jing Hwang, and Stella X Yu. "Universal weakly supervised segmentation by pixel-to-segment contrastive learning". In: *arXiv preprint arXiv:2105.00957* (2021).

[154] Jiapei Feng et al. "Weakly-supervised semantic segmentation via online pseudo-mask correcting". In: *Pattern Recognition Letters* 165 (2023), pp. 33–38.

[155] Tianfei Zhou et al. "Regional semantic contrast and aggregation for weakly supervised semantic segmentation". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2022, pp. 4299–4309.

[156] Ye Du et al. "Weakly supervised semantic segmentation by pixel-to-prototype contrast". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2022, pp. 4320–4329.

[157] Jinheng Xie et al. "C2am: Contrastive learning of class-agnostic activation map for weakly supervised object localization and semantic segmentation". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2022, pp. 989–998.

[158] Daniel Morales-Brotons, Thijs Vogels, and Hadrien Hendrikx. "Exponential moving average of weights in deep learning: Dynamics and benefits". In: *Transactions on Machine Learning Research* (2024).

[159] Zhaozheng Chen et al. "Class re-activation maps for weakly-supervised semantic segmentation". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2022, pp. 969–978.

[160] Jinlong Li et al. "Weakly supervised semantic segmentation via progressive patch learning". In: *IEEE Transactions on Multimedia* (2022).

[161] Jie Qin et al. "Activation modulation and recalibration scheme for weakly supervised semantic segmentation". In: *The AAAI Conference on Artificial Intelligence*. Vol. 36. 2022, pp. 2117–2125.

[162] Jungbeom Lee et al. "Weakly supervised semantic segmentation using out-of-distribution data". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2022, pp. 16897–16906.

[163] Meijie Zhang, Jianwu Li, and Tianfei Zhou. "Multi-granular semantic mining for weakly supervised semantic segmentation". In: *Proceedings of the 30th ACM International Conference on Multimedia*. 2022, pp. 6019–6028.

[164] Weixuan Sun et al. "An Alternative to WSSS? An Empirical Study of the Segment Anything Model (SAM) on Weakly-Supervised Semantic Segmentation Problems". In: *arXiv preprint arXiv:2305.01586* (2023).

[165] Shenghai Rong et al. "Boundary-Enhanced Co-Training for Weakly Supervised Semantic Segmentation". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2023, pp. 19574–19584.

[166] Junwen Pan et al. "Learning self-supervised low-rank network for single-stage weakly and semi-supervised semantic segmentation". In: *Proceedings of the IEEE International Conference on Computer Vision* 130.5 (2022), pp. 1181–1195.

[167] Hugo Touvron et al. "Training data-efficient image transformers & distillation through attention". In: *International conference on machine learning*. PMLR. 2021, pp. 10347–10357.

[168] Enze Xie et al. "SegFormer: Simple and efficient design for semantic segmentation with transformers". In: *Advances in neural information processing systems* 34 (2021), pp. 12077–12090.

[169] Fangwen Wu et al. "Masked Collaborative Contrast for Weakly Supervised Semantic Segmentation". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2024, pp. 862–871.

[170] Lianghui Zhu et al. "Weaktr: Exploring plain vision transformer for weakly-supervised semantic segmentation". In: *arXiv preprint arXiv:2304.01184* (2023).

[171] Zelin Peng et al. "Usage: A unified seed area generation paradigm for weakly supervised semantic segmentation". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 624–634.

[172] Zesen Cheng et al. “Out-of-candidate rectification for weakly supervised semantic segmentation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 23673–23684.

[173] Simone Rossetti et al. “Max pooling with vision transformers reconciles class and shape in weakly supervised semantic segmentation”. In: *Proceedings of the European Conference on Computer Vision*. Springer. 2022, pp. 446–463.

[174] Jingdong Wang et al. “Deep high-resolution representation learning for visual recognition”. In: *IEEE transactions on Pattern Analysis and Machine Intelligence* 43.10 (2020), pp. 3349–3364.

Zhengyang LYU

Doctorat : Optimisation et Sûreté des Systèmes

Année 2024

Amélioration des pseudo-masques pour la segmentation sémantique faiblement supervisée

La segmentation sémantique vise à attribuer à chaque pixel d'une image une classe donnée. Pour éviter la coûteuse annotation pixel par pixel nécessaire à l'approche entièrement supervisée, la segmentation sémantique faiblement supervisée a été proposée. Dans ce travail, seule les étiquettes des objets présents dans l'image sont supposées connues. Le point clé est de générer des pseudo-masques précis comme vérité terrain pour chaque image afin d'entraîner un modèle de segmentation. Bien que de nombreux travaux aient été menés dans ce domaine, les pseudo-masques comportent toujours des erreurs. L'objectif de cette thèse est donc d'améliorer leur précision. La génération des pseudo-masques implique deux étapes : la détermination de cartes d'activation de classe (CAM pour *class activation maps*) à partir d'un réseau de classification, puis la génération des pseudo-masques à partir des CAM. Cette thèse propose de nouvelles méthodes pour chaque étape. Premièrement, pour améliorer le CAM, nous introduisons des contraintes supplémentaires dans la fonction coût pour l'apprentissage du réseau de classification. Deuxièmement, pour améliorer les pseudo-masques nous proposons une méthode de classification des pixels spécifique à chaque image basée sur différentes caractéristiques, ce qui permet d'atteindre un bon équilibre entre résolution et sémantique. Les expériences montrent que les pseudo-masques ainsi obtenus comportent moins d'erreurs et permettent d'apprendre un meilleur modèle de segmentation.

Mots clés : vision par ordinateur – apprentissage faiblement supervisé – apprentissage profond – apprentissage automatique – réseaux neuronaux (informatique) – machines à vecteurs de support.

Improving the Pseudo-masks for Weakly Supervised Semantic Segmentation

Semantic segmentation aims at assigning each pixel of an image to a given class. To avoid expensive pixel-wise annotations when dealing with fully supervised approach, Weakly Supervised Semantic Segmentation (WSSS) has been proposed. In this work, we consider the approach with image-level labels. The key point is to generate accurate pseudo-masks as the target labels for training a segmentation model. Although lots of work has been studied in this field, the pseudo-masks always deviate from the ground truth. Thus, the aim of this thesis is to improve the accuracy of the pseudo-masks. The generation of the pseudo-masks involves two steps, which are generating the Class Activation Maps (CAM) from a classification network, and then obtaining pseudo-masks from the CAM. New methods are proposed for each step. Firstly, we introduce additional objectives within the training of the classification network for generating the enhanced CAM. Secondly, we propose a multi-feature image-specific pixel classification approach to obtain accurate pseudo-masks, which achieves a good balance between resolution and semantics. Experiments show that the resulting pseudo-masks are enhanced and ultimately improve segmentation results as part of the overall WSSS pipeline.

Keywords: computer vision – weakly supervised learning – deep learning (machine learning) – machine learning – neural networks (computer science) – support vector machines.

Thèse réalisée en partenariat entre :





Ecole Doctorale "Sciences pour l'Ingénieur"