Dense prediction and deep learning in complex visual scenes

Wang, Yi

2021

Wang, Y. (2021). Dense prediction and deep learning in complex visual scenes. Doctoral thesis, Nanyang Technological University, Singapore. <https://hdl.handle.net/10356/152009>

<https://hdl.handle.net/10356/152009>

<https://doi.org/10.32657/10356/152009>

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License (CC BY-NC 4.0).

Downloaded on 26 Apr 2023 18:19:13 SGT

Dense Prediction and Deep Learning in Complex Visual Scenes

Wang Yi

School of Electrical & Electronic Engineering

2021

Dense Prediction and Deep Learning in Complex Visual Scenes

Wang Yi

School of Electrical & Electronic Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

2021

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

30/06/2021

.....

Date

Wang Yi

.....

Wang Yi

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

30/06/2021

.....

Date

.....

Lap-Pui Chau

Authorship Attribution Statement

This thesis contains material from 3 papers published and 1 paper accepted for publication in the following peer-reviewed journals, and 2 papers presented in conferences in which I am the first author.

Chapter 3 is published as:

[a] Yi Wang, Hui Liu and Lap-Pui Chau, “Single underwater image restoration using adaptive attenuation-curve prior,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, 65(3): 992-1002, 2018.

[b] Yi Wang, Lap-Pui Chau, and Xiaoxi Ma, “Airlight estimation based on distant region segmentation,” in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1-5, 2019.

The contributions of the co-authors are as follows:

- I proposed the initial methods, designed the experiments, and prepared the manuscript drafts.
- Hui Liu and I collected underwater images and conducted the experiments of underwater image restoration.
- The methods were improved by discussions with Lap-Pui Chau and Hui Liu.
- Lap-Pui Chau, Hui Liu, Xiaoxi Ma, and I revised the manuscripts.

Chapter 4 is published as:

[c] Yi Wang, Junhui Hou, and Lap-Pui Chau, “Object counting in video surveillance using multi-scale density map regression,” in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2422-2426, 2019.

[d] Yi Wang, Junhui Hou, Xinyu Hou, and Lap-Pui Chau, “A self-training approach for point-supervised object detection and counting in crowds,” *IEEE Transactions on Image Processing*, 30: 2876-2887, 2021.

The contributions of the co-authors are as follows:

- I proposed the initial methods, designed the experiments, and prepared the manuscript drafts.
- Xinyu Hou and I conducted the experiments of crowd detection methods.
- Junhui Hou, Lap-Pui Chau, and I discussed and improved the methods.
- Junhui Hou, Lap-Pui Chau, Xinyu Hou, and I revised the manuscripts.

Chapter 5 is published as:

[e] Yi Wang, Zhen-Peng Bian, Junhui Hou, and Lap-Pui Chau, “Convolutional neural networks with dynamic regularization,” *IEEE Transactions on Neural Networks and Learning Systems*, 32(5): 2299-2304, 2021.

and accepted for publication as:

[f] Yi Wang, Zhen-Peng Bian, Yunhao Zhou, and Lap-Pui Chau, “Rethinking and designing a high-performing automatic license plate recognition approach,” *IEEE Transactions on Intelligent Transport System*, 2021.

The contributions of the co-authors are as follows:

- Zhen-Peng Bian created the initial frameworks.
- I improved the methods, designed the experiments, and prepared the manuscript drafts.
- I conducted the experiments of both dynamic regularization and license plate recognition.
- Junhui Hou, Zhen-Peng Bian, Lap-Pui Chau, Yunhao Zhou, and I discussed and revised the manuscripts.

30/06/2021

.....

Date

Wang Yi

.....

Wang Yi

Acknowledgements

I would like to take this opportunity to thank all the people who have cared for and helped me during my study and life at NTU.

First, I wish to express my sincere gratitude to my supervisor, Dr. Chau Lap-Pui, who gives me a significant guide and help during my Ph.D. studies. His professional attitude and open mind bring me great inspiration and let me know many research methods. Meanwhile, working as a research associate, I need to thank him for giving me such creative and challenging research works, which brings me meaningful work experiences.

Second, I am grateful to my lovely friends for their kindness, appreciation, support, and even criticism. Without them, there would be no progress for me. I sincerely thank Dr. Hou Junhui, Dr. Bian Zhen-peng, Ms. Liu Hui, Ms. Hou Xinyu, Ms. Ma Xiaoxi, and Mr. Zhou Yunhao for their supports and cooperation in my Ph.D. research. It is my great privilege to work with Ms. Liu Hui, Dr. Yan Yongsheng, Dr. Yang Ming, Dr. Tan Cheen Hau, Dr. Alexander Matyasko, who made our research projects going smoothly. I also appreciate my Master's supervisors, Dr. Wan Shuai and Dr. Mei Shaohui, who recommended and supported me to study for doctorate. Besides, I want to thanks Dr. Chen Jie, Dr. Gong Yanchao, Dr. Liu Qinglai, Mr. Zhuang Huiping, Ms. Ni Yun, Ms. Nan Shengyu, Ms. Tang Lisha, and other talented friends for making my Ph.D. journey unforgettable. Especially, thank Ms. Yao Hangyue for the companion and care.

In addition, I would like to thank my colleagues at the Centre for Information Sciences and Systems (CISS), Smart Mobility Experience Lab (SMEL), Rapid-Rich Object Search (ROSE) Lab, and ST Engineering-NTU Corporate Lab.

Last but not least, I would like to express my deep gratitude to my beloved mother Pan Tuye, father Wang Kui, Grandmother Long Yuanxiu, and other family members. With their understanding and support, I can made it this far.

“Imagination is more important than knowledge.”

—Einstein, Albert

To my dear family

Summary

Many computer vision applications, such as video surveillance, autonomous driving, and crowd analysis, suffer from the challenging conditions of complex scenes, including haze, underwater, extreme lighting, and crowded and small objects. These scenes might degrade or compromise the performance of or even fail computer vision algorithms. It is valuable and important to develop methods to address such complex visual scenes. In this thesis, we follow a unified thinking for a series of dense prediction problems from low-level vision to high-level vision, i.e., restoration, detection, and recognition.

In the restoration problem, haze and underwater scenes degrade the contrast and color of images due to light scattering and absorption. Research on de-scattering or dehazing refers to restore images captured in such scenes. As the first research direction of this thesis, we propose a novel image restoration approach for underwater imagery based on an adaptive attenuation-curve prior (AACP). The prior describes the fact that all pixel values of a clear image can be partitioned into several hundred distinct clusters in RGB space, and the pixel values in each cluster will be distributed on a curve with a power-function form after attenuated by water. Therefore, the pixel-wise medium transmission can be predicted according to the pixel value's distribution on such a curve. This method is generalizable and can be extended to hazy images. Moreover, according to the fact that ambient light exists in the infinite distant region of an outdoor image, we propose a new deep learning-based framework to estimate the ambient light by distant region segmentation (DRS). Qualitative and quantitative results show that the proposed methods achieve superior performance in comparison with state-of-the-art methods.

In the detection problem, crowded objects present large scale variation and severe occlusion, posing great challenges to object detectors. In addition, current crowd datasets only provide coarse point-level annotations, i.e., human heads are labeled as points, so state-of-the-art object detectors cannot be trivially applied to such point supervision. In our second research direction, we propose a novel self-training

approach that enables a typical object detector trained only with point-level annotations to densely predict center points and sizes of crowded objects, termed Crowd-DCNet. Specifically, we propose the locally-uniform distribution assumption (LUDA) for initializing pseudo object sizes from point-level supervisory information, the crowdedness-aware loss for regressing object sizes, and the confidence and order-aware refinement scheme for refining the pseudo object sizes continuously during training. With our self-training approach, the ability of the detector is increasingly boosted. Moreover, bypassing object detection, we introduce a compact convolutional neural network (CNN) for object counting in video surveillance, in which a multi-scale density (MSD) regressor is employed to predict the coarse- and fine-scale density maps. The comprehensive experimental results on six challenging benchmark datasets show that our approach significantly outperforms state-of-the-art methods under both detection and counting tasks.

In the recognition problem, small objects in unconstrained scenes adversely affect the accuracy of automatic recognition systems. Our third research direction focuses on automatic license plate recognition (ALPR) in unconstrained environments, such as oblique views, uneven illumination, and various weather conditions. Our study produces an outstanding design of ALPR with four insights: (1) the resampling-based cascaded framework is beneficial to both speed and accuracy; (2) the highly efficient license plate recognition should abandon additional character segmentation and recurrent neural network (RNN), but adopt a plain CNN; (3) in the case of CNN, taking advantage of vertex information on license plates improves recognition performance; and (4) the weight-sharing character classifier addresses the lack of training images in small-scale datasets. Based on these insights, we propose a real-time and high-performing ALPR approach, termed VSNet. The vertex supervisory information is fully exploited for training a detector (VertexNet) to predict the geometric shapes of license plates such that license plates can be rectified and their characters can be densely predicted by a recognizer (SCR-Net). Moreover, we propose a dynamic regularization method to avoid overfitting and improve the generalization ability of CNN. Experimental results on two challenging benchmark datasets demonstrate the effectiveness of the proposed method.

Contents

Acknowledgements	ix
Summary	xiii
List of Figures	xix
List of Tables	xxv
List of Acronyms	xxix
1 Introduction	1
1.1 Background	1
1.2 Scope of Dense Prediction	3
1.3 Challenge and Motivation	5
1.3.1 Light Scattering Environments	5
1.3.2 Crowded Objects and Small Objects	6
1.3.3 Lack of Data Annotations	6
1.4 Major Contributions	7
1.5 Organization of the Thesis	8
2 Literature Review	9
2.1 Related Work	9
2.1.1 Dense Prediction with Prior Knowledge	9
2.1.1.1 Atmospheric Scattering Model	10
2.1.1.2 Image Restoration Methods by Prior Knowledge	11
2.1.2 Dense Prediction with Convolutional Neural Networks	13
2.1.3 CNN-based Dense Regression	15
2.1.3.1 Image Restoration	16
2.1.3.2 Crowd Counting	17
2.1.4 CNN-based Dense Classification	18
2.1.4.1 Crowd Localization and Detection	18
2.1.4.2 License Plate Detection and Recognition	19
2.2 Benchmark Dataset	20
2.2.1 Image Restoration Dataset	21

2.2.2	Crowd Detection and Counting Dataset	22
2.2.3	License Plate Detection and Recognition Dataset	23
3	Image Restoration in Light Scattering Scenes	25
3.1	Introduction	26
3.2	Underwater Image Restoration with Adaptive Attenuation-curve Prior	29
3.2.1	Underwater Image Modeling and Adaptive Attenuation-curve Prior	29
3.2.1.1	Modeling of Underwater Images	29
3.2.1.2	Adaptive Attenuation-curve Prior	31
3.2.2	Underwater Image Restoration	33
3.2.2.1	Waterlight Estimation	34
3.2.2.2	Transmission Estimation	35
3.2.2.3	Scene Radiance Restoration	39
3.3	Ambient Light Estimation with Deep Learning	39
3.3.1	Airlight Estimation Framework	40
3.3.2	Training Data Preparation	42
3.4	Experiments	43
3.4.1	Evaluation of Underwater Image Restoration	43
3.4.1.1	Qualitative Assessment	44
3.4.1.2	Quantitative Assessment	46
3.4.1.3	Complexity Analysis	48
3.4.1.4	Other Applications in the Atmosphere	49
3.4.2	Evaluation of Ambient Light Estimation	49
3.4.2.1	Synthetic Images	50
3.4.2.2	Natural Images	51
3.4.2.3	Integration of Ambient Light Estimation into Underwater Image Restoration Method	52
3.5	Chapter Summary	52
4	Crowd Detection and Counting	55
4.1	Introduction	56
4.2	Crowd-DCNet for Crowd Detection and Counting	59
4.2.1	Crowd-DCNet	59
4.2.1.1	Architecture	59
4.2.1.2	Supervision information	61
4.2.1.3	Loss function	61
4.2.1.4	Inference	62
4.2.2	LUDA-based Pseudo Object Size Generation	62
4.2.3	Crowdedness-aware Loss for Object Size Regression	64
4.2.4	Confidence and Order-aware Refinement Scheme for Pseudo Size Updating	65
4.3	MSD Regressor for Crowd Counting	67
4.3.1	Architecture	67

4.3.1.1	Feature-Shared Network	67
4.3.1.2	Coarse-Scale Network	68
4.3.1.3	Fine-Scale Network	68
4.3.2	Loss Function	69
4.3.3	Ground Truth Generation	69
4.4	Experiments	70
4.4.1	Evaluation of Crowd-DCNet	70
4.4.1.1	Experimental Settings	71
4.4.1.2	Ablation Study	73
4.4.1.3	Comparison with State-of-the-art Methods	76
4.4.2	Evaluation of MSD Regressor	80
4.4.2.1	Comparison with State-of-the-art Methods	81
4.4.2.2	Ablation Study	83
4.5	Chapter Summary	83
5	License Plate Detection and Recognition	85
5.1	Introduction	86
5.2	Proposed VSNet for ALPR	89
5.2.1	VertexNet for LP Detection	89
5.2.1.1	Architecture of VertexNet	90
5.2.1.2	Training of VertexNet	94
5.2.2	SCR-Net for LP Recognition	94
5.2.2.1	License Plate Rectification	95
5.2.2.2	Architecture of SCR-Net	95
5.2.2.3	Training of SCR-Net	97
5.2.3	Dynamic Regularization	98
5.2.3.1	SCR-Net with Dynamic Regularization	98
5.2.3.2	Update of Regularization Strength	100
5.3	Experiments	102
5.3.1	Datasets and Evaluation Metrics	102
5.3.1.1	Dataset	102
5.3.1.2	Evaluation Metrics	103
5.3.2	Implementation Details	103
5.3.3	Evaluation of License Plate Detection	104
5.3.3.1	Comparison with State-of-the-art License Plate De- tection Methods	104
5.3.3.2	Ablation Study of VertexNet	106
5.3.4	Evaluation of License Plate Recognition	107
5.3.4.1	Comparison with State-of-the-art License Plate Recog- nition Methods	107
5.3.4.2	Ablation Study of SCR-Net	110
5.4	Chapter Summary	112
6	Conclusions and Future Work	113

6.1	Conclusions	113
6.2	Future Work	115
6.2.1	High-level Vision Tasks in Poor Weather	115
6.2.2	Learning from Less Labeled Data	115
6.2.3	Multi-task Learning for Dense Prediction	116
 List of Author's Publications		 119
Bibliography		121

List of Figures

1.1	Complex visual scenes. The images in (a), (b), (c) are obtained from [1–3].	2
1.2	Computer vision tasks for image understanding.	2
1.3	Typical dense prediction methods and the predicted structural maps. We will present them in detail in the following chapters. <i>Zoom in the figure for better viewing.</i>	4
2.1	Categories of dense prediction. The tasks are listed in the non-blue boxes. Among the tasks, object detection connects with dense classification and regression because it not only densely classify every bounding box but also regress the offset of the bounding boxes. The tasks in green boxes will be described in detail in this chapter and the following chapters accordingly.	10
2.2	Architecture of LeNet-5 for hand-written digit recognition [4] ©1998 IEEE.	14
2.3	Architecture of FCN for semantic segmentation [5] ©2015 IEEE.	15
2.4	Underwater images.	21
2.5	Synthetic hazy images and haze-free images. Left to right then top to bottom corresponds to Church, Couch, Flower1, Flower2, Lawn1, Lawn2, Mansion, Moebius, Raindeer, Road1, and Road2, respectively.	21
2.6	Example images with crowds or vehicles on (a) WiderFace [6], (b) SHA [7], (c) SHB [7], (d) NWPU-Crowd [2], and (e) CARPK and PUCPR+ [8]. The crowd locations are presented by the green boxes or points.	22
2.7	License plate images captured in unconstrained scenarios, e.g., (a) and (e) oblique view, (b) uneven illumination, and (c) snow in the nighttime, labeled by ground-truth (GT) bounding boxes, four vertices, and characters.	23
3.1	Image restoration and ambient light estimation methods.	25
3.2	Atmospheric scene of light propagation. Airlight mixes with scattered light along the line of sight to the camera.	26
3.3	Underwater scene of light propagation, airlight enters from air to the water, part of the light attenuates along the line of sight. The point x represents the nearest scene point to the camera.	27
3.4	Penetration of light with different wavelengths in the ocean.	27

3.5	Attenuation-curve prior to model the degradation process of a clear image in water. (a) Four clusters' pixels generated by k-means in a clear image. (b) Corresponding pixel values of the clusters in RGB space. (c) The degraded image by water with the same clusters as (a). (d) Corresponding degraded pixel values in RGB space, forming attenuation curves.	32
3.6	Flowchart of the AACCP-based underwater image restoration.	34
3.7	Finding attenuation-curves. (a) Power-function relationship between color channels. (b) Linear relationship. (c) Rotated coordinates from (b) for clustering.	36
3.8	Airlight estimation framework (DRS-based airlight estimation method). The numbers below U-Net indicate the number of the feature maps.	40
3.9	Synthesis of pair of the hazy image and binary mask. (a) ~ (f) corresponds to the clear image, airlight, hazy image, transmission map, superpixels of transmission map, and binary mask, respectively.	43
3.10	Sample testing images, from left to right and top to bottom, corresponding to "Image-1" to "Image-20", respectively.	44
3.11	Visual comparison of four underwater images with different scenes. (a) The original images. The restored results with the corresponding waterlight and transmission yielded by: (b) MIP [9], (c) UDCP [10], (d) BLA [11], (e) Our FACP [12], (f) Our AACCP.	45
3.12	Comparisons of visual results by different methods. Row 1 is the input images; row 2 is the results of MIP [9]; row 3 is the results of UDCP [10]; row 4 is the results of BLA [11]; row 5 is the results of our FACP [12]; row 6 is the results of our AACCP.	46
3.13	Comparison with the non-IFM image enhancement method. (a) The input images; (b) the results of ACE [13]; (c) the results of our AACCP. Row 1, 2, 3, and 4 correspond to Image-1, 5, 10, and 12, respectively.	47
3.14	Dehazing results of three images taken in the atmospheric environment. Row 1 shows input hazy images. Row 2 to 4 corresponds to the restored images by He's [14], Berman's [15], and the proposed method, respectively.	50
3.15	Examples of synthetic images. GT stands for ground-truth airlight. The airlight above the image is estimated by He <i>et al.</i> [14], Bahat and Irani [1], Berman <i>et al.</i> [16], and DRS (ours), respectively. The green regions are the output masks of U-Net by our method.	51
3.16	Examples of natural images. GT stands for ground-truth airlight. The airlight above the image is estimated by Zhang and Patel [17], He <i>et al.</i> [14], Bahat and Irani [1], Berman <i>et al.</i> [16], and DRS (ours), respectively. The green regions are the output masks of U-Net by our method.	52
3.17	Quantitative comparison between (b) our AACCP with DSR and (c) deep learning-based DCPDN [17]. (a) shows input hazy images.	53
4.1	Crowd detection and counting methods. <i>Zoom in for better viewing.</i>	55

4.2	Illustration of generated training examples at different phases by our self-training approach. The images are from the WiderFace [6] dataset. The green dots shown at the centers of faces stand for the point-level annotations. (a): Before training, the pseudo object sizes (red boxes) were generated by the proposed locally-uniform distribution assumption. The numbers shown at the top-left corner of red boxes stand for object crowdedness. (b): After training, the refined pseudo object sizes (blue boxes) by our crowdedness-aware loss and the confidence and order-aware refinement scheme. The numbers shown at the top-right corner of blue boxes stand for the pseudo object sizes' posterior probabilities. <i>Zoom in on the figure for better viewing.</i>	58
4.3	Illustration of the proposed self-training framework for training a detector (Crowd-DCNet) only with point-wise supervision. The decoder produces high-resolution feature maps for the estimation of objects' center points and sizes. Before training, the pseudo object sizes are first generated from the point-wise supervisory information, based on our locally-uniform distribution assumption. During training, the crowdedness-aware loss supervises the size regression between pseudo sizes and predicted sizes. Built by center points and sizes, the blue and red bounding boxes are the prediction and pseudo supervision, respectively. In every training iteration, our confidence and order-aware refinement scheme updates the red boxes with the corresponding blue boxes based on their confidence, as shown in the box's top-right corner. Here an image with sparse objects is used for clear visualization purposes.	60
4.4	Architecture of the proposed MSD regressor with a cascaded structure.	67
4.5	Generation of fine and coarse density map. The objects are marked as red points in the annotation map.	70
4.6	Visual comparison of the pseudo object sizes generated by GAK and LUDA in a crowded scene. For LUDA, the number on top of each bounding box refers to the object crowdedness. The high crowdedness value means the pseudo size is accurate.	74
4.7	Percentage of refined training examples with respect to the training epoch.	77
4.8	Qualitative results on (a) WiderFace [6], (b) SHA [7], (c) SHB [7], (d) PUCPR+ [8], and (e) CARPK [8]. The top row shows the ground-truth boxes or points, and counts. The bottom row shows the bounding boxes and counts predicted by our Crowd-DCNet. <i>Zoom in on the figure for better viewing.</i>	78

4.9	Comparisons between the bounding boxes (at top row) produced by our Crowd-DCNet and the density maps (at bottom row) produced by CSRNet [18] on the SHA and SHB [7] datasets. The predicted count is shown at the bottom-left corner of an image. The ground-truth counts from left to right images are 1068, 584, 139, and 274. <i>Zoom in on the figure for better viewing.</i>	80
4.10	Qualitative results from the UCSD (first two columns) and TRANCOS (last two columns) dataset. The ROI is inside the green lines. The first row shows the input images. The second row shows the corresponding ground truth density map and count. The third row shows the estimated density map and count of our MSD regressor.	82
5.1	Automatic license plate recognition method.	85
5.2	Images captured in unconstrained scenarios, e.g., (a) oblique view, (b) uneven illumination, and (c) snow in the nighttime, labeled by ground-truth (GT) bounding boxes, four vertices, and characters. Our AOLP approach can handle these cases and generate correct predictions (Pred).	86
5.3	Framework of the proposed VSNet. An input image is resized to a small resolution, i.e., 256×256 , for fast inference in VertexNet. The LP patch is then resampled from the finest input image and rectified to high resolution according to the predicted vertices by VertexNet. Finally, SCR-Net recognizes all characters in the LP.	87
5.4	Architecture of VSNet. (1)-(4) are executed sequentially in inference. H and W are set to 256 for fast inference. VertexNet consists of the backbone, fusion, and head networks, which predicts the bounding boxes and vertices of LPs. In the fusion structure, “ $2 \times U$ ” is the bilinear upsampling with a factor of 2. “BR” is the batch normalization followed by the ReLU. “CBR” is the 1×1 -s1 convolution followed by the “BR”. $[\cdot]$ is the concatenation operation. SCR-Net resamples and rectifies LP images from the finest input images based on the predicted vertices (red points in the LP) and recognizes the characters on the LP.	90
5.5	Implementation of the integration block. F_r , F_{sq} , and F_{ex} are the residual branch, squeeze step, and excitation step, respectively. $[\cdot]$, $\odot$, and $\oplus$ denote the concatenation operation, element-wise product, and element-wise addition, respectively. $\{\cdot\} \times x$ denotes performing the $\{\cdot\}$ structure by x times. The convolutional layers have the kernel size of 3×3 in F_r and 1×1 in F_{ex}	91
5.6	Implementation of the weight-sharing classifiers.	97

5.7	Proposed dynamic regularization in S-Stage-1, 2, and 3 of SCR-Net. S-Conv-2 and S-Stage-4 are omitted for simplification. F_r denotes the residual function, i.e., $BN-Conv3-BN-ReLU-Conv3-BN$. $\nabla f(loss)$ denotes a backward difference of the training loss. The dynamic regularization aims to make a self-adaptive schedule during training by adjusting the strength of the random perturbation θ . As a manner of feature augmentation, the θ introduces noises for the residual branch in the forward and backward process.	98
5.8	Residual block of SCR-Net with dynamic regularization.	100
5.9	Qualitative results of VertexNet. Green, blue, and red bounding boxes represent ground truth, truth positive detections, and failure detections, respectively. White points denote the predicted vertices. (a)-(g) show truth positive detections with the $IoU \in [0.7, 1.0]$. (h)-(o) shows failure detections with the $IoU \in [0.0, 0.7]$	105
5.10	The accuracy of SCR-Net and the count of detections of VertexNet with respect to IoU interval.	108
5.11	Qualitative results of our SCR-Net. Error characters are marked by the red color. False detections in (c) cause false recognition, but SCR-Net can still recognize these images as the “background” rather than LPs.	109
5.12	Some sampled LP patches with their predictions by SCR-Net and SCR-Net-Small. Error characters are marked by the red color. . . .	110

List of Tables

3.1	UIQM values among restoration methods.	48
3.2	Computation time of image restoration methods.	48
3.3	Comparison of L1 errors over synthetic hazy images. The table presents the L1 errors of the estimated transmission (left) and the dehazed images (right).	49
3.4	Quantitative comparisons on synthetic images.	51
3.5	Quantitative comparisons on natural images.	51
4.1	Crowd counting and detection tasks addressed by dense regression-based methods and dense classification-based methods, respectively.	57
4.2	Parameter settings of the LUDA-based pseudo size generation method for different datasets. “-” means the parameter is not required.	72
4.3	Comparisons of our LUDA-based pseudo size generation method with the GAK-based method on the training set of WiderFace [6]. Refinement of pseudo object sizes by our self-training approach is listed at the last entry. The number in the parentheses denotes the IoU threshold for AP calculation. The larger the value of AP is, the better.	73
4.4	Ablative studies towards the crowdedness-aware loss and the confidence and order-aware refinement scheme, as well as comparisons with state-of-the-art point-supervised detection method [19] and counting method [20] on the validation set of WiderFace [6]. “C-by-D” means counting by detection, and “C-by-R” means counting by regression. “Size Init.” stands for directly applying the pseudo object size initialization method on the validation set to obtain the detection result. “-” is not applicable. The larger the value of AP is, the better. The lower the values of MAE and NAE are, the better.	75
4.5	Comparisons with state-of-the-art point-supervised detection methods on the validation set of WiderFace [6]. “*”: the AP results are provided by [19]. The larger the value of AP is, the better. The initialized pseudo boxes were used when training CSP with point supervision.	77
4.6	Comparisons with crowd localization methods on SHA and SHB [7] datasets. The larger the value of AP is and the lower the value of MLE is, the better.	78

4.7	Comparison with state-of-the-art crowd localization methods on the NWPU-Crowd test set. The values of F1-measure, Precision, and Recall were calculated under the threshold σ_l [2].	79
4.8	Comparisons with state-of-the-art counting-by-regression methods (in the top part) and counting-by-detection methods (in the bottom part) on SHA and SHB [7], and WiderFace [6] (WF) datasets. “-”: the author does not provide the result. “*”: the results are provided by [21]. The lower the values of MAE and RMSE are, the better.	80
4.9	Comparisons with state-of-the-art vehicle counting methods on the CARPK and PUCPR+ benchmark [8]. The lower the values of MAE and RMSE are, the better.	81
4.10	Comparison on UCSD dataset.	81
4.11	Comparison on TRANCOS dataset. ‘-’ indicates the method does not provide the results.	82
4.12	Ablation study. ‘-’ indicates the model diverges.	83
5.1	Comparisons of the characteristics between state-of-the-art methods and VSNet. Performance is evaluated on CCPD [3].	89
5.2	Descriptions of SCR-Net. “Size” and “CH” are the output size and channel of feature maps, respectively. Conv3: 3×3 convolution. BN: batch normalization. Conv4: 4×4 -s2 convolution for downsampling the feature maps by a factor of 2. ReLU: rectified linear unit. Dy: dynamic regularization. Conv8 $\times$ 1: 8×1 -s1 convolution for a horizontal encoding. Conv1 $\times$ 3: 1×3 -s1 convolution for 1×32 features encoding. Res-block: residual block [22] with the branch of two Conv1 $\times$ 3-BN-ReLUs. $\{ \cdot \} \times x$ denotes performing the $\{ \cdot \}$ structure by x times.	96
5.3	Comparisons of LP detection methods on the CCPD testing set. “-” means the result is not provided. “Time” means the inference time (ms/image).	104
5.4	Ablation studies of VertexNet on the CCPD testing set. “Augment”: vertex-based data augmentation. “Est.”: vertex estimation. “#Params”: the number of network’s parameters. “ $\sqrt{}$ ” means using the corresponding technique.	106
5.5	Comparisons of LP recognition methods on the CCPD testing set. “-” means the result is not provided. “Time” means the inference time (ms/image). “*” means the recognition results of HC [23]. “**” means the method trained with synthetic data. In our method, the inference time of VertexNet+SCR-Net (i.e., VSNet) and only SCR-Net are shown outside and inside the parentheses, respectively.	108
5.6	Comparisons of LP recognition methods on the AOLP dataset. “box” means the predicted boxes by Attentional Net [24]. “vertex” is the ground truth vertex. “FC” is the fully-connected classifier.	109

5.7	Ablation studies of SCR-Net with different input types. “Box” and “Vertex” mean the LPs are sampled according to bounding boxes and vertices, respectively. “Predicted” and “GT” mean the boxes (or vertices) are generated by VertexNet and ground truth, respectively. “revised metric” in the parentheses means removing the condition of IoU larger than 0.6.	111
-----	---	-----

List of Acronyms

CNN	Convolutional Neural Network
DNN	Deep Neural Network
FCN	Fully Convolutional Network
RNN	Recurrent Neural Network
LOS	Line of Sight
IFM	Image Formation Model
TV	Total Variation
AACP	Adaptive Attenuation-curve Prior
FACP	Fixed Attenuation-curve Prior
DRS	Distant Region Segmentation
Crowd-DCNet	Crowd Detection and Counting Network
LUDA	Locally-uniform Distribution Assumption
MSD	Multi-scale Density
ALPR	Automatic License Plate Recognition
VertexNet	Vertex Detection Network
SCR-Net	Squeeze Character Recognition Network
VSNet	Cascaded Framework with VertexNet and SCR-Net
IB	Integration Block
SE	Squeeze-and-Excitation
MTL	Multi-task Learning
AP	Average Precision
GAME	Grid Average Mean Absolute Error
MAE	Mean Absolute Error
MLE	Mean Localization Error
MSE	Mean Squared Error

NAE	Normalized Mean Absolute Error
UIQM	Underwater Image Quality Metrics
IoU	Intersection of Union
NMS	Non-maximum Suppression
ROI	Region of Interest
GT	Ground Truth

Chapter 1

Introduction

1.1 Background

Image processing and computer vision have been gaining much attention for years. Mimicking the human visual system, the computer vision system attempts to perceive and understand the surroundings [25] through extracting information from images or videos. With the advent of deep neural networks (DNNs), computer vision algorithms achieve high performance in a wide range of applications, such as video surveillance [26], autonomous driving [27], and crowd analysis [28].

However, images/videos captured in complex visual scenes pose a challenge to computer vision systems. Fig. 1.1 demonstrates some complex scenes. The images captured in haze and underwater present a low contrast and color distortion due to light scattering, as shown in Fig. 1.1(a). Another issue is that the objects themselves are complicated, such as crowds spread all over the image with severe occlusion and large-scale variation, as shown in Fig. 1.1(b). Besides, small objects in complex environments present more challenging cases, such as license plates captured in bad weather, uneven light, and oblique views, as shown in Fig. 1.1(c). Computer vision tasks in such scenes are still a challenging problem.

It is valuable and important to develop a robust and efficient computer vision system to address those issues. The system should accomplish a series of low-level and high-level vision tasks in order to understand image/video contents, as shown in Fig. 1.2. For example, low-level vision tasks involve noise removal techniques [29] for

FIGURE 1.1: Complex visual scenes. The images in (a), (b), (c) are obtained from [1–3].

noisy images, haze removal techniques [14] for hazy images, or image enhancement techniques [30] to enhance some object and scene features. High-level vision tasks focus more on understanding images by object detection [31], tracking [32], and recognition [33] techniques. With large-scale labeled datasets, modern DNN-based methods tend to solve the problems by a single end-to-end step. However, computer vision problems are complicated and cannot be simply solved by one of the methods.

FIGURE 1.2: Computer vision tasks for image understanding.

This thesis will follow the mainstream of image restoration, object detection, and object recognition, all of which we will discuss from a perspective of dense prediction. Besides, we will provide our solutions in these research fields.

1.2 Scope of Dense Prediction

Many abovementioned computer vision tasks can be considered as a dense prediction problem. Dense prediction aims at predicting the continuous or discrete labels for each pixel (or a group of pixels) of an image. Thus, the dense prediction is also known as a pixel-wise prediction. Compared with the image-wise prediction (like image classification [34] and face recognition [35]) which concerns the low-resolution and translation-invariance feature representation, the dense prediction instead recovers high-resolution representation with location-sensitive information. The dense prediction can be simply defined as

$$D = F(I), \quad (1.1)$$

where an input image I is processed by a set of functions F , generating a structural map D . Specifically, F is the task-specific functions, such as image filter, clustering operation, and DNN. Each pixel in D contains a label value as a representation for the corresponding pixel of the input image, and the structure of D is also task-specific. Various definitions of the labels decide the types of vision tasks, which can be divided into two categories, i.e., regression and classification. Dense regression methods are usually used for the prediction of continuous labels (like image restoration [30], image super-resolution [36], depth estimation [37], and density map-based crowd counting [7]), while dense classification methods generally deal with the prediction of discrete labels (like semantic segmentation [38] and keypoint-based detection [39]).

Some typical dense prediction methods (F) and the predicted structural maps (D) can be seen in Fig. 1.3. From the second column of Fig. 1.3, dense prediction with prior knowledge is employed for a low-level vision task, e.g., image restoration. Dense prediction with convolutional neural networks (CNNs) is employed for a median-level vision task (e.g., segmentation), high-level vision tasks (e.g., crowd detection, crowd counting, and license plate recognition). The structure of CNNs

FIGURE 1.3: Typical dense prediction methods and the predicted structural maps. We will present them in detail in the following chapters. *Zoom in the figure for better viewing.*

usually exploits an encoder-decoder manner. The encoder extracts a set of features from an input image, followed by a decoder to predict a structural map.

In this thesis, we extend the definition of dense prediction, i.e., the prediction from an image to the structured map. From the third column of Fig. 1.3, we can observe that the structured map can be a transmission map for image restoration (see Row 1), a binary mask for segmentation (see Row 2), a heatmap for object localization (see Row 3), a density map for crowd counting (see Row 4), or even a predefined dense vector for vehicle license plate recognition (see Row 5). Finally, with the

dense predictions, we can obtain the desired output results for each task, as shown in the last column of Fig. 1.3.

In this dense prediction scope, we will present the challenges and motivations of this thesis in what follows.

1.3 Challenge and Motivation

Complex visual scenes include extreme weather conditions, underwater, crowded objects, and small objects, which are accompanied by challenges and motivations.

1.3.1 Light Scattering Environments

Imaging is not a perfect process, which might be affected by extreme conditions, such as haze and underwater. Haze and underwater environments cause light scattering and absorption, leading to the low contrast and the distorted color of images, as shown in Fig. 1.1(a). Poor visibility not only harms photography (e.g., subsea exploration [40]) but also hinders vision-based applications (e.g., autonomous vehicle [41]). These extreme environments motivate us to investigate the image restoration and enhancement methods based on the image formation model (IFM). Accurate parameter estimation in IFM can produce clear natural images. (See Chapter 3)

On the other hand, the ambient light is a critical parameter for IFM in image restoration. It is nontrivial to obtain the ambient light from a single image. Traditional methods utilized some prior knowledge to estimate it, like dark channel prior [14]. However, these methods may be hampered by complex conditions, such as white objects and underwater scenes. Motivated by the ambient light's characteristics, we take advantage of a deep learning-based model to segment distant regions on an image and estimate the ambient light by the pixel values inside that region. (See Chapter 3)

1.3.2 Crowded Objects and Small Objects

Apart from complex surroundings, the objects themselves (e.g., pedestrians or vehicles) may present a crowded situation, resulting in a large-scale variation and severe occlusion in the image. For example, the persons near the camera have large sizes in the image, while the persons far from the camera show small sizes, even occupying several pixels. Besides, a large number of persons distributed in a region of the image present a severe occlusion, which introduces challenges for object detection and crowd counting, as shown in Fig. 1.1(b). Motivated by this condition, we propose a robust object detection method to localize and count objects, and a regression method for crowd counting. (See Chapter 4)

Another challenging issue is the small object size, which causes inaccurate object location and recognition. For example, license plate recognition methods need to localize and recognize small characters on license plates in unconstrained environments, as shown in Fig. 1.1(c). These methods should fulfill real-time and high-performance requirements, which encourages us to design an efficient and robust license plate detection and recognition method. (See Chapter 5)

1.3.3 Lack of Data Annotations

Traditional supervised learning of DNNs requires a huge number of labeled data to achieve a good generalization ability. For some visual scenes, however, such labeled data are unavailable or incomplete. For example, we can only capture the distorted images in underwater environments but cannot obtain clear images without water. This motivates us to study some prior knowledge of light attenuation in water and propose a method to remove that light attenuation. Besides, given clear images and the corresponding depth maps by RGB-D datasets (like NYU-depth2 [42]), the IFM can be employed to simulate the distorted images. (See Chapter 3)

Another case is that data annotations are incomplete and inaccurate due to the labeling costs and difficulties. In object detection tasks, the generic detectors are trained with the bounding boxes encompassing each object, and the target of those detectors is to predict the bounding boxes. However, most crowd counting datasets only contain point-level annotations, i.e., each person is labeled by a point inaccurately centered at the head part, which hinders the application of

state-of-the-art detection methods to localize each instance of crowds. By observing point-level annotations' characteristics, we propose a self-training approach to train a typical detector with such point annotations, enabling the detector to predict object sizes. (see Chapter 4)

1.4 Major Contributions

This thesis adopts dense prediction and deep learning methods for complex visual scenes. The main contributions of this thesis are summarized as follows.

(1) Image restoration in light scattering scenes.

We introduce an adaptive attenuation-curve prior (AACP), which can model the light attenuation process in various scattering environments. Based on the prior, we propose an underwater image restoration method to predict the transmission maps and recover the clear images. This method can also be extended to deal with hazy images. Moreover, according to ambient light's characteristics, we propose a deep learning-based framework for ambient light estimation by distant region segmentation (DRS). To deal with the lack of training examples, we introduce a technique to generate the distant region masks as ground truth during training. Qualitative and quantitative experiments demonstrate the effectiveness of the proposed methods.

(2) Object detection and counting in crowds.

We propose a novel self-training approach that enables a typical object detector trained only with point-level annotations to predict the center points and sizes of crowded objects. To address extremely crowded scenes, we propose an effective decoding method to improve the detector's representation ability. The enhanced detector not only localizes each object instance but also counts them accurately, termed Crowd-DCNet. Extensive experiments show the superiority of our approach in both crowd detection and counting tasks. Moreover, for low-resolution video surveillance, we propose an effective CNN-based method to estimate the crowd count by multi-scale density (MSD) regression. The model has a compact architecture and high performance in the video surveillance datasets.

(3) Small object detection and recognition in unconstrained scenes.

Targeting small objects (i.e., license plates), we propose a novel automatic license

plate recognition (ALPR) approach, termed VSNet. We employ a resampling-based cascaded manner to combine the detection (VertexNet) and recognition (SCR-Net) components. Without additional character segmentation or RNNs, the well-designed CNN-based approach presents a real-time capability (149 FPS) on a GPU. With vertex supervisory information, we propose a vertex-estimation branch for predicting the geometric shapes of license plates, which is further utilized for license plate rectification and recognition. Finally, a weight-sharing classifier is proposed to predict the character sequences on license plates. Those novel modules ensure the outstanding performance of our ALPR approach, which is demonstrated in the experiments.

1.5 Organization of the Thesis

The rest of this thesis is organized as follows. In Chapter 2, we introduce the related works and benchmark datasets about dense prediction, deep learning, and applications. In Chapter 3, we present the underwater image restoration method and the ambient light estimation method in detail. In Chapter 4, the proposed self-training method is introduced for point-supervised crowd detection and counting. Besides, a compact network is proposed to count crowds in video surveillance. In Chapter 5, we describe the real-time and robust approach for automatic license plate recognition. Finally, we conclude this thesis and present future research directions in Chapter 6.

Chapter 2

Literature Review

This chapter reviews the dense prediction methods for different-level vision tasks and presents the benchmark datasets relevant to the tasks, i.e., restoration, detection, and recognition.

2.1 Related Work

We divide dense prediction into two categories, i.e., dense prediction with prior knowledge and dense prediction with convolutional neural networks (CNNs), each of which contains several sub-categories and representative tasks, as shown in Fig. [2.1](#).

2.1.1 Dense Prediction with Prior Knowledge

In general, many image processing problems can be considered as *densely predicting* an output image from an input image, where both input and output are intensity images, and the process does not increase image information content [25]. Image pre-processing is a typical dense prediction operation. The purpose of pre-processing is to suppress the undesired distortions (like noise or light scattering) due to the imperfect image acquisition, named restoration, or to enhance specific image features for higher-level processing, named enhancement.

FIGURE 2.1: Categories of dense prediction. The tasks are listed in the non-blue boxes. Among the tasks, object detection connects with dense classification and regression because it not only densely classify every bounding box but also regress the offset of the bounding boxes. The tasks in green boxes will be described in detail in this chapter and the following chapters accordingly.

Prior knowledge refers to the knowledge acquired from sample data about the problem. Image restoration methods employ prior knowledge to describe the nature of the image degradation process and restore clear images. For example, the additive Gaussian noise model is used for denoising [29], and the dark channel prior is used for dehazing [14] based on the atmospheric scattering model. Here we investigate a typical degradation process, i.e., light scattering, and present state-of-the-art image restoration methods for dehazing (or de-scattering).

2.1.1.1 Atmospheric Scattering Model

The atmospheric scattering model [14, 43] is a physical-based model, which can be formulated as follow:

$$I(x) = J(x)t(x) + A(1 - t(x)), \quad (2.1)$$

where x represents pixel coordinates, I is the observed image, J is the scene radiance, i.e., the clear image, A is the ambient light (namely airlight in atmospheric condition), and t is the medium transmission representing the rate of light is not scattered and reaches the camera. The transmission $t(x)$ depends on the scene depth $d(x)$ and

scattering coefficient β :

$$t(x) = e^{-\beta d(x)}, \quad (2.2)$$

where β is a constant in the homogeneous atmosphere. In Eq. (2.2), we can know that the degradation process is spatially variant. That is, the degree of light scattering relies on the distance from scenes to the camera. Therefore, the distant scenes lose their original color and become airlight's color, leading to low contrast. This phenomenon also appears in water media. Underwater images suffer from light scattering and light absorption, which will be described in Chapter 3.

Image restoration aims to recover the scene radiance by estimating the airlight and transmission, which is a kind of ill-posed problem since we only know the observed image. With the estimated airlight and transmission, the clear image is arithmetically obtained by

$$J(x) = \frac{I(x) - \hat{A}}{\hat{t}(x)} + \hat{A}. \quad (2.3)$$

2.1.1.2 Image Restoration Methods by Prior Knowledge

Prior knowledge can be used to remove haze effects. Tan [44] proposed a novel approach based on two priors: contrast prior that the contrast of the haze-free image should be higher than that of the hazy image, and smooth prior that the light attenuation of field spots is smooth. Fattal [43] introduced an effective method to estimate the medium transmission by the knowledge that the transmission is not related to the object surface's shading. Tarel *et al.* [45] presented a contrast-based method by assuming the atmospheric veil function changes smoothly over a local region.

Dark channel prior and its variants. He *et al.* [14] proposed a dark channel prior (DCP) to estimate the transmission and airlight. The definition of DCP is, in most of the non-sky image patches, there exist certain pixels whose values are very low or even close to zero in a certain color channel. There are three factors for the low intensity in the dark channel: shadows of objects, colorful objects or surfaces, and dark objects or surfaces. Due to attenuated scene radiance and additive brighter ambient light (see Eq. (2.1)), the hazy image's dark channel is brighter than that of the corresponding haze-free image.

Based on DCP, many approaches have been proposed to improve image restoration performance. Meng *et al.* [46] employed DCP to estimate an initial transmission and proposed a novel contextual regularization by a weighted L_1 -norm. They presented a variable splitting-based algorithm to solve this optimization problem. The results show a visually pleasing image with faithful color and finer details. Golts *et al.* [47] introduced “Deep DCP”, i.e., an unsupervised training method to minimize the DCP energy function on hazy outdoor images. However, the DCP does not work well in sky areas, white objects, and underwater scenes.

DCP can be adjusted for underwater environments. Chiang and Chen [48] restore underwater images by DCP for haze removal and wavelength compensation for color collection. Galdran *et al.* [49] presented a red channel prior, in which the most contaminated red color is recovered. Drews *et al.* [10] presented an underwater dark channel prior (UDCP), which considers the blue and green color channels as visual information sources in the underwater situation. Based on UDCP, Hou *et al.* [50] proposed an underwater total variation (UTV) model, where the data and smooth items of the unified variational model are established.

Other state-of-the-art prior knowledge. While Retinex has been widely explored for image enhancement, it can be seen as prior knowledge and explicitly or implicitly used in image restoration. Galdran *et al.* [51] proved a simple linear relationship between Retinex and image dehazing:

$$\text{Dehazing}(I) = 1 - \text{Retinex}(1 - I). \quad (2.4)$$

They also showed that the equivalence holds at both the algorithmic level and the modelization level. Based on this, the existing Retinex-based algorithms can use a channel-wise intensity inversion operation to remove haze. Fattal *et al.* [52] proposed color-lines to fit small image patches with a constant transmission. A haze-free image’s pixel values pass through the RGB space’s origin, forming the color-lines [45]. However, in hazy images, the color-lines shift from the origin by a distance due to the additive airlight. The method [52] fits a line to a small image patch and estimates the transmission by the shift. Since several parts of the patches satisfy this assumption, they predict the rest of the patches’ transmission by Markov random field.

Inspired by color-lines, Berman *et al.* [15] proposed a haze-line that is formed by non-local pixels distributed in the entire hazy image. Specifically, the haze-line prior describes that a hazy image's pixel values can be modeled as lines intersected at the airlight in RGB space. The method finds the haze-lines centered on the airlight, calculates each pixel's initial transmission, and regularizes them with weighted least squares (WLS) [53]. Berman *et al.* [16] also proposed an airlight estimation method, assuming that the airlight location can fit pixel values well in such a haze-line model. They used the Hough transform to vote for each candidate location of RGB space and determine which location has the highest vote. An efficient Gradient channel prior (GCP) [54] was proposed to resolve the texture distortion, errors in transmission estimate, color distortion, and edge degradation, and then a guided L_0 filter was introduced to refine the transmission map.

For underwater images, Carlevaris-Bianco *et al.* [9] proposed the maximum intensity prior (MIP), describing a large attenuation difference among three color channels in the water. They utilize MIP to estimate the underwater transmission map. Peng and Cosman [11] proposed a depth estimation method based on image blurriness and light absorption to handle varying lighting conditions. An improved guided filter to refine the transmission map, named secondary guided transmission map [55], was proposed to recover the contrast, color, and natural appearance of underwater images.

2.1.2 Dense Prediction with Convolutional Neural Networks

Modern deep learning provides deep neural networks (DNNs) for powerful feature representation [56]. Via stacking a large number of layers, DNNs can solve complex nonlinear problems, even though human beings can easily perform but cannot describe the problem by a list of mathematical rules, such as face detection and recognition. Given sufficiently large labeled datasets, DNNs leverage the multiple layers to map input signals to semantic information. The input signals can be images, videos, and audios. Since this thesis focuses on images, the convolutional neural networks (CNNs) will be presented in detail.

A typical CNNs adopt a stack of 2D convolutional layers (linear operation), activation functions (nonlinear operation), and pooling layers for sub-sampling. For example, LeNet-5 [4], a pioneer network architecture, was used for hand-written

digit recognition (see Fig. 2.2). Three critical ideas make CNNs powerful. (1) Local connectivity. Instead of connecting high-dimensional inputs, CNNs adopt a small convolutional kernel, which only concerns spatially local correlation. This operation significantly reduces the number of parameters. (2) Parameter sharing. The kernel is used at every place of inputs, which reduces memory usage and implicitly increases the training samples. (3) Translation invariance of convolution. The outputs follow the translation of inputs. Besides, pooling layers reduce the dimension of features and the computation of networks and produce translation-invariant features.

FIGURE 2.2: Architecture of LeNet-5 for hand-written digit recognition [4] © 1998 IEEE.

Recently, with large-scale datasets (e.g., ImageNet [34]) built, researchers started exploring deeper and more effective structures, e.g., AlexNet [34], VGG [57], GoogleNet [58], and ResNet [22]. With batch normalization [59] and skip connection, ResNet can reach hundreds of layers. Such deep CNNs has a powerful feature representation ability.

Those type of networks were originated for image classification, which predicts a label for an image without considering object and scene positions. However, in dense prediction, CNNs should predict labels for every part of an image, or even pixel-level labels. Accordingly, Long *et al.* [5] proposed a fully convolutional network (FCN), where a classification network is used as the backbone network (termed encoder), followed by a sequence of deconvolutions (or transposed convolutions) to upsample the encoded small-resolution features to high resolution (termed decoder). Fig. 2.3 shows the structure of FCN for semantic segmentation (sourced from [5]).

Inspired by FCN, a number of state-of-the-art methods have been proposed, like U-Net [60], deep layer aggregation (DLA) [61], Image-to-Image Translation [62], and high-resolution network (HRNet) [63], for dense prediction. The key insight of these networks is fusing multi-scale features and aggregating contextual features to

FIGURE 2.3: Architecture of FCN for semantic segmentation [5] © 2015 IEEE.

generate high-resolution informative features. They can be used for position-sensitive tasks, e.g., object detection, semantic segmentation, human pose estimation, crowd localization.

In this thesis, we classify CNN-based dense prediction methods by their training objective rather than the tasks they accomplish. Specifically, they can be classified into two sub-categories, i.e., dense regression and dense classification, with respect to the predicted label's characteristics. From the right branch of Fig. 2.1, we can observe the category hierarchy and the corresponding tasks.

The predicted labels can be either continuous values (e.g., transmission map for image restoration [64], density map for crowd counting [65], and depth map for depth estimation [37]) or discrete values (e.g., binary mask for semantic segmentation [60], heatmap for crowd localization [39], and vector for license plate recognition [66]). Therefore, the dense prediction tasks become either a dense regression problem of continuous labels or a dense classification problem of discrete labels. We will discuss each sub-categories and the relevant computer vision tasks (see the green boxes of Fig. 2.1) in detail as follows.

2.1.3 CNN-based Dense Regression

If the predicted labels are continuous values, the dense prediction becomes a dense regression problem wherein a CNN extracts features by an encoder from an input image and estimates a continuous-value structural map. Given a labeled dataset, training the model is to build a mapping between the input images and the structural map. For the loss function, the L1, L2, Smooth-L1 [67], structural similarity (SSIM)

[68], and adversarial [62] losses are commonly used according to tasks' characteristics. In video surveillance and safety monitor, restoring high-quality images and analyzing crowd status are two basic tasks. Here we introduce several related dense regression works about image restoration and crowd counting.

2.1.3.1 Image Restoration

CNNs have been exploited for image restoration in light scattering scenes. These approaches use an encoder-decoder network to generate a transmission map and airlight, thereby calculating the haze-free image by Eq. (2.3), or directly yield the haze-free image from the hazy image. Cai *et al.* [64] introduced a dehazing CNN (named DehazeNet) to predict the transmission map with a bilateral rectified linear unit (BReLU). Ren *et al.* [69] proposed a multi-scale deep neural network (MSCNN), which can take a hazy image as input and produce a transmission map. The method uses a coarse-scale net to estimate a holistic transmission map and a fine-scale net to refine it. The airlight is the brightest pixel value among the pixels whose transmission value smaller than 0.1. Combining the physical image formation model with deep learning, Liu *et al.* [70] formulated the image dehazing problem as a minimization of the variational model, where the model is regularized with favorable data fidelity terms and prior terms. The model is resolved by a gradient descent-based Gauss-Seidel iteration algorithm.

Another type of structure is an end-to-end CNN. Li *et al.* [71] proposed an all-in-one dehazing network (AOD-Net), in which the atmospheric scattering model (Eq. (2.1)) is reformulated as

$$\begin{aligned} J(x) &= K(x)I(x) - K(x) + b, \text{ where} \\ K(x) &= \frac{\frac{1}{t(x)}(I(x)-A)+(A-b)}{I(x)-1}. \end{aligned} \quad (2.5)$$

$K(x)$ integrates $t(x)$ and A into one formula, and b is a constant bias. They used AOD-Net to estimate $K(x)$ and calculate $J(x)$ by Eq. (2.5). Based on Image-to-Image Translation [62], Zhang and Patel [17] proposed a novel densely connected pyramid dehazing network (DCPDN), which embeds Eq. (2.1) into the network and jointly optimizes transmission map and airlight estimation. Qin *et al.* [72] introduced a feature fusion attention network (FFA-Net) to recover the haze-free image from a hazy image.

Underwater image restoration is a more complicated image de-scattering task, where light scattering and absorption occur concurrently. Hu *et al.* [73] proposed a CNN-based model to estimate ambient light and transmission. To train the model, they synthesized underwater images by randomly selected green-hue or blue-hue ambient light, indoor images, and depth maps from the Middlebury Stereo [74] dataset. Taking advantage of CycleGAN [75], Fabbri *et al.* [76] improved the quality of visual underwater scenes by generating pairs of underwater and clear images without depth information. Dudhane *et al.* [77] introduced a network composed of channel-wise color feature extraction and dense-residual feature extraction. The network is trained with the combination of losses (i.e., GAN loss, L_1 loss, SSIM loss, and edge loss) on the synthetic underwater image database. Inspired by CycleGAN, Cho *et al.* [78] proposed an unpaired image translation framework into image restoration, where the framework is trained with multiple cyclic consistency losses.

However, CNN-based methods depend on training examples. Although some depth databases are used to synthesize training examples, these databases only contain indoor scenes, resulting in cross-domain problems for natural outdoor images. The lack of ground-truth image pairs limits the dehazing performance for deep learning-based approaches.

2.1.3.2 Crowd Counting

Modern crowd counting approaches [18, 20, 68] exploit CNNs to estimate a density map [79], over which the count is calculated via the integration operation. The typical network architecture is generally composed of an encoder for extracting a set of features from an image and a decoder for regressing a density map [80]. In [7], a multi-column structure was introduced to learn the multi-scale features for representing the large-scale variation of objects. Li *et al.* [18] demonstrated the redundancy of features in the multi-column structure and proposed a single-column deep structure (e.g., VGG [57]) with dilated convolution layers, which achieved better results. Cao *et al.* [68] proposed a scale aggregation module to learn the scale diversity of features. Shi *et al.* [20] repurposed point annotations as a segmentation map and a global density map to training the network.

By exploiting the attention mechanism, Sindagi *et al.* [81] proposed an attention-based crowd counting network, named HA-CCN, enhancing different-level features.

Wang *et al.* [80] proposed a pixel shuffle decoder (PSD) to generate a high-resolution density map and presented a density-aware curriculum learning (DCL) strategy to improve the network’s generalization ability and reduce training time. A recent survey paper by Gao *et al.* [82] reviews over two-hundred crowd counting works and some datasets, showing the development of this field. The paper also summarizes the performance evaluation of representative crowd counting methods.

2.1.4 CNN-based Dense Classification

If the predicted labels are discrete values, the dense prediction becomes a dense classification (or pixel-wise classification) on whether the pixel is the target class or background. A typical CNN-based model extracts the features by an encoder from the input image and estimates a discrete-value structural map. The cross-entropy loss and its variants were widely used. In video surveillance, detecting crowded pedestrians and recognizing small characters on license plates are two challenging tasks. Here we introduce these two tasks in detail.

2.1.4.1 Crowd Localization and Detection

Benefiting from the advances of deep CNNs, current object detectors, such as Faster R-CNN [83], SSD [84], RetinaNet [85], and CenterNet [39, 86], achieve appealing performance. However, these methods encounter difficulties when counting small and heavily occluded objects in crowded scenes. To enhance the detectors’ abilities, Liu *et al.* [87] proposed a keypoint-based detector for pedestrians’ center and scale prediction, named CSP. Goldman *et al.* [88] presented a deep learning-based method for precise object detection in densely packed scenes. They introduced a soft intersection-over-union (Soft-IoU) network layer and an expectation-maximization (EM) based clustering method to deal with overlapped objects in the dense scenes. Despite such remarkable progresses, they have to be trained and supervised by box-annotated examples.

Most crowd counting datasets only provide point annotations for dense crowds, i.e., labeling human heads with points. It is nontrivial to train a detector with point supervision. Recently, several works begin to use the blobs [89] to localize all instances of crowds and even to estimate the heads’ sizes [19, 21], with only

point-level annotations. Laradji *et al.* [89] argued that the unnecessary size and shape estimation drags the performance of detection-based methods in the counting problem. A localization-based counting loss that combines image-level, point-level, split-level, and false-positive loss is used to train an FCN such that it could produce the blobs at the center of objects. Based on a regression network, Idrees *et al.* [90] proposed a post-processing method to find the local peaks at the density map as heads' center locations. As a baseline method, Sam *et al.* [21] applied a threshold technique on the density map of the CSRNet [18] to obtain detections (called CSR-A-thr). However, these methods only localize the center points of heads.

To further estimate heads' sizes, Liu *et al.* [19] proposed a detection network named PSDDN, which builds a strong baseline for point-supervised detection and counting in crowds. The PSDDN employs the nearest neighbor distance [7] to initialize the pseudo boxes and updates them by choosing smaller box predictions. Another state-of-the-art method, named LSC-CNN, was proposed recently by Sam *et al.* [21], where the objective is to locate, size, and count every person in crowds. The detection methods can be directly used to count objects [21] since the number of detections is also the count of objects.

2.1.4.2 License Plate Detection and Recognition

The scope of automatic license plate recognition (ALPR) is the task of automatically localizing and recognizing license plates (LPs) in images. The typical process is comprised of two components, i.e., LP detection and LP recognition.

License plate detection. TE2E [91] employed RPN [83] to generate high-quality bounding box proposals, and used the RoI pooling layer to obtain region features for LP detection. YOLO networks [92, 93] are prevalent in LP detection because of their high efficiency. Silva and Jung [94] are pioneers who introduced two cascaded YOLO networks to detect vehicle and LPs, respectively. Other works in [66, 95–98] also adopted the modified YOLO, YOLOv2 networks. Xu *et al.* [3] introduced multi-scale feature maps for box regression and character recognition. The multi-scale features can help to detect objects on different scales. Based on the Cascaded CNN [99], Wang *et al.* [100] proposed a multi-task CNN (MTCNN) to detect LPs, including plate classification, bounding box regression, plate landmark localization, and plate color recognition.

License plate recognition. The ultimate goal of the ALPR system is to recognize all letters and numbers on a given LP. There are two categories of LP recognition: segmentation-based recognition and segmentation-free recognition.

Segmentation-based recognition methods first segment an LP into character regions, and then the characters are recognized one-by-one via optical character recognition (OCR) techniques [101, 102]. However, these methods suffer from segmentation errors, which hinders their applications in unconstrained environments. A semantic segmentation network was proposed to segment characters by Zhuang *et al.* [103], improving the recognition accuracy. However, the results are achieved by laborious character annotations.

Segmentation-free recognition directly performs LP recognition without character segmentation. LP recognition can be referred to as recognizing a sequence of characters, and RNN-based structures were widely adopted. In TE2E [91], Li *et al.* proposed a bidirectional RNNs (BRNNs) with connectionist temporal classification (CTC) loss [104] to process LP features by RPN. Similarly, Wang *et al.* [100] proposed a convolutional RNN (CRNN) followed by CTC for character recognition. Combining with the abovementioned MTCNN detection, they proposed a multi-task license plate detection and recognition (MTLPR) model. To handle LPs' oblique views, Zhang *et al.* [24] proposed a tailored Xception network for feature extraction and a 2D-attention-based RNN for character recognition, named Attentional Net, which obtains the state-of-the-art performance in both constrained and unconstrained environments. Generally, RNN-based structures are less efficient than CNN-based methods when they have the same parameters. In RPnet [3], Xu *et al.* employed a series of CNN-based classifiers to recognize characters in predefined positions. Although the RPnet improves the inference speed, its performance is inferior to the RNN-based methods, such as MTLPR and Attentional Net.

2.2 Benchmark Dataset

This section demonstrates popular benchmark datasets for dense prediction problems, i.e., image restoration, crowd detection and counting, and license plate detection and recognition.

2.2.1 Image Restoration Dataset

Underwater datasets usually provide the degraded images without the corresponding clear images as reference. It is difficult to collect those pairs of images and perform a reference comparison. Thus, we adopted a non-reference comparison for underwater image restoration. Fig. 2.4 shows 37 sample images available from References [9, 43, 49, 105, 106] for evaluating image restoration methods.

FIGURE 2.4: Underwater images.

In terms of hazy images captured in the atmosphere, a synthetic benchmark dataset [52] was proposed to evaluate image dehazing methods. The hazy images were produced by the corresponding haze-free images and depth maps. Fig. 2.5 shows pairs of synthetic hazy images (top rows) and haze-free images (bottom rows).

FIGURE 2.5: Synthetic hazy images and haze-free images. Left to right then top to bottom corresponds to Church, Couch, Flower1, Flower2, Lawn1, Lawn2, Mansion, Moebius, Raindeer, Road1, and Road2, respectively.

2.2.2 Crowd Detection and Counting Dataset

We introduce four representative datasets in crowd scenes, i.e., WiderFace [6] for dense face detection, ShanghaiTech [7] for crowd counting, NWPU-Crowd [2] for crowd localization, and CARPK and PUCPR+ [8] for vehicle counting from the drone view. Fig. 2.6 shows examples in those datasets.

FIGURE 2.6: Example images with crowds or vehicles on (a) WiderFace [6], (b) SHA [7], (c) SHB [7], (d) NWPU-Crowd [2], and (e) CARPK and PUCPR+ [8]. The crowd locations are presented by the green boxes or points.

WiderFace is one of the most challenging face detection benchmarks. The 32,203 images contain 393,703 human-labeled faces were captured in a wide variety of imaging conditions, such as large variations in scale and pose, severe occlusion, and varying illumination conditions. 40%, 10%, and 50% of the images were used for training, validation, and testing, respectively. Following the experimental protocols of existing point-supervised detection methods [19, 21], we trained our model on the training set and reported detection and counting results on the validation set.

ShanghaiTech presents high-density crowds, including 482 images on Part_A (SHA) and 716 images on Part_B (SHB). The number of people in an image ranges from 33 to 3139 on SHA and 9 to 578 on SHB. We followed the training and testing split in [7] to evaluate the counting and central point localization performance.

NWPU-Crowd provides a large-scale benchmark for crowd counting and localization. It consists of 3109, 500, and 1500 images for the training, validation, and test, respectively, and the images contain more than 2 million annotated heads with points and boxes.

CARPK and PUCPR+ are composed of images of parking lots from the drone view and high-rise buildings, respectively. CARPK contains nearly 90k cars, while PUCPR+ contains about 17k cars in total. We used the evaluation protocol of [8] to evaluate the counting performance.

2.2.3 License Plate Detection and Recognition Dataset

CCPD. The license plate detection and recognition methods that can handle uncontrolled conditions have become attractive and challenging, so we use an extensive and comprehensive LP benchmark, i.e., Chinese city parking dataset (CCPD) [3]. Examples of the dataset are shown in Fig. 2.7. The CCPD dataset contains over 280k vehicle images captured in uncontrolled conditions, e.g., different weathers, illuminations, rotation, and vagueness, two orders of magnitude more than other LP datasets. Each image has three color channels and a resolution of 720×1160 . The dataset provides sufficient annotations, such as LP character, bounding box, four vertices, horizontal and vertical tilt degree, and the degree of brightness and vagueness.

FIGURE 2.7: License plate images captured in unconstrained scenarios, e.g., (a) and (e) oblique view, (b) uneven illumination, and (c) snow in the nighttime, labeled by ground-truth (GT) bounding boxes, four vertices, and characters.

AOLP. The application-oriented license plate (AOLP) dataset proposed by Hsu *et al.* [101] shows three types of constrained subsets, i.e., access control (AC) for the entrance/exit of a region, law enforcement (LE) for the vehicles that violate traffic laws captured by a roadside camera, and road patrol (RP) for the vehicles with arbitrary viewpoints and distances. AOLP only contains 2,049 images, which is challenging for DNN-based classifiers.

Chapter 3

Image Restoration in Light Scattering Scenes

Image restoration can be referred to as a dense prediction problem or image-to-image translation [62], i.e., translating a degraded image to a clear image. This chapter presents a generic image restoration method in light scattering scenes, in which a full-resolution transmission map is estimated for recovering a clear image. Meanwhile, a deep learning-based method is proposed to estimate ambient light (airlight), which is regarded as a distant region segmentation problem. The method predicts a binary mask to distinguish the distant region and the non-distant one. These two dense prediction methods are shown in Fig. 3.1.

FIGURE 3.1: Image restoration and ambient light estimation methods.

We study the underwater situation because it is a general case of light scattering. Haze or fog in atmospheric environments is a special case where the medium transmission is wavelength-independent. In various water types, light is absorbed and scattered at different degrees where the median transmission is wavelength-dependent. Therefore, the proposed method can also be applied to remove haze. In experiments, we will evaluate the effectiveness of the proposed methods and combine them to obtain further improvement.

3.1 Introduction

Under scattering media, such as haze, fog, or water, imaging suffers from light attenuation, resulting in low contrast and distorted color. Poor visibility not only harms consumer photography but also hinders computer vision applications, e.g., subsea exploration [40, 107] and autonomous vehicles [41]. Low-quality underwater images/videos impede the study of marine biology and geological environments. Those images may also compromise the performance of high-level vision tasks, such as image detection [71] and recognition [108]. Hence, this topic has attracted much attention in image processing and computer vision communities.

FIGURE 3.2: Atmospheric scene of light propagation. Airlight mixes with scattered light along the line of sight to the camera.

These problems are mainly caused by light attenuation, i.e., light scattering and absorption of the reflected light from the scene [40, 109]. For atmospheric environments, light scattering contributes the most parts of attenuation, as shown in Fig. 3.2. Due to the existence of dust-like particles floating in the atmosphere, images are contaminated by the scattering effect. When light interacts with the particles, the irradiance from the scene point is scattered from the line of sight

(LOS), and the airlight is concurrently mixed into the LOS, which fades the observed pixel value. Thus, light scattering reduces the contrast of images.

Apart from scattering, the light absorption phenomenon occurs in underwater environments (see Fig. 3.3), where the light absorption rate varies with its wavelengths. Different wavelengths of light disappear diversely along with the depth of water, as shown in Fig. 3.4. The red light with the longest wavelength is absorbed rapidly, while the blue and green light can penetrate longer distances. Therefore, underwater images have a bluish or greenish tone. In such scenes, we have to deal with two major issues, i.e., low contrast and color cast.

FIGURE 3.3: Underwater scene of light propagation, airlight enters from air to the water, part of the light attenuates along the line of sight. The point x represents the nearest scene point to the camera.

FIGURE 3.4: Penetration of light with different wavelengths in the ocean.

Recently, many researchers have made significant efforts to resolve these problems. The image de-scattering (or dehazing) methods have improved the imaging quality significantly. In terms of the algorithms, we divide image dehazing methods into three categories: image formation model (IFM)-based methods, IFM-free methods, and deep learning-based methods. The IFM-based methods are based on the image formation model and exploit prior knowledge to estimate key parameters in the model and then reverse the degradation processes. Dark channel prior [14] is a canonical work for this kind of method. The IFM-free methods include histogram equalization [13], Retinex method [51], fusion techniques [110], etc. They do not take the image formation model into consideration but mainly employ traditional image processing methods. Recent learning-based methods use convolutional neural networks (CNNs) to estimate key parameters of IFM (like [64, 69]) or design an end-to-end CNN to obtain clear images directly (like [71]).

In this chapter, we mainly focus on IFM-based methods and deep learning-based methods. We propose two novel approaches: underwater image restoration and airlight estimation. We first study the degeneration process of underwater images. Depending on the underwater image formation model, we propose an adaptive attenuation-curve prior (AACP). The prior can model the light attenuation process in different underwater scenes. With this prior, we present a novel underwater image restoration method to recover clear images adaptively. Besides, this method can prevent over-saturation and decrease noises when dehazing. Qualitative and quantitative results show that the proposed approach can achieve better performance in comparison with state-of-the-art methods. It is also a generic method that can be extended to recover other kinds of degraded images, such as hazy images taken in atmospheric environments.

Ambient light (or airlight) is an important parameter of IFM in image restoration. The accuracy of airlight estimation impacts the color and brightness of restored images. We present a deep learning-based approach to estimate this parameter in IFM. Via the assumption that ambient airlight exists in the infinite distance, we employ U-Net [60] to find the distant region segmentation (termed DRS) and then estimate the airlight inside that region. Due to the lack of ground truth, we present a method to synthesize outdoor training examples, in which pairs of the hazy image and distant region mask are generated. The proposed framework not

only performs well on synthetic images but also has a good generalization ability for natural images.

The rest of this chapter is organized as follows. In Sections 3.2, we introduce the proposed adaptive attenuation-curve prior, and we present the underwater image restoration method based on this prior. Then, in Section 3.3, we present the learning-based framework for ambient light estimation. In Sections 3.4, we conduct the experiments for image restoration and ambient light estimation, respectively. Finally, we conclude this chapter in Section 3.5.

3.2 Underwater Image Restoration with Adaptive Attenuation-curve Prior

In this section, we first introduce the underwater image modeling and adaptive attenuation-curve prior, and then we present the underwater image restoration method in detail.

3.2.1 Underwater Image Modeling and Adaptive Attenuation-curve Prior

3.2.1.1 Modeling of Underwater Images

Different from the degradation model of images captured in atmospheric environments (see Section 2.1.1.1), the degeneration process for underwater images can be generally formulated as:

$$I_c(x) = t_c(x) \cdot J_c(x) + (1 - t_c(x)) \cdot B_c, \quad c \in \{r, g, b\}, \quad (3.1)$$

where x represents pixel coordinates, c is the color channel index, I denotes the observed image, J denotes the scene radiance, i.e., the clear image, B denotes the waterlight determined by ambient illumination, and t is the medium transmission. The first term on the right-hand side of Eq. (3.1) is called *direct attenuation* that interprets the degradation of the scene radiance in the medium [14]. The second term is called *waterlight shift* which defines the shift from scene radiance to waterlight.

In such an ill-posed problem, we only know the observed image, and our purpose is to restore the scene radiance by estimating the waterlight and transmission.

Analysis of Eq. (3.1). The transmission $t_c(x)$ relies on the scene depth $d(x)$ and wavelength-dependent attenuation coefficient β_c :

$$t_c(x) = e^{-\beta_c d(x)} = e^{-(a_c+s_c)d(x)}, \quad c \in \{r, g, b\}, \quad (3.2)$$

where $\beta = a + s$ with a and s being the absorption and scattering coefficients, respectively. To get more accurate transmission, we represent the scene depth d as the summation of two distances, i.e., d_0 , the distance between the camera and the closest scene point, and d_n , the distance between the closest scene point and the farthest scene point, as shown in Fig. 3.3. Thus, Eq. (3.2) can be rewritten as:

$$t_c(x) = e^{-\beta_c d_0} \cdot e^{-\beta_c d_n(x)} = K_c \cdot t_c^n(x), \quad c \in \{r, g, b\}, \quad (3.3)$$

where $K_c = e^{-\beta_c d_0}$ is the attenuation factor, and $t_c^n = e^{-\beta_c d_n(x)}$ is named relative transmission.

Moreover, the transmission of green and blue channels exhibits a power function relationship with the transmission of the red channel [49]:

$$t_{c'}(x) = e^{-\beta_{c'} d(x)} = \left(e^{-\beta_r d(x)}\right)^{\beta_{c'}/\beta_r} = t_r(x)^{\lambda_{c'}}, \quad (3.4)$$

where $c' \in \{g, b\}$, and $\lambda_g = \beta_g/\beta_r$ and $\lambda_b = \beta_b/\beta_r$ are attenuation ratios between color channels. They are equal to one in the atmospheric environment, but usually, they are not identical in the underwater environments.

The absorption coefficient a varies irregularly with the light wavelength, while the scattering coefficient s changes little for different types of water, e.g., case 1 and case 2 water types [111]. Particularly, for red, green, and blue lights, the quantitative relationship between their scattering coefficients towards certain types of water has been derived via the least-square regression [111]:

$$\frac{s_{c'}}{s_r} = \frac{-0.00113\Lambda_{c'} + 1.6251}{-0.00113\Lambda_r + 1.6251}, \quad c' \in \{g, b\}, \quad (3.5)$$

where we choose three standard wavelengths $\Lambda_r=620$ nm, $\Lambda_g=540$ nm, $\Lambda_b=450$ nm for red, green and blue lights, respectively. Furthermore, based on the inherent

optical properties, we can calculate the attenuation ratios $\lambda_{c'}$ between green-red, blue-red color channels [112]:

$$\lambda_{c'} = \frac{\beta_{c'}}{\beta_r} = \frac{s_{c'}}{s_r} \cdot \frac{B_r}{B_{c'}}, c' \in \{g, b\}. \quad (3.6)$$

It can be observed that the reliability of λ_g and λ_b depends on the accuracy of the estimated waterlight B . The ratios are key parameters for adaptively modeling water types.

From the above analysis, we can get a useful conclusion, i.e., only one color channel's transmission should be estimated, and the rest of color channel's transmission is obtained by the color channel relationship (Eq. (3.4)) and the ratios (Eq. (3.6)). We will apply this in Section 3.2.2.

3.2.1.2 Adaptive Attenuation-curve Prior

The non-local haze-line prior in [15] that describes the difference between the clear image J and the observed image I . However, in the underwater environment, the change is much more complicated to deal with. The reason is that the particles in the water lead to wavelength-dependent attenuation coefficients. According to Eq. (3.2), the transmission $t_c(x)$ depends on the distance and wavelength. We propose a more general prior, namely adaptive attenuation-curve prior (AACP), for various imaging environments, like haze, fog and water.

We define that pixels with similar colors belong to the same cluster in a clear image. For the clear natural outdoor image, the work in [15] has verified that several hundred color clusters can represent all colors of the image with the acceptable result on the BSDS300 dataset. Specifically, they used K-means clustering to find 500 clusters for each image and replace every pixel value with its corresponding cluster center. The generated images had satisfactory PSNR from 36.6dB to 52.6dB. The observation is also valid for underwater images.

In the underwater environment, there are two factors affecting the difference between clear image and the observed image: the varying scene depth $d(x)$ from the camera to objects, which leads to the difference of transmission $t(x)$ for every pixel, and the wavelength-dependent attenuation coefficient β_c , which results in the difference of RGB channels in transmission, generating $t_c(x)$. When a clear scene is distorted by

water, the scene radiance $J_c(x)$ from the same cluster (with similar original colors) is attenuated in varying degrees based on Eq. (3.1), thereby generating different degraded colors $I_c(x)$ in the observed image. We depict every degraded pixel value of four clusters in RGB space, as shown in Fig. 3.5(d). Due to pixels' varying transmissions $t_c(x)$ in RGB channels, the pixel values are degraded and form a curve, which starts from the original color J (if $t = 1$) towards the waterlight color B (if $t = 0$). Therefore, we name it as an attenuation curve.

FIGURE 3.5: Attenuation-curve prior to model the degradation process of a clear image in water. (a) Four clusters' pixels generated by k-means in a clear image. (b) Corresponding pixel values of the clusters in RGB space. (c) The degraded image by water with the same clusters as (a). (d) Corresponding degraded pixel values in RGB space, forming attenuation curves.

To visualize the attenuation-curve prior, we conduct a underwater image simulation, as shown in Fig. 3.5. The Fig. 3.5(a) shows that pixels of a clear image are clustered into several hundred clusters by k-means. We only depict four clusters marked by four colors for simple illustration. In RGB space, four clusters center on four different locations as shown in Fig. 3.5(b), in which color markers correspond to those of Fig. 3.5(a). We synthesize an underwater image using Eq. (3.1) and Eq. (3.4) by setting $B = [0.16, 0.66, 0.58]$, $\lambda_g = 0.34$ and $\lambda_b = 0.27$, and the ground-truth t_r from [52]. Fig. 3.5(c) shows the synthetic underwater image with the same clusters as Fig. 3.5(a). In each cluster, their degraded colors are different from original ones because of the varying scene depth and different attenuation coefficients in RGB channels. The new distribution of clusters can be seen in Fig.

3.5(d). Each cluster contains different degraded colors, forming a curve in RGB space.

The prior can be explained by substituting Eq. (3.4) into Eq. (3.1):

$$(I_{c'} - B_{c'}) = (I_r - B_r)^{\lambda_{c'}} \cdot \frac{J_{c'} - B_{c'}}{(J_r - B_r)^{\lambda_{c'}}}, \quad c' \in \{g, b\}. \quad (3.7)$$

If we define the RGB coordinate space with the waterlight as the origin, the intensity of the green or blue channel ($I_{c'} - B_{c'}$) of the observed pixels has a power-function relationship with that of the red channel ($I_r - B_r$) with a power factor of $\lambda_{c'}$. Because the values of B and $\lambda_{c'}$ are the constants for a certain underwater scene, the curve uniquely relies on the scene radiance J . A different J corresponds to a different curve. So, the formation of attenuation curves can model the degeneration process for underwater images and adapt to different water types if we obtain the attenuation ratios between color channels according to Eq. (3.6).

Following Eq. (3.3), it can also guarantee the power-function relationship:

$$(I_{c'} - B_{c'}) = (I_r - B_r)^{\lambda_{c'}} \cdot \frac{K_{c'}(J_c - B_{c'})}{[K_r(J_r - B_r)]^{\lambda_{c'}}}, \quad c' \in \{g, b\}. \quad (3.8)$$

We will use the prior to cluster pixel values into curves and estimate pixel-wise transmission for each curve in the following section.

3.2.2 Underwater Image Restoration

The flowchart of the proposed method is illustrated in Fig. 3.6. It consists of two main components: waterlight estimation and transmission estimation. First, we estimate the waterlight using the smoothness and wavelength-dependent light attenuation properties. Then, we calculate the attenuation ratios between color channels. For transmission estimation, based on AACP, we estimate the initial relative transmission for red channel and refine it. Then, we calculate the attenuation factor and adjust the transmission for the three color channels by the saturation constraints. Finally, the image can be restored by the estimated waterlight and transmission. We call this method as AACP-based underwater image restoration.

FIGURE 3.6: Flowchart of the AACP-based underwater image restoration.

3.2.2.1 Waterlight Estimation

The waterlight B can be referred to as the homogeneous background light, arising from backscattering. Generally, it can be estimated from the pixels belonging to background regions and having a low variance. To estimate B , we first empirically partition the degraded image into overlapping patches, and then choose the patches whose total variation (TV, which calculated by $\text{TV}(P) = \sum_x \|\nabla P(x)\|_1$, where P stands for image patch) is less than a predefined threshold.

Within the smooth patches, we calculate the differences between the intensity of the red channel and the green or blue channels:

$$D_{c'}(x) = I_{c'}(x) - I_r(x), \quad c' \in \{g, b\}, \quad (3.9)$$

where $x \in S$, $S = \{\text{all pixels in the chosen patches}\}$. Because the red light attenuates more rapidly than the green and blue light in the water, those differences for the waterlight are larger. Thus, if R-G channel and R-B channel of a pixel have considerable differences, the pixel is most likely to be a candidate of the waterlight.

To find the pixel, we use the index function $\text{Ind}(\cdot)$ to sort the differences D_g and D_b in descending order (e.g., $\text{Ind}_{c'}(1)$ are coordinates of the pixel with the maximum value in $D_{c'}(x)$):

$$\text{Ind}_{c'}(n) = \text{sort}(D_{c'}(x)), \quad c' \in \{g, b\}, \quad (3.10)$$

where $\text{Ind}_{c'}(n)$ indicates the indexes (coordinates). We calculate the intersection by $Z = \text{Ind}_g(n \leq N) \cap \text{Ind}_b(n \leq N)$, obtaining the indexes of the largest values in both D_g and D_b , where N is a parameter to control the number of the intersection's elements. We consider the pixel with the maximum value of the red channel inside Z as the waterlight, which can be expressed as:

$$B_c = I_c \left(\arg \max_{x \in Z} I_r(x) \right), \quad c \in \{r, g, b\}. \quad (3.11)$$

The small red rectangle in Fig. 3.6 shows the selected waterlight.

3.2.2.2 Transmission Estimation

According to Eq. (3.3), the transmission estimation is composed of two parts: relative transmission t_c^n estimation and attenuation factor K_c estimation. The first part is to estimate the relative transmission for the distance between the farthest scene point and the nearest scene point, while the second part is to estimate the attenuation from the nearest scene point to the camera. We divide the transmission estimation into five steps: finding attenuation-curves, estimating an initial relative transmission t_c^n , refining the transmission, estimating attenuation factor K_c , adjusting the transmission of three color channels.

Finding Attenuation-Curves: According to the AACP, we need to cluster the pixels of the image into the attenuation-curves in RGB space as shown in Fig. 3.7 (where four of them are depicted). However, it is challenging and inefficient to cluster pixels into curves by their RGB coordinates directly. To facilitate this task, we calculate logarithm on both sides of Eq. (3.8):

$$\ln |I_{c'} - B_{c'}| = \lambda_{c'} \ln |I_r - B_r| + \ln \frac{K_{c'} |J_{c'} - B_{c'}|}{(K_r |J_r - B_r|)^{\lambda_{c'}}}, \quad c' \in \{g, b\}. \quad (3.12)$$

Compared to the power-function relationship of Eq. (3.8), $\ln |I_{c'} - B_{c'}|$ and $\ln |I_r - B_r|$ follow the linear relationship in RGB space. The straight lines have the same slope $\lambda_{c'}$ but different intercepts (the values are the second term of Eq. (3.12)). The scene radiance J determines the intercepts. Instead of clustering the pixels into curves, we can group them into the lines. Four curves turn into the lines using Eq. (3.12) as shown in Fig. 3.7(b).

FIGURE 3.7: Finding attenuation-curves. (a) Power-function relationship between color channels. (b) Linear relationship. (c) Rotated coordinates from (b) for clustering.

To speed up the clustering process, we rotate the 3D coordinate system to make the R-axis parallel to the direction of the lines. In the new coordinate system, it is efficient to cluster the pixels by building a KD-Tree from the predefined tessellation in the rotated G-B plane and querying the tree by their rotated G and B coordinates. Fig. 3.7(c) shows the four clusters in the rotated RGB system. Finding attenuation-curves only need pixels' rotated G and B coordinates.

Estimating an Initial Relative Transmission t_c^n : After finding the attenuation-curves, we estimate the initial relative transmission for the pixels in each curve. We reformulate the model (Eq. (3.1) with Eq. (3.3)) as:

$$|I_c(x) - B_c| = t_c^n(x) \cdot K_c |J_c(x) - B_c|, \quad x \in L, \quad c \in \{r, g, b\}, \quad (3.13)$$

where L is a curve, so $x \in L$ stands for the pixels belonging to the curve. For each curve, the maximum value of $|I_c(x) - B_c|$ is close to $K_c |J_c(x) - B_c|$ when $t_c^n(x) \rightarrow 1$. We consider the maximum value as the estimation of $K_c |J_c(x) - B_c|$ and obtain pixel-wise transmission $\tilde{t}_c^n(x)$ for each curve by:

$$\tilde{t}_c^n(x) = |I_c(x) - B_c| / \max_{x \in L} |I_c(x) - B_c|, \quad c \in \{r, g, b\}. \quad (3.14)$$

The maximum value of $|I_c - B_c|$ is closer to $K_c |J_c - B_c|$ ($K_c < 1$) than $|J_c - B_c|$ used in [12]. This is because scene radiance is first degraded by K_c and then the relative transmission.

According to Eq. (3.6), λ_g and λ_b can be obtained such that we only need to estimate the transmission of one color channel, and the transmission of the green and blue channel can be calculated by Eq. (3.4). Here we calculate the transmission of the red channel $\tilde{t}_r^n(x)$.

Refining the Relative Transmission: Although the above method can obtain the initial transmission, the assumption we use to estimate the value of $K_c |J_c - B_c|$ may be wrong if the curve is very short and close to the waterlight, such as the distant background area of the initial transmission in Fig. 3.6. Based on smoothness observation of the depth, the transmission map should be patch-wise smooth apart from the depth discontinuities [15]. The transmission is refined by applying the WLS filter [15, 53]:

$$\min_{\hat{t}_r^n(x)} \sum_x \frac{[\hat{t}_r^n(x) - \tilde{t}_r^n(x)]^2}{\sigma_r^2(x)} + \lambda \sum_x \sum_{y \in N_x} \frac{[\hat{t}_r^n(x) - \hat{t}_r^n(y)]^2}{\|I(x) - I(y)\|^2}, \quad (3.15)$$

where $\hat{t}_r^n(x)$ is the refined transmission, λ is a regularization parameter to adjust the data and the smoothness terms, N_x is the four nearest neighbors of x , and $\sigma_r(x)$ is a parameter to measure the reliability of the assumption for $\tilde{t}_r^n(x)$.

Two factors determine $\sigma_r(x)$, i.e., the standard deviation and the number of pixels in a curve. First, we consider the standard deviation $\text{std}_{x \in L}(I_r(x))$ for the red channel of pixels located on the curve L . We normalize the standard deviations by the maximum standard deviation, and get: $\widetilde{\text{std}}_{x \in L}(I_r(x))$. The higher deviation means that the assumption for estimating $\tilde{t}_r^n(x)$ is more reliable. Second, we take into account the number of pixels in a curve, i.e., $n(L)$. More pixels in the curve imply that the assumption for estimating $\tilde{t}_r^n(x)$ is more reliable. We define the

count reliability for a curve as:

$$\text{count_reliability}(L) = \min(1, n(L)/50). \quad (3.16)$$

By combining those two factors, $\sigma_r(x)$ for a curve L can be obtained by:

$$\frac{1}{\sigma_r(x)} \propto \widetilde{\text{std}}_{x \in L}(I_r(x)) \cdot \text{count_reliability}(L). \quad (3.17)$$

The larger value of $1/\sigma_r(x)$ makes $\tilde{t}_r^n(x)$ more reliable such that the first term of Eq. (3.15) will make a significant contribution to the initial relative transmission refining. That is, the refined transmission is close to the initial transmission ($\hat{t}_r^n(x) \rightarrow \tilde{t}_r^n(x)$), rather than being smoothed by the neighboring transmission (the second term of Eq. (3.15)).

Estimating Attenuation Factor K_c : Based on the observation of outdoor clear images, we assume that there must be the brightest or darkest pixel α in one of three color channels ($J_c(x=\alpha) = 1$ or 0). Formally, there exists a pixel α satisfying the equation:

$$\max_{c \in \{r, g, b\}} \frac{|J_c(\alpha) - B_c|}{\max(B_c, 1 - B_c)} = 1. \quad (3.18)$$

Similar to Eq. (3.14), $K_c|J_c(\alpha) - B_c|$ can be estimated by $\max |I_c(x) - B_c|$. We simplify three attenuation factors with one factor ($K_c = K$). So, we substitute $K|J_c(\alpha) - B_c| = \max |I_c(x) - B_c|$ into Eq. (3.18) and get:

$$K = \max_{c \in \{r, g, b\}} \frac{\max |I_c(x) - B_c|}{\max(B_c, 1 - B_c)}. \quad (3.19)$$

After getting the relative transmission $\hat{t}_r^n(x)$ and the attenuation factor K , we can recover the transmission $\hat{t}_r(x)$ by Eq. (3.3).

Adjusting the Transmission of Three Color Channels: Via the transmission of the red channel, we generate the transmission of the other channels by Eq. (3.4) and Eq. (3.6). Although Eq. (3.6) provides the relation of the three color channels, the reliability of the attenuation ratios relies on accurate estimation of B . Thus, we cannot get the accurate relationships only from Eq. (3.6). For instance, a small λ_g makes the range of t_g small, which reduces the contrast of the green channel of the image. Conversely, the large λ_g makes the range of t_g close to t_r (with the large range), which produces dark or saturated recovered pixels. Therefore, we

reformulate the ratios as $\lambda'_c, c \in \{r, g, b\}$ (λ'_c represents the self-adjust ratio) to adjust the transmission. According to Eq. (3.1) and Eq. (3.3), there is a saturation constraint for the restored image:

$$0 \leq \frac{I_c(x) - B_c}{K \cdot \hat{t}_r^n(x)^{\lambda'_c}} + B_c \leq 1, \quad c \in \{r, g, b\}. \quad (3.20)$$

Then, we get:

$$\begin{cases} \lambda'_c \geq \ln(\frac{I_c(x) - B_c}{K(1 - B_c)}) / \ln(\hat{t}_r^n(x)) + \varepsilon_c & \text{if } I_c(x) > B_c \\ \lambda'_c \geq \ln(\frac{B_c - I_c(x)}{K \cdot B_c}) / \ln(\hat{t}_r^n(x)) + \varepsilon_c & \text{if } I_c(x) < B_c \end{cases}, \quad (3.21)$$

where we limit λ'_c in the range $[\lambda_c, 1]$, and set $\lambda_r = \max(\lambda_g, \lambda_b)$. Since Eq. (3.21) is a strict constraint, we use a tolerance of ε_c to ensure high contrast. Here we choose ε_c empirically that makes 8% of recovered pixels exceed the limitations at each color channel. We use the new ratios to adjust the transmission for each color channel (see Eq. (3.22)).

3.2.2.3 Scene Radiance Restoration

Once the waterlight and the final transmission are estimated, the scene radiance J can be restored according to Eq. (3.1), (3.3), (3.4), and λ'_c :

$$J_c(x) = \frac{I_c(x) - B_c}{\max(K \cdot \hat{t}_r^n(x)^{\lambda'_c}, t_0)} + B_c, \quad c \in \{r, g, b\}. \quad (3.22)$$

where $t_0 = 0.1$ [14]. Because some recovered images look dim, we use a global linear contrast stretch (clipping 0.5% the darkest and brightest pixels) for display only in qualitative comparison, following the procedure of [15]. We will conduct the experiments to evaluate our method in Section 3.4.1.

3.3 Ambient Light Estimation with Deep Learning

In this section, we introduce a property of ambient light, named airlight. With the property, we propose a deep learning-based framework to estimate the airlight. To

train the framework, we provide a way to prepare the training examples.

The assumption is that airlight exists in the distant region of hazy images. Formally, according to Eq. (2.1), the distant region with high $d(x)$ results in low transmission ($t(x) \rightarrow 0$), so that pixel values in that region is close to airlight ($I(x) = A$). Here we define the airlight as A , compared with the waterlight B in the previous section, though both of them belong to ambient light. With this property, if we find the distant region of hazy images, we can obtain the airlight.

3.3.1 Airlight Estimation Framework

We propose a novel airlight estimation framework as shown in Fig. 3.8. It is composed of two key steps: U-Net and Weighted Sum. U-Net aims to distinguish the distant and non-distant region of images. Weighted Sum can calculate airlight value inside the region, in which the weights depend on the classification probability of each pixel. Since the framework is based on distant region segmentation (DRS), we term it as DRS-based airlight estimation method.

FIGURE 3.8: Airlight estimation framework (DRS-based airlight estimation method). The numbers below U-Net indicate the number of the feature maps.

U-Net. The airlight information exists in the distant region of hazy images. The main task of U-Net is to perform distant region segmentation, producing a pixel-wise probability map. The higher probability means that the corresponding pixel is more likely to be in the distant region. The binary mask can be generated by a confidence threshold of 0.5. If the probability is larger than or equal to the threshold, the pixel value in the mask is 1; otherwise, the pixel value is 0. Thus, the intensity 1 stands for the distant pixel, while 0 stands for the non-distant pixel. In Fig. 3.8, the white color means high value, and the black color means low value for the probability map and binary mask.

For the U-Net architecture [60], we adopt a 19-layer convolutional neural network, which is an encoder and decoder structure. The basic block is composed of two 3×3 convolutions, each of which is followed by a rectified linear unit (ReLU). The encoder consists of five basic blocks, each of which is followed by a max pooling with the stride of 2. The decoder consists of four symmetric basic blocks and four upsample operations. There are four skip connections concatenating four pairs of symmetric blocks, which combines low-level spatial information with high-level context information. The last layer is the 1×1 convolution followed by a sigmoid function for binary classification.

Weighted Sum. In this step, we first perform an element-wise multiplication between the original hazy image I and the binary mask, which detaches the distant region $D = \{(\alpha) | \alpha \in D\}$, where α is the pixel coordinates in the region D . With the probability map, we know the probability $p(\alpha)$ of each pixel. High probabilities correspond to high weights. We calculate the normalized weights by:

$$w(\alpha) = \frac{p(\alpha)}{\sum_{\alpha \in D} p(\alpha)}. \quad (3.23)$$

Hence, the airlight A can be expressed as:

$$A_c = \sum_{\alpha \in D} I_c(\alpha) \cdot w(\alpha), c \in \{r, g, b\}. \quad (3.24)$$

Reference [17] also uses the U-Net structure to estimate the airlight. It predicts a 2D airlight map with the same size of the input image. The airlight is obtained by averaging the map within each color channel. However, this method ignores original information of the hazy image and only uses the output of U-Net to regress airlight values. In addition, all output values of U-Net are of equal importance because of the average over the whole airlight map. Different from [17], the proposed approach first finds the distant region, and then estimate the airlight by a combination between the original color of the hazy image and the normalized weight. We will compare this two methods in Section 3.4.2.

3.3.2 Training Data Preparation

Although deep learning-based methods can achieve high performance in many applications, one tough issue for supervised learning is to collect training examples. For the haze removal field, it is difficult to capture pairs of hazy image and clear image. Some recent learning-based methods [17, 64] used the NYU-depth2 [42] dataset to synthesize training examples via Eq. (2.1). If we have a clear image and corresponding depth, we can generate a transmission map and hazy image by setting a certain scattering coefficient β and airlight A . However, NYU-depth2 is the indoor dataset which shows a different distribution compared to outdoor images and does not contain large depth for the real haze environment. Outdoor datasets have the problem of the inaccurate depth with noise.

To solve these problems when training U-Net, we propose a superpixel-based method to synthesize outdoor training examples, i.e., pairs of hazy image and binary mask. We adopt 550 clear images from Cityscapes [113] and the corresponding 1650 transmission maps from Foggy Cityscapes-refined [41] (in which three scattering coefficients are set for each clear image). Three scattering coefficients (β) are 0.005, 0.01 and 0.02, corresponding to maximum visibility distances of 600, 300, and 150 meters, respectively. Thicker haze means larger scattering coefficient such that the maximum visibility distance is shorter. By using these data, a pair of the hazy image and binary mask can be synthesized as follows.

- **Hazy image:** With a clear image, randomly generated airlight, and transmission map (corresponding to Fig. 3.9(a), (b), and (d), respectively), we can obtain a hazy image (Fig. 3.9(c)) via Eq. (2.1).
- **Binary mask:** A transmission map (Fig. 3.9(d)) is segmented into about 200 superpixels by using the SLIC [114] algorithm, as shown in Fig. 3.9(e). The mean transmission $\bar{t}(\theta)$ is computed for each superpixel, where θ is the pixel coordinates within the superpixel. We define the distant region by the mean transmission less than 0.1. If $\bar{t}(\theta)$ is less than 0.1, the mask value is 1; otherwise, the mask value is 0. Hence, we can obtain a binary mask as shown in Fig. 3.9(f).

The 550 images are divided into non-overlapping 498 training images and 52 testing images. For the training set, each image has three transmission maps with different scattering, and each transmission map assembles four randomly generated airlight.

FIGURE 3.9: Synthesis of pair of the hazy image and binary mask. (a) ~ (f) corresponds to the clear image, airlight, hazy image, transmission map, superpixels of transmission map, and binary mask, respectively.

Hence, there are 5976 ($498 \times 3 \times 4$) training samples. For the test set, each image has three transmission maps, and each transmission map assembles one randomly generated airlight. So, there are 156 (52×3) testing samples. We will conduct the experiments to evaluate our method in Section 3.4.2.

3.4 Experiments

In this section, we first compare our underwater image restoration method with state-of-the-art methods. Then, we evaluate the proposed ambient light estimation method.

3.4.1 Evaluation of Underwater Image Restoration

We compare our AACP-based underwater image restoration method against state-of-the-art IFM-based methods: MIP-based [9], DCP-based (UDCP [10]), blurriness and light absorption-based (BLA-based) [11], and fixed attenuation-curve prior-based (FACP-based, our previous work) [12] methods. For a fair assessment, all of the compared methods use codes provided by their authors and set default parameters. We evaluate our proposed method in three ways:

- Qualitative comparison,
- Quantitative no-reference comparison, and
- Complexity analysis.

For the underwater testing images, we use 20 images with different light conditions and water tones from [9, 43, 49, 105, 106, 115], as shown in Fig. 3.10¹. We use the same parameters in our method for all of the testing images. For waterlight estimation, we use 6-by-6 patches and set TV threshold as 0.9. N is a parameter that makes the intersection's elements larger than 1. In order to find the curves, we set the number of clusters equal to 900 for the underwater images, and 3600 for the synthetic images. In Eq. (3.15), we set $\lambda=0.05$ and scale $1/\sigma_r^2(x)$ to be in the range $[0, 1]$ as [15].

FIGURE 3.10: Sample testing images, from left to right and top to bottom, corresponding to "Image-1" to "Image-20", respectively.

3.4.1.1 Qualitative Assessment

In the subjective visual comparison, we choose four representative underwater images (Image-1, 5, 10, 12), as shown in Fig. 3.11, and use jet colormap to display the estimated transmission of the image. For those methods producing three transmission maps, we show the green channel's transmission. In addition, we also show results of the estimated waterlight by each method. MIP [9] estimates the transmission firstly and finds a pixel with the minimum transmission as the waterlight. UDCP [10] estimates the waterlight via finding the brightest pixel in the underwater dark channel. BLA [11] proposed a more reliable waterlight estimation method based on image blurriness and variance. Unlike them, we estimate the waterlight based on the observation of light attenuation in the water and low variance of the background.

In Fig. 3.11, the estimated waterlights are presented in the first row, and the corresponding restored images and the estimated transmission are illustrated in the second and third rows. From the figure, we can observe that our method cannot only enhance the contrast but also remove the color distortion of underwater images

¹The experiment for all 37 testing images (including 17 images that are shown in Fig. 2.4 from [9, 43, 49, 105, 106]) has been described in detail in Part A of the supplemental material. See the link: <https://personal.ntu.edu.sg/elpchau/underwater.zip>

FIGURE 3.11: Visual comparison of four underwater images with different scenes. (a) The original images. The restored results with the corresponding waterlight and transmission yielded by: (b) MIP [9], (c) UDCP [10], (d) BLA [11], (e) Our FACP [12], (f) Our AACCP.

(see Image-1, Image-5 and Image-10). The proposed AACCP method can recover more red color than our previous work (FACP [12]) because of the attenuation factor estimated in AACCP. Moreover, with the saturation constrain to adjust the transmission, our AACCP reduces the noise when increasing the contrast (see the distant coral reefs in Image-10). More qualitative comparisons are shown in Fig.

3.12, which further validates that the proposed method can obtain better results than state-of-the-art methods.

FIGURE 3.12: Comparisons of visual results by different methods. Row 1 is the input images; row 2 is the results of MIP [9]; row 3 is the results of UDCP [10]; row 4 is the results of BLA [11]; row 5 is the results of our FACP [12]; row 6 is the results of our AACP.

Although our approach is built on IFM and we compare it with the IFM-based methods, we still intend to compare it with the ACE [13] method which is a non-IFM image enhancement technique. The experimental results are shown in Fig. 3.13, illustrating that ACE produces an inconsistent color correction for Image-5 and Image-10, especially for distant coral reefs. The proposed AACP produces high-contrast and consistent-color images compared with ACE, which shows the effectiveness of our method.

3.4.1.2 Quantitative Assessment

In addition to the subjective visual comparison, the underwater image quality metrics (UIQM) [11, 116] measurement is utilized to evaluate the performance of underwater image restoration. The UIQM measurement is composed of underwater image colorfulness measure (UICM), underwater image sharpness measure (UISM), underwater image contrast measure (UIConM). UIQM is the linear combination of

FIGURE 3.13: Comparison with the non-IFM image enhancement method. (a) The input images; (b) the results of ACE [13]; (c) the results of our AACP. Row 1, 2, 3, and 4 correspond to Image-1, 5, 10, and 12, respectively.

the color, sharpness and contrast measures. UIQM is calculated as follows:

$$\text{UIQM} = c_1 \cdot \text{UICM} + c_2 \cdot \text{UISM} + c_3 \cdot \text{UIConM}, \quad (3.25)$$

where the default coefficients $c_1 = 0.0282$, $c_2 = 0.2953$ and $c_3 = 3.5753$.

The higher value of the UIQM stands for the better performance. In the experiments, we set the parameters $\alpha_L = \alpha_R = 0.1$ when calculating UICM. The patch size is set to 8×8 and the coefficients $\delta_1 = 0.299$, $\delta_2 = 0.587$ and $\delta_3 = 0.114$ in UISM. For the sample testing images, Table 3.1 shows the UIQM values of the restored images by different methods. It demonstrates our method performs better in most cases. For Image-2, 8, 13, 20 out of 20 images, BLA [11] obtains better results. For those images, the waterlight estimates of our method are relatively inaccurate compared with BLA [11] because an additional lamp illuminates the nearby rocks (see Fig. 3.10). The waterlight estimation in the non-uniform illumination will be our future work.

TABLE 3.1: UIQM values among restoration methods.

# Image	UIQM				
	MIP [9]	UDCP [10]	BLA [11]	FACP [12]	AACP (ours)
1	1.1112	2.5157	2.3295	3.175	3.5887
2	4.5072	3.9947	4.7803	4.2967	4.5907
3	5.033	5.0559	4.4216	5.3502	5.2595
4	2.5981	4.2301	3.7809	4.1648	4.3133
5	4.0478	4.3145	4.2518	4.785	4.7567
6	0.6235	3.4309	2.7203	3.6758	3.9611
7	2.3868	2.1942	2.6519	2.698	2.9193
8	4.0743	4.2122	4.3741	3.8583	3.9894
9	3.2611	2.2732	3.6926	3.718	3.6461
10	4.7316	4.5172	4.5156	5.0294	5.0132
11	0.6973	3.3234	3.3979	2.5245	3.8666
12	2.4927	3.0643	2.7725	4.1311	4.0734
13	4.0572	4.4427	4.9488	4.8074	4.8536
14	3.2528	3.0608	2.5855	3.0244	3.3188
15	2.8381	3.1636	2.8087	3.096	3.4492
16	4.2071	4.1544	4.5712	4.5607	4.5946
17	1.8945	2.8919	2.2753	4.2789	4.4814
18	4.5493	3.7596	4.5628	4.9285	4.8462
19	2.5251	2.7899	3.0232	3.3851	3.4149
20	4.8277	3.7059	5.0385	4.3988	4.779
Average	3.1858	3.5548	3.6752	3.9943	4.1858

3.4.1.3 Complexity Analysis

Our underwater restoration approach is non-local pixel based, which is more efficient than those patch-based methods. Because we estimate pixel-wise transmission for each curve, given an image with the size of $m \times n$, the computational complexity of initial transmission estimation is $O(m \times n)$. According to the analysis of [15], the computational complexity of refining transmission is also $O(m \times n)$. Besides, KD-Tree is fast when we use predefined tessellation to cluster pixels. As shown in Table 3.2, we present the computation time in comparison to MIP [9], UDCP [10], and BLA [11] on MATLAB R2015b with a CPU (Intel Xeon E5-1630 v3, 3.7 GHz). The results show our method is much faster than the others at different image resolutions.

TABLE 3.2: Computation time of image restoration methods.

Image Resolution	[9]	[10]	[11]	Ours
512 × 384	431.19 s	3.08 s	22.38 s	1.78 s
640 × 480	881.83 s	4.80 s	34.98 s	2.18 s
960 × 720	3714.58 s	10.50 s	81.72 s	5.07 s
1024 × 768	4605.45 s	12.65 s	90.69 s	5.52 s

3.4.1.4 Other Applications in the Atmosphere

The proposed methods can also be extended to the degraded images captured in the atmosphere, i.e., hazy images. Although AACP is proposed for underwater images, we would still like to compare our approach with state-of-the-art dehazing methods [14, 15] using a synthetic dataset [52]. For hazy images, there is an assumption that the attenuation coefficients of color channels should be identical. Thus, we set $t_r = t_g = t_b$ in the model (Eq. (3.1)). Table 3.3 presents the L1 errors of the estimated transmission maps and the dehazed images (calculated on non-sky pixels). Our method outperforms the other methods in most cases².

TABLE 3.3: Comparison of L1 errors over synthetic hazy images. The table presents the L1 errors of the estimated transmission (left) and the dehazed images (right).

	He's [14] Dack Channel	Berman's [15] Haze-Line	Our AACP- based method
Church	0.070 / 0.048	0.047 / 0.032	0.066 / 0.047
Couch	0.069 / 0.041	0.051 / 0.031	0.037 / 0.022
Flower1	0.190 / 0.098	0.061 / 0.022	0.040 / 0.015
Flower2	0.203 / 0.094	0.115 / 0.045	0.079 / 0.032
Lawn1	0.118 / 0.063	0.032 / 0.026	0.042 / 0.031
Lawn2	0.115 / 0.066	0.041 / 0.031	0.059 / 0.042
Mansion	0.074 / 0.043	0.080 / 0.049	0.074 / 0.046
Moebius	0.235 / 0.146	0.153 / 0.081	0.088 / 0.042
Raindeer	0.127 / 0.068	0.089 / 0.045	0.088 / 0.042
Road1	0.097 / 0.051	0.058 / 0.040	0.064 / 0.045
Road2	0.087 / 0.056	0.062 / 0.042	0.062 / 0.041
Average	0.126 / 0.070	0.072 / 0.040	0.066 / 0.037

Fig. 3.14 presents the dehazing results of three images taken in the atmospheric environment. The image contrast in distant regions can be regarded as an important metric of visual comparison. Our method can recover the contrast of distant regions (see the red boxes of Fig. 3.14) compared with He's [14] and Berman's [15]. It shows that our method can adapt to image dehazing and obtain outstanding performance.

3.4.2 Evaluation of Ambient Light Estimation

Here we evaluate our DRS-based airlight estimation method on the synthetic images and natural images. In addition, we show the improvement of the AACP-based

²The corresponding estimated transmission maps and recovered images for Table 3.3 can be found in Part B of the supplemental material. See the link: <https://personal.ntu.edu.sg/elpchau/underwater.zip>

FIGURE 3.14: Dehazing results of three images taken in the atmospheric environment. Row 1 shows input hazy images. Row 2 to 4 corresponds to the restored images by He’s [14], Berman’s [15], and the proposed method, respectively.

underwater image restoration method with the estimated airlight by DRS. For quantitative measurement, we report L_2 error between the ground-truth airlight and estimated one. Following [1] and [16], we calculate mean L_2 error, median L_2 error, and error variance over the testing images.

3.4.2.1 Synthetic Images

We train our method on the training set of synthetic images and evaluate it on the test set. The quantitative comparisons are shown in Table 3.4. Our DRS-based method outperforms state-of-the-art airlight estimation methods with a significant margin. Fig. 3.15 demonstrates two synthetic images and the estimated airlight by different methods. The green regions in the images are the output masks of U-Net by our method. The estimated airlight is weighted summed over those regions.

TABLE 3.4: Quantitative comparisons on synthetic images.

Method	Mean L_2 error	Median L_2 error	Error variance
He [14]	0.216	0.198	0.028
Bahat [1]	0.176	0.106	0.031
Berman [16]	0.276	0.259	0.045
DRS (ours)	0.047	0.009	0.016

FIGURE 3.15: Examples of synthetic images. GT stands for ground-truth airlight. The airlight above the image is estimated by He *et al.* [14], Bahat and Irani [1], Berman *et al.* [16], and DRS (ours), respectively. The green regions are the output masks of U-Net by our method.

3.4.2.2 Natural Images

Following the same experiment of [1, 16], we use the 40 natural images [1] to evaluate the generalization ability of our approach. According to the infinite distance assumption, the authors manually marked the distant region of hazy images and obtained the ground-truth airlight by averaging colors over the distant region. The results are shown in Table 3.5. Although our method is trained with synthetic foggy images in urban street scenes, it can generalize to other scenes and achieve the best results at the mean L_2 error and median L_2 error compared with the other approaches. Although [17] also adopts U-Net structure, the results are the worst. We think the main problem is that they used U-Net for airlight regression without considering original color information of hazy images. Fig. 3.16 shows four natural images and the estimated airlight by the different methods.

TABLE 3.5: Quantitative comparisons on natural images.

Method	Mean L_2 error	Median L_2 error	Error variance
Zhang [17]	0.264	0.242	0.026
He [14]	0.129	0.109	0.013
Bahat [1]	0.111	0.086	0.013
Berman [16]	0.119	0.063	0.014
DRS (ours)	0.103	0.060	0.014

FIGURE 3.16: Examples of natural images. GT stands for ground-truth airlight. The airlight above the image is estimated by Zhang and Patel [17], He *et al.* [14], Bahat and Irani [1], Berman *et al.* [16], and DRS (ours), respectively. The green regions are the output masks of U-Net by our method.

3.4.2.3 Integration of Ambient Light Estimation into Underwater Image Restoration Method

We evaluate the AACP-based underwater image restoration method integrated with the DSR-based ambient light estimation. Specifically, we replaced the waterlight estimation module in AACP (see Section 3.2.2.1) by the DSR-based ambient light estimation (see Section 3.3.1). We conducted a quantitative comparison with a deep learning-based method, DCPDN [17], on three hazy images, as shown in Fig. 3.17. We can observe that our method can remove most of the haze of images than DCPDN. The restored images of DCPDN still contain an atmospheric veil spread over the images.

3.5 Chapter Summary

This chapter has introduced two dense prediction tasks, i.e., image restoration and ambient light estimation, and our corresponding solutions. We presented

FIGURE 3.17: Quantitative comparison between (b) our ACCP with DSR and (c) deep learning-based DCPDN [17]. (a) shows input hazy images.

a novel and effective underwater image restoration method to predict a dense transmission map based on the adaptive attenuation-curve prior. The qualitative results demonstrated that our proposed method improves the contrast, removes the color cast, and suppresses the noises. The quantitative results showed that our method outperforms state-of-the-art approaches by a significant margin. Meanwhile, according to ambient light property, we presented a deep learning-based framework to segment the distant regions and predict the ambient light in the regions. Our ambient light estimation method achieved superior performance on both synthetic and natural images. With the combination of these two methods, the restoration performance is further improved.

Chapter 4

Crowd Detection and Counting

In the previous chapter, the adaptive attenuation-curve-based method is proposed for image restoration. The restored images are pleasing for human eyes, but they do not provide a direct route for computers to understand the image’s contents. In this chapter, we study two fundamental computer vision problems, i.e., detection and counting, in crowd scenes. Crowd detection and counting are related and can be considered as a dense prediction. Crowd detection is composed of the object’s center point localization and size regression. First, we propose a novel self-training approach capable of training a typical object detector only with point-level annotations to yield bounding boxes. The count is obtained by filtering low-confidence detections. Second, we present a straightforward dense regression method for crowd counting without detecting objects. These two dense prediction methods are illustrated in Figure 4.1.

FIGURE 4.1: Crowd detection and counting methods. *Zoom in for better viewing.*

4.1 Introduction

With a massive amount of population living in cities, crowd scenes have become a fundamental yet challenging scenario in a wide variety of computer vision applications, such as video surveillance [65], crowd analysis [117, 118], and safety monitoring [119, 120]. Objects in dense crowds present small sizes, large-scale variation, and severe occlusion, which poses significant challenges to object detection methods that simultaneously predict objects’ locations and sizes in an image.

Modern crowd counting methods [18, 20, 82], named counting-by-regression methods, bypass the locations and sizes of objects but employ deep neural networks (DNNs) to regress a density map, which is further integrated to obtain the overall count. These methods have achieved outstanding counting performance in dense crowds. However, they only aim to count the number of objects and lose individual information, i.e., object instance’s location and size. We argue that the density map only contains weak information about the crowds. Locations and sizes of object instances provide much more important information for other computer vision applications, such as multi-object tracking [32, 121], face recognition [6, 122], and person re-identification [123].

It is very costly and laborious to collect object-level bounding box annotations usually needed for training DNN-based object detection methods, especially for images containing thousands of objects. Current crowd counting datasets provide only point-level annotations, and usually, human heads are labeled as the central points, e.g., the green dots shown in Fig. 4.2. Due to the lack of object sizes, state-of-the-art DNN-based object detectors [31] cannot be trivially applied to such point supervision. As pioneers, Liu *et al.* [19] introduced a pseudo size updating scheme in a detection network to estimate object sizes. Sam *et al.* [21] proposed an LSC-CNN to achieve higher detection performance in crowd scenes. However, these works are still not on par with box-supervised methods (e.g., Faster R-CNN [83]) in the detection task. As for the counting task, these methods, denoted as counting-by-detection methods, can count objects by filtering out low-confidence objects with a threshold. However, they also suffer from highly crowded objects.

Table 4.1 shows the crowd counting and detection tasks can be addressed by dense regression-based methods (i.e., density map regression) and dense classification-based methods (i.e., keypoint, blob, or box detection), respectively. Therefore, we

TABLE 4.1: Crowd counting and detection tasks addressed by dense regression-based methods and dense classification-based methods, respectively.

Method \ Task	Crowd counting	Crowd detection
Dense regression	Density map regression	Post-processing of density map
Dense classification	Thresholding of keypoint, blob, or box	Keypoint, blob, or box detection

consider the crowd counting and detection tasks as a dense prediction problem. The problem aims to predict a structural feature map with either continuous labels (i.e., the density map) or discrete labels (i.e., the heatmap with pixel-wise classification). In the meantime, the crowd counting and detection tasks are relevant because some post-processing techniques can be employed to obtain the corresponding detections and counts. For example, a simple local peak selection was proposed by Idrees *et al.* [90] to localize objects on the density map. A thresholding technique [21] was used to calculate the crowd count from the predicted detections.

In this chapter, we first propose a novel self-training approach to train a typical detection method only with point-level annotations such that it can accurately and simultaneously detect and count objects in dense crowds, termed Crowd-DCNet. Specifically, based on a keypoint-based detector, i.e., CSP [87], we decouple detection as the separate estimation of objects' center points and sizes. The available point-level annotations directly supervise the estimation of the center points during training. As the ground-truth object sizes are not accessible, we propose a simple yet effective assumption in crowd scenes, called locally-uniform distribution assumption (LUDA), to generate the initial pseudo size for each object (see the red bounding boxes shown in Fig. 4.2(a)). Meanwhile, we propose a crowdedness-aware loss to emphasize the contributions of crowded objects in object size regression (see the crowdedness at the top-left corner of red boxes in Fig. 4.2(a)). Moreover, we propose a confidence and order-aware refinement scheme to continuously update the pseudo sizes during training. It performs the refinement operation by considering both the prior confidences and the updating order of pseudo sizes, such that the detection ability of the detector is increasingly boosted (see the blue bounding boxes shown in Fig. 4.2(b) for the refined pseudo object sizes). Besides, to deal with highly dense crowds (e.g., one person represented by several pixels on the ShanghaiTech [7] dataset), we propose an effective decoding method to improve the representation ability of the output features, in which a feature fusion and decoding technique is employed to restore the full-resolution feature maps.

FIGURE 4.2: Illustration of generated training examples at different phases by our self-training approach. The images are from the WiderFace [6] dataset. The green dots shown at the centers of faces stand for the point-level annotations. (a): Before training, the pseudo object sizes (red boxes) were generated by the proposed locally-uniform distribution assumption. The numbers shown at the top-left corner of red boxes stand for object crowdedness. (b): After training, the refined pseudo object sizes (blue boxes) by our crowdedness-aware loss and the confidence and order-aware refinement scheme. The numbers shown at the top-right corner of blue boxes stand for the pseudo object sizes’ posterior probabilities. *Zoom in on the figure for better viewing.*

Second, bypassing object detection, we propose a compact CNN model for crowd counting, namely multi-scale density (MSD) regressor. Our MSD regressor can be decomposed into two coherent regression stages: coarse and fine density map estimation. These two stages are connected by employing the multi-task cascades [38]. The coarse density map can be considered as a rough estimate of the fine density map. The supervision of the coarse density map implemented in the bottleneck of the hourglass structure can partially provide the blocked information for gradient back-propagation. This compact architecture only has about 372k parameters. The experimental results demonstrate the outstanding performance of our light model on the UCSD [119] and TRANCOS [124] surveillance datasets, and our model can obtain comparable results on the densely crowded ShanghaiTech [7] dataset.

The rest of this chapter is organized as follows. In Section 4.2, we introduce the

self-training approach for Crowd-DCNet in detail. Then, the MSD regressor is presented in Section 4.3. Experimental results and ablation studies are provided in Section 4.4. We conclude this chapter in Section 4.5.

4.2 Crowd-DCNet for Crowd Detection and Counting

In this section, we introduce a CNN-based method for crowd detection and counting. Fig. 4.3 illustrates the proposed self-training framework, which is capable of training a typical object detector only with point-level annotations for simultaneous object detection and counting, namely Crowd-DCNet. To be specific, the framework is based on an anchor-free keypoint-based object detector, i.e., CSP detector [87]. A locally-uniform distribution assumption is proposed to generate the initial pseudo size for each object, and a crowdedness-aware loss is proposed to emphasize the pseudo sizes of crowded objects in size regression. Furthermore, a confidence and order-aware refinement scheme is proposed to update the pseudo sizes in each training iteration. In addition, a decoding method is proposed to handle dense crowds. In what follows, we present the detection network and the self-training approach in detail.

4.2.1 Crowd-DCNet

We employ the keypoint-based detection method because it allows us to estimate the center point and size of objects separately. Therefore, the center points could be directly supervised by the point-level annotation, while the size estimation is accomplished by the proposed modules. In addition, it is anchor box-free.

4.2.1.1 Architecture

For the datasets with medium-density crowds (e.g., WideFace [6], CARPK and PUCPR+ [8]), we employ the CSP in [87]. For the dataset with high-density crowds (e.g., ShanghaiTech [7] and NWPU-Crowd [2]), however, the original network performs poorly since the center point prediction fails to represent highly dense

FIGURE 4.3: Illustration of the proposed self-training framework for training a detector (Crowd-DCNet) only with point-wise supervision. The decoder produces high-resolution feature maps for the estimation of objects’ center points and sizes. Before training, the pseudo object sizes are first generated from the point-wise supervisory information, based on our locally-uniform distribution assumption. During training, the crowdedness-aware loss supervises the size regression between pseudo sizes and predicted sizes. Built by center points and sizes, the blue and red bounding boxes are the prediction and pseudo supervision, respectively. In every training iteration, our confidence and order-aware refinement scheme updates the red boxes with the corresponding blue boxes based on their confidence, as shown in the box’s top-right corner. Here an image with sparse objects is used for clear visualization purposes.

objects in its output feature map of size $\frac{H}{4} \times \frac{W}{4}$, where H and W are the height and width of the input image, respectively. Hence, we propose an effective decoding method to handle this issue, forming a detection network for crowd detection and counting, termed Crowd-DCNet.

As illustrated in Fig. 4.3, Crowd-DCNet adopts the five-stage ResNet-50 [22] as the backbone network, where each stage downsamples the feature maps by a factor of 2, except for Stage-5 that uses the dilated convolutions to keep the stride the same as Stage-4. Let $C_i, i \in \{1, 2, \dots, 5\}$ be the output feature maps of the i -th stage. In the *Decoder*, three Conv(1x1-s1-c256) are applied to reduce the number of channels of C_5 , C_4 , and C_3 , where Conv denotes the convolutional layer, and (1x1-s1-c256) means the layer with the kernel of 1x1, the stride of 1, and the channel of 256. Then, we employ a top-down feature fusion manner to merge C_5 , C_4 , and C_3 , generating the fused features P_3 . The fusion manners are shown as the dotted yellow and blue rectangle in Fig. 4.3. Finally, we use an $8 \times$ Up structure to decode the fused features, consisting a $2 \times$ nearest upsampling layer followed by two Deconv(4x4-s2)-Conv(3x3-s1)-BN-ReLU, where Deconv, BN, and ReLU denote the deconvolution, batch normalization, rectified linear unit, respectively.

The decoder produces the output feature maps (P_0) with the same size as the input, i.e., $H \times W$. There are two separate heads (i.e., $\text{Conv}(1 \times 1\text{-s1-c1})$) for center point prediction and size prediction, producing the center point heatmap and the size map.

4.2.1.2 Supervision information

Let $\{\mathbf{p}_j\}_{j=1}^M$ be the point-wise annotations, where $p_j := (x_j, y_j)$ is the 2D coordinates of the center of the j -th object in an image, and M is the total number of objects in an image. As shown in Fig. 4.3, to supervise the estimation of object center points, we generate a ground-truth center point heatmap $Q \in [0, 1]^{H \times W}$ with 1 for objects' center points and 0 for negative points. To decrease the ambiguity of negative points surrounding the positive ones, we place a normalized 2D Gaussian mask at the center location of each positive point, as performed in [87]. If two masks overlap, we choose the element-wise maximum for the overlapped region. For the object size supervision, we generate pseudo object sizes denoted by s_j , which will be introduced in Section 4.2.2. Here we assume that the objects (e.g., heads and faces) have an aspect ratio of 1 in crowded scenes. We assign $\log(s_j)$ to the j -th object's center coordinates (x_j, y_j) and zeros to other locations, generating the size map $S \in \mathbb{R}^{H \times W}$. For the original CSP with the output of size $\frac{H}{4} \times \frac{W}{4}$, an offset map is appended to estimate the discretization error caused by the stride of 4. The ground-truth offset for p_j is defined as $\frac{x_j}{4} - \lfloor \frac{x_j}{4} \rfloor$ and $\frac{y_j}{4} - \lfloor \frac{y_j}{4} \rfloor$ on the x -axis and y -axis, respectively, which is assigned to (x_j, y_j) on the offset map. $\lfloor r \rfloor : \mathbb{R} \rightarrow \mathbb{Z}$ of a real number r denotes the greatest integer less than or equal to r .

4.2.1.3 Loss function

We apply the focal cross-entropy loss [39, 87] to each pixel on the center point heatmap:

$$L_c = -\frac{1}{M} \sum_{j=1}^M \begin{cases} (1 - \hat{q}_j)^\gamma \log(\hat{q}_j), & \text{if } q = 1, \\ A(1 - q_j)^\delta (\hat{q}_j)^\gamma \log(1 - \hat{q}_j), & \text{otherwise,} \end{cases} \quad (4.1)$$

where $\hat{q}_j$ and q_j are the predicted probability and the ground truth, respectively; γ is the hyper-parameter of the focal loss [85] which is set to 2 in all experiments;

δ is the hyper-parameter to control the penalty of negatives, which is set to 4 in all experiments; A is the coefficient to address the imbalance between positive and negative points, which is set to 1 (resp. 1/16) for the original CSP (resp. the proposed decoding structure) in all experiments. Intuitively, if we enlarge the CSP’s output feature map by the decoding method from $\frac{H}{4} \times \frac{W}{4}$ to $H \times W$, the number of negative points will increase by 16 times. Hence, we set $A = 1/16$ to balance the positive and negative points. For object size regression, we propose a crowdedness-aware loss $L_{size-\alpha}$, of which the details will be described in Section 4.2.3. For offset estimation, the smooth L1 loss [67] is calculated between the ground-truth offsets and predicted ones, denoted as L_o . The overall training objective is

$$L = \lambda L_c + L_{size-\alpha} + L_o, \quad (4.2)$$

where λ is the weight for center point classification, which is experimentally set to 0.1.

4.2.1.4 Inference

In inference, the detector first performs a forward pass to generate the center point heatmap, the size map, and the offset map (if it is used). The peak points in the center point heatmap are extracted by a local maximum operation with the kernel of 3×3 . Then, the center points of objects are selected from those peak points whose probabilities are larger than a predefined confidence. The object sizes are obtained in the size map by the corresponding center points’ coordinates (see the “Prediction” in Fig. 4.3). If the offset map is appended, the corresponding offsets are added to object coordinates. Finally, the bounding boxes can be decoded by the center points’ coordinates and the corresponding sizes.

4.2.2 LUDA-based Pseudo Object Size Generation

As the ground-truth object sizes are unavailable, we generate pseudo object sizes from the point-wise supervisory information to train the detector. In crowded scenarios, object instances, e.g., heads, faces, or cars, are usually uniformly distributed in an image. According to the geometry-adaptive kernel [7], a typical object’s size is proportional to the distance to its nearest neighbors in dense crowds.

This assumption is relatively weak as the objects are not always dense enough in an image. To tackle this problem, we employ the non-uniform kernel [20] to locally restrict the above assumption and propose the locally-uniform distribution assumption (LUDA). That is, 1) the objects in crowd scenarios are uniformly distributed in a local region and have a similar size in that region; and 2) the crowdedness of the region affects the precision of size estimates. In what follows, we detail our pseudo object size generation method based on LUDA.

Following [7, 20], we first calculate the initial object size of point $\mathbf{p}_j$ according to the distances to its K nearest points, i.e.,

$$\bar{d}_j = \frac{1}{K} \sum_{k=1}^K \beta d_{j,k}, \quad (4.3)$$

where $\bar{d}_j$ is the initial object size of $\mathbf{p}_j$, $d_{j,k}$ is the distance between point $\mathbf{p}_j$ and its k -th nearest neighbor, and β is a scalar. The initial object size is further smoothed to reduce the variation in a local region, leading to the pseudo object size s_j :

$$s_j = \frac{1}{|R_{p_j}|} \sum_{l \in R_{p_j}} \bar{d}_l, \quad (4.4)$$

where R_{p_j} is the p_j -centered local region, $|R_{p_j}|$ is the number of the points inside R_{p_j} , and $\bar{d}_l$ is the initial size of the l -th point contained in R_{p_j} . We set the R_{p_j} to a circular region so that KD-Tree can be used to improve the calculation speed. $|R_{p_j}|$ affects the precision of size estimates, which will be exploited in the crowdedness-aware loss in Section 4.2.3.

Since the crowdedness differs from region to region, a single set of parameters in Eqs. (4.3) and (4.4) can only fit a specific crowdedness. In experiments, we choose multiple sets of parameters intuitively for different training sets such that the crowded objects have precise pseudo sizes. The detailed parameter settings will be explained in Section 4.4.1.1.

4.2.3 Crowdedness-aware Loss for Object Size Regression

With the pseudo object size s_j , a straightforward way to supervise the regression of object sizes is using the smooth L1 loss [67], i.e.,

$$L_s = \frac{1}{M} \sum_{j=1}^M \text{SmoothL1}(\hat{s}_j, s_j), \quad (4.5)$$

where $\hat{s}_j$ is the size prediction of the j -th object. However, there is an obvious drawback for Eq. (4.5), i.e., all object instances equally contribute to the loss, and if some pseudo object sizes are inaccurate (i.e., noisy supervision), the training of the detector will be adversely affected. Moreover, inaccurate pseudo sizes are inevitable since the simple LUDA cannot resolve the complexity of crowded scenes (see Fig. 4.2(a)). Such a drawback is experimentally verified in the following Table 4.4 (see the 5-th entry of Table 4.4).

To this end, we propose a crowdedness-aware loss associating each object instance with its crowdedness, which indicates the importance in object size regression. We formulate the crowdedness-aware loss as

$$L_{size-\alpha} = \frac{1}{M} \sum_{j=1}^M \alpha_j \text{SmoothL1}(\hat{s}_j, s_j), \quad (4.6)$$

where $\alpha_j \in [0, +\infty)$ is the crowdedness-aware factor controlling weight of the s_j . Based on LUDA, crowded objects can produce more accurate pseudo object sizes than sparse ones, and thus the crowded objects should be assigned with larger weights. Specifically, we define the α_j as an exponential function of the number of objects (crowdedness) inside the local region R_{p_j} , i.e.,

$$\alpha_j = (|R_{p_j}|)^\eta. \quad (4.7)$$

Here, we use a tunable parameter $\eta \geq 0$ to scale the factor. In practice, we use a threshold to limit the maximum value of α_j to avoid gradient explosion. By assigning crowdedness-aware factors to the pseudo sizes, the loss function emphasizes the influence of the crowded objects and weakens that of the sparse ones. The crowdedness-aware loss enhances the robustness of the training with the noisy pseudo sizes.

4.2.4 Confidence and Order-aware Refinement Scheme for Pseudo Size Updating

Although the crowdedness-aware loss is able to boost the robustness of the detector when trained with noisy pseudo sizes, the inaccurate sizes still exist and will affect the backward pass of training. Thus, we propose a confidence and order-aware refinement scheme to update pseudo object sizes for better training the detector. In [19], the pseudo bounding boxes are updated by selecting the predicted boxes with the highest scores among those whose sizes are smaller than the pseudo ones. However, such a criterion may not be true in practice since it ignores the following two key issues, resulting in the inaccurate refinement and unstable training process. 1) The prior information. It updates the pseudo sizes without considering their prior confidences. 2) The updating order. All pseudo sizes are treated identically, resulting in the synchronous updating of both accurate and inaccurate sizes.

Instead, by taking prior information into account, we assign every pseudo size with a prior probability at the beginning of training, and if and only if the predicted posterior probability of the detector is greater than the prior probability, we update the pseudo size (resp. prior probability) with the predicted size (resp. posterior probability) for the next epoch. Referring to the prior confidence, our refinement scheme guarantees the detector is trained with increasingly confident examples. Meanwhile, it can update the most inaccurate sizes first, then followed by updating the relatively accurate ones. This is achieved by setting the same prior probability for all pseudo sizes before training. During training, easy examples (e.g., sparse and large objects) with noisy size supervision that achieve high posterior probabilities rapidly are updated first. Then, hard examples (e.g., crowded and small objects) with more accurate size supervision are updated. In other words, the updating order is from noisy sizes to noiseless sizes. The proposed refinement scheme presents a more powerful error-correction capability than the one in [19]. See the experimental results in Table 4.4.

More specifically, let $b_j = \{p_j, s_j\}$ be a pseudo bounding box centered at p_j with the pseudo size s_j , and the prior probability $P(b_j)$ is assigned to b_j . The object detection problem is cast as learning the posterior $P(C, B|X)$, where X is the input image, B is the bounding boxes of objects, and $C \in \{0, 1\}$ is the binary class with 0 for background and 1 for object instance. Our refinement scheme, which is merged

Algorithm 1 Confidence and Order-aware Refinement Scheme.

Input: The i -th input image X_i with a set of pseudo bounding boxes $B_i = \{b_1, \dots, b_{M_i}\}$, and prior probabilities of the boxes $P(b_j), j \in \{1, \dots, M_i\}$.

Output: Refined pseudo bounding boxes $\tilde{B}_i = \{\tilde{b}_1, \dots, \tilde{b}_{M_i}\}$, and refined prior probabilities $P(\tilde{b}_j)$.

```

1: Forward passing the detector to generate the detections  $\hat{B}_i = \{\hat{b}_1, \dots, \hat{b}_{M_i}\}$  with the posterior
   probabilities  $P(C, \hat{B}_i | X_i)$ .
2: for  $j$  in  $\{1, \dots, M_i\}$  do
3:   if  $P(C, \hat{b}_j | X_i) > P(b_j)$  then
4:      $P(\tilde{b}_j) \leftarrow P(C, \hat{b}_j | X_i)$ 
5:      $\tilde{b}_j \leftarrow \hat{b}_j$ 
6:   else
7:      $P(\tilde{b}_j) \leftarrow P(b_j)$ 
8:      $\tilde{b}_j \leftarrow b_j$ 

```

into the training process, is summarized in Algorithm 1. In each training iteration, after executing a forward path, we obtain the predicted detections $\hat{B}_i$ with the posterior $P(C, \hat{B}_i | X_i)$ (i.e., Line 1). For each instance $j \in \{1, \dots, M_i\}$, if $P(C, \hat{b}_j | X_i)$ is larger than $P(b_j)$, we update the prior probabilities $P(\tilde{b}_j)$ with the posterior $P(C, \hat{b}_j | X_i)$, and update the pseudo bounding boxes $\tilde{b}_j$ with the prediction $\hat{b}_j$ (i.e., Lines 3-5). Otherwise, the prior probability and pseudo bounding box remain unchanged (i.e. Lines 7 and 8). A constant prior probability of 0.6 is assigned to all $\{B_i\}_i^N$ at the beginning of training, where N is the number of training images.

Remark. Here we provide more explanations of why our self-training approach enables the detector to update the pseudo size towards the correct direction. The main reasons come from two aspects. First, as analyzed in Section 4.2.2, our LUDA-based pseudo size generation method ensures that the pseudo sizes of crowded objects (i.e., hard examples) are more accurate than those of sparse objects. Also, it is known that hard examples have a heavy influence on the detector’s training than easy examples [125, 126]. Thus, when incorporated with the crowdedness-aware loss, the larger number of relatively accurate hard examples in crowded scenes dominate the detector, making it robust to the outliers (e.g., inaccurate pseudo sizes). Second, the pseudo sizes are updated in a robust and orderly manner by our refinement scheme. We set the same initial prior probability for all pseudo bounding boxes. The posterior probability of easy examples will first meet the initial prior probability during training, and thus Algorithm 1 begins with updating the pseudo sizes of inaccurate easy examples. With increasingly accurate easy examples, the detector becomes stronger such that hard examples are then refined.

More experimental results to illustrate the effectiveness of the proposed method can be found in Section 4.4.1.2.

4.3 MSD Regressor for Crowd Counting

In this section, we introduce the multi-scale density (MSD) regressor for crowd counting. Bypassing object detection, MSD can produce a density map, where the crowd count is obtained by summing all values over that map. The multi-scale structure can prevent gradient vanishing at the bottleneck of MSD during training and perform a coarse-to-fine prediction manner in inference. We will describe the proposed architecture, loss function, and ground truth generation, successively.

4.3.1 Architecture

Inspired by the cascaded CNNs in [38, 127], we propose a novel CNN-based MSD regressor as shown in Fig. 4.4, which is composed of a Feature-Shared Network and two regressors (i.e., Coarse-Scale Network and Fine-Scale Network).

FIGURE 4.4: Architecture of the proposed MSD regressor with a cascaded structure.

4.3.1.1 Feature-Shared Network

To extract features, we design a straight-forward CNN architecture, namely Feature-Shared Network. The Feature-Shared Network consists of four convolutional layers each followed by a parametric rectified linear unit (PReLU) as the activation function, and two max-pooling layers with a filter of size 2×2 and stride 2. As we use two pooling layers, the output feature maps are downsampled by a factor of 4.

The output features of this network will be shared by the following two regressors: Coarse- and Fine-Scale Networks.

4.3.1.2 Coarse-Scale Network

In Coarse-Scale Network, the coarse density map is estimated by taking the shared features as input. We use three convolutional layers with filters of sizes 5×5 , 5×5 and 1×1 , respectively. Each of the first two layers is followed by a PReLU, and the last layer is followed by a ReLU to ensure that the density map produces positive values. The output coarse density map has $1/4$ of the original input size. Furthermore, the features of the second convolutional layer will be fed into the subsequent Fine-Scale Network at the next stage. The features supervised by the coarse density map can provide rough locations and densities of objects of the image. In other words, the Coarse-Scale Network can learn a global understanding of scenes.

4.3.1.3 Fine-Scale Network

From the output features of the Feature-Shared Network, we set up another forward path to generate a fine-scale density map with the same size as the input image, namely Fine-Scale Network. First, two convolutional layers with the filter of size 5×5 are used to extract features. Then, the features of the second layer at this network are concatenated with features from Coarse-Scale Network. After that, two convolutional layers with the filter of size 3×3 are exploited to fuse the concatenated features. To generate a full-resolution density map, we design a simple yet powerful structure: upsampling network. It consists of a nearest-neighbor upsampling operation followed by a convolutional layer with a 3×3 kernel. We use two upsampling structures, so the feature maps are upsampled by a factor of 4. At the end of this stage, we employ one more convolutional layer with 1×1 kernel to generate the fine-scale density map. Each convolutional layer is followed by a PReLU activation function except for the last layer (with ReLU afterward).

Remark. In contrast to the CNN model in [127] supervised by two tasks that belong to different domains, i.e., classification and density map regression, our model is composed of two tasks in the same domain, i.e., regression of coarse and

fine density maps, which can facilitate feature extraction for better performance. Besides, with the multi-scale supervisions, our proposed architecture can avoid the bottleneck of information in this hourglass architecture (See Section 4.4.2.2).

4.3.2 Loss Function

The loss function of the MSD regressor is given by:

$$L = L_f + \lambda L_c \quad (4.8)$$

where L_c and L_f are the loss of the coarse- and fine-scale density map regressor, respectively, and λ is the weight term to control the balance between two regressors.

We use the smooth L1 loss [67] for each regressor since it is more robust to outliers than the mean squared error (MSE) loss. The loss functions of the two regressors are defined as:

$$L_c = \frac{1}{N} \sum_{i=1}^N \text{Smooth}_{L1} (F_c(X_i; \Theta_c) - D_i^c), \quad (4.9)$$

$$L_f = \frac{1}{N} \sum_{i=1}^N \text{Smooth}_{L1} (F_f(X_i; \Theta_f, \theta_c) - D_i^f), \quad (4.10)$$

where N is the number of training images; X_i is the i -th image; D^c and D^f are the ground truth coarse and fine density map, respectively; F_c and F_f are the output of Coarse- and Fine-Scale Network, respectively; Θ_c and Θ_f are the parameters of Coarse- and Fine-Scale Network, respectively; θ_c is the features from the second convolutional layer of Coarse-Scale Network. $F_f(X; \Theta_f, \theta_c)$ indicates that the fine density map estimation relies on the features of both Coarse- and Fine-Scale Network.

4.3.3 Ground Truth Generation

Object counting datasets provide dot annotations to represent the location of objects in an image. We firstly produce an annotation map by setting one at the object locations and zero at the other place. Then, the ground truth density map is generated by blurring the annotation map with a 2D Gaussian kernel G_σ with variance σ , where the kernel is normalized to one. The sum of each pixel value of a

density map is equal to the count of objects. The ground truth fine density map for the i -th image can be calculated by

$$D_i^f(x) = \sum_{j=1}^{C_i} \delta(x - x_j) * G_\sigma(x), \quad (4.11)$$

where C_i is the ground truth count corresponding to the i -th image; x_j is the j -th object location; the symbol '*' is a convolution operation. The computational process can be seen in Fig. 4.5(a).

FIGURE 4.5: Generation of fine and coarse density map. The objects are marked as red points in the annotation map.

To obtain the coarse density map, we need to resize the fine density map by the $1/4$ size. It will be generated automatically at the training stage. As shown in Fig. 4.5(b) we downsample the fine density map by a factor of 4 using one average pooling layers with a filter of size 4×4 and stride 4, which is followed by a multiplier with a factor of 16. Due to the average pooling, the energy of the coarse density map is $1/16$ compared with the fine one. Therefore, we compensate the loss of the energy by multiplying 16 to guarantee the equal energy (or equal crowd count).

4.4 Experiments

In the experiments, we separately evaluate Crowd-DCNet and MSD regressor compared with state-of-the-art methods.

4.4.1 Evaluation of Crowd-DCNet

We first describe the experimental settings, including datasets, implementation details, and evaluation metrics, followed by ablation studies for verifying the

effectiveness of each component of our approach. Finally, we compare Crowd-DCNet with state-of-the-art methods in both detection and counting tasks.

4.4.1.1 Experimental Settings

Datasets. We test our method on five representative datasets in crowd scenes, i.e., WiderFace [6] for dense face detection, ShanghaiTech [7] for crowd counting and localization, NWPU-Crowd [2] for crowd localization, and CARPK and PUCPR+ [8] for vehicle counting from the drone view. The descriptions detail in Section 2.2.2.

Implementation details. The backbone, ResNet-50, was initialized with the pre-trained ImageNet [34]. We trained our detector on 3 GPUs with the batch size of 12. We adopted Adam [128] optimizer with the learning rate of 7.5×10^{-6} for the WiderFace dataset and 7.5×10^{-5} for the remaining ones. The input images were randomly re-scaled, color distorted, flipped, and then cropped into 704×704 image patches. We used the same re-scale technique as CSP [87]. All training processes can only access to the point annotations. For the datasets with bounding box annotations, i.e., WiderFace, CARPK, and PUCPR+, we calculated their center points for training. We stopped training at 200k, 8k, 60k, 45k, and 4.5k iterations for WiderFace, ShanghaiTech, NWPU-Crowd, CARPK, and PUCPR+, respectively. Like [19, 87], we performed the multi-scale testing to generate bounding boxes. Then, non-maximum suppression (NMS) was used to filter the generated boxes. Following [21], we randomly took 10% training images of ShanghaiTech as the validation set. For ShanghaiTech and NWPU-Crowd, we chose the best model by performing a threshold search to minimize the counting error on the validation set. The other datasets adopted the confidence threshold of 0.4 to produce boxes for counting.

As mentioned in Section 4.2.2, we empirically chose multiple sets of parameters by which the precise pseudo sizes were generated for crowded objects on training sets. Table 4.2 shows the parameter settings used in Eqs. (4.3) and (4.7) for different datasets. We especially set the maximum value of α_j to 50 to avoid large gradients since dense crowds of ShanghaiTech produce large values of α_j . For all datasets, we set $\eta = 1$ in Eq. (4.7).

TABLE 4.2: Parameter settings of the LUDA-based pseudo size generation method for different datasets. “-” means the parameter is not required.

Dataset	Generation Parameters		
	K of Eq. (4.3)	β of Eq. (4.3)	Max α_j of Eq. (4.7)
WiderFace [6]	2	0.5	-
SHA [7]	2	1	50
SHA [7]	2	0.5	50
NWPU-Crowd [2]	2	0.8	200
CARPK [8]	1	1.3	-
PUCPR+ [8]	1	1.2	-

Evaluation metrics. For the detection task, we adopted the evaluation protocol in WiderFace [6] to calculate average precision (AP). The true positive is defined as the intersection of union (IoU) between ground truth boxes and detected boxes greater than a threshold of 0.5. For the counting task, we adopted the commonly-used mean absolute error (MAE) and root mean square error (RMSE) to evaluate the errors between the predicted counts and the ground-truth ones. The MAE indicates the accuracy of methods, while the RMSE reflects their robustness. They are defined as

$$\text{MAE} = \frac{1}{N_t} \sum_{N_t} |\hat{C}_i - C_i|, \text{RMSE} = \sqrt{\frac{1}{N_t} \sum_{N_t} (\hat{C}_i - C_i)^2}, \quad (4.12)$$

where N_t is the total number of testing images, and $\hat{C}_i$ and C_i are the estimated count and ground-truth count of the i -th image, respectively. When counting on WiderFace, we used a normalized MAE (NAE) [20], which normalizes the absolute error by the ground-truth face count.

For the center point localization task, we adopted two evaluation metrics, i.e., AP and mean localization error (MLE) respectively from [19] and [21]. AP defines true positive center point as those whose distance to its ground truth is smaller than a threshold of 20 pixels. MLE calculates the distances in pixels between the predicted center points and the ground truth, and then averages the distances over the testing set. One-to-one matching associates the predictions and the ground truth. The lower the value of MLE is, the better. The AP and MLE are suitable and reasonable for the datasets without bounding-box annotations, e.g., ShanghaiTech. For NWPU-Crowd, we used the evaluation protocol of [2], i.e., the Precision, Recall, and F1-measure. These metrics show the localization ability of detection-based algorithms in crowded environments.

TABLE 4.3: Comparisons of our LUDA-based pseudo size generation method with the GAK-based method on the training set of WiderFace [6]. Refinement of pseudo object sizes by our self-training approach is listed at the last entry. The number in the parentheses denotes the IoU threshold for AP calculation. The larger the value of AP is, the better.

Size Generation	AP (0.3)	AP (0.5)	AP (0.7)
GAK [7]	55.8	29.5	7.0
LUDA (ours)	60.3	31.2	7.5
Refined size (ours)	79.4	50.6	14.4

4.4.1.2 Ablation Study

We conducted ablation studies on the WiderFace [6] benchmark to evaluate and analyze the improvement of several important modules of our approach, including the locally-uniform distribution assumption (LUDA), the crowdedness-aware loss, and the confidence and order-aware refinement scheme. We evaluated our approach on both the detection and counting tasks. Table 4.3 shows the AP results of pseudo size generation methods on the training set of WiderFace, while Table 4.4 shows the AP, MAE, and NAE results on the validation set of WiderFace. The results of state-of-the-art point-supervised detection method (i.e., PSDDN [19]) and the crowd counting method (i.e., Shi *et al.* [20]) are respectively shown in the 1st and 2nd entries of Table 4.4 for comparison.

Effectiveness of the pseudo object size generation method. We compared the proposed LUDA-based pseudo size generation method with the geometry-adaptive kernel (GAK) based method in [7, 19]. The generated boxes were evaluated on the training set of WiderFace. We calculated the AP scores in three IoU thresholds of 0.3, 0.5, and 0.7, so the true positives become gradually harder to reach. The results are shown in Table 4.3, where it can be observed that the proposed LUDA-based method obtains higher AP scores under all thresholds, compared with the GAK-based method, validating the effectiveness of our LUDA. Such an advantage also means that we can generate more accurate pseudo bounding boxes at the beginning of training.

Fig. 4.6 shows a visual comparison of the pseudo object sizes generated by GAK and LUDA in a crowded scene. We can observe that: 1) both methods generate more accurate sizes for crowded objects than sparse objects; and 2) LUDA makes

the sparse objects' sizes be accurate when they are surrounded by crowded objects, as illustrated in the green and yellow boxes of Fig. 4.6.

FIGURE 4.6: Visual comparison of the pseudo object sizes generated by GAK and LUDA in a crowded scene. For LUDA, the number on top of each bounding box refers to the object crowdedness. The high crowdedness value means the pseudo size is accurate.

We also evaluated the GAK-based and LUDA-based methods by yielding bounding boxes on the validation set of WiderFace. The AP scores are shown in the 3rd and 4th entries of Table 4.4, where it can be seen that the accuracy of the generated bounding boxes is extremely low, and especially the AP score is only 7.2% on the easy subset. This observation shows that only applying pseudo size generation methods cannot obtain accurate object sizes even with the ground-truth center points.

Effectiveness of the crowdedness-aware loss and the confidence and order-aware refinement scheme. The crowdedness-aware loss and the confidence and order-aware refinement scheme are the critical components to improve the AP for point-supervised detection. Based on the pseudo bounding boxes, we trained the detector on the training set of WiderFace with four sittings 1) using only the crowdedness-aware loss, 2) using only the confidence and order-aware refinement scheme, 3) using neither of the two modules, and 4) using both of the two modules. The AP, MAE, and NAE results are listed in the 5th to 8th entries of Table 4.4.

TABLE 4.4: Ablative studies towards the crowdedness-aware loss and the confidence and order-aware refinement scheme, as well as comparisons with state-of-the-art point-supervised detection method [19] and counting method [20] on the validation set of WiderFace [6]. “C-by-D” means counting by detection, and “C-by-R” means counting by regression. “Size Init.” stands for directly applying the pseudo object size initialization method on the validation set to obtain the detection result. “-” is not applicable. The larger the value of AP is, the better. The lower the values of MAE and NAE are, the better.

Method	Category	Crowdedness-aware loss	Confidence and order-aware refinement scheme	Detection AP			Counting	
				easy	medium	hard	MAE	NAE
PSDDN [19]	C-by-D	-	-	60.5	60.5	39.6	-	-
Shi <i>et al.</i> [20]	C-by-R	-	-	-	-	-	3.2	0.40
GAK [7]	Size Init.	-	-	7.1	12.5	27.3	-	-
LUDA (ours)	Size Init.	-	-	7.2	12.8	29.5	-	-
Crowd-DCNet (ours)	C-by-D	$\times$	$\times$	18.6	23.4	30.8	3.4	0.81
		$\checkmark$	$\times$	21.3	26.8	35.2	3.6	0.97
		$\times$	$\checkmark$	55.1	55.1	52.5	2.3	0.27
		$\checkmark$	$\checkmark$	75.8	71.0	64.4	2.2	0.29

(1) Pseudo sizes only. The detector was trained with the pseudo object sizes only, as listed in the 5th entry of Table 4.4. The AP scores are improved by about 11% on the easy and medium subsets in comparison with the LUDA-based size generation method. However, the values are still low, e.g., 18.6%, 23.4%, and 30.8% AP on the easy, medium, and hard subsets, respectively. This observation demonstrates that it is hard to attain an acceptable detector when trained only with pseudo bounding boxes.

(2) Either crowdedness-aware loss or confidence and order-aware refinement scheme. As shown in the 6th and 7th entries of Table 4.4, with the use of the crowdedness-aware loss (resp. the confidence and order-aware refinement scheme), the AP scores increase to 21.3% (easy), 26.8% (medium), and 30.8% (hard) (resp. 55.1% (easy), 55.1% (medium), and 52.5% (hard)), which validate the effectiveness of these two modules. Moreover, it can be known that the refinement module is more effective than the loss. Note that only with the refinement scheme, our approach outperforms PSDDN [19] on the hard subset in the detection task (52.5% AP vs. 39.4% AP) and Shi *et al.* [20] in the counting task (2.3 MAE vs. 3.2 MAE).

(3) Both the crowdedness-aware loss and confidence and order-aware refinement scheme. When both modules are activated, our approach achieves the best results in both detection and counting tasks. There is a big jump of AP from using a single module to both modules. The reason may be that the crowdedness-aware loss emphasizes the accurate pseudo sizes, but neglects to update the noisy sizes during training, while the confidence and order-aware refinement scheme updates

the pseudo sizes but it overlooks the importance of the accurate pseudo sizes in the network’s weights updating. The combination of such two modules could well compensate to each other. Numerically, the AP scores of our approach with both two modules increase to 75.8% (easy), 71.0% (medium), and 64.4% (hard). Compared with PSDDN [19], our method improves the AP by more than 10%. The MAE and NAE of our approach respectively decrease approximately 31.2% and 27.5% against [20]. The results demonstrate the significant superiority of our method in both detection and counting tasks.

Besides, the last row of Table 4.3 shows that the refined pseudo sizes improve the AP (0.3) and AP (0.5) by nearly 20%. The gradually enhanced quality of training examples helps the detector become stronger. To demonstrate this improvement, we plot some training examples before and after training on the WiderFace dataset in Fig. 4.2. Especially for large and sparse objects, the bounding boxes are refined to encompass the face regions.

Impact of the confidence and order-aware refinement scheme. In our confidence and order-aware refinement scheme, the pseudo object sizes are updated in a robust and orderly manner, i.e., the easy examples (sparse and large objects) with inaccurate sizes are first refined, followed by the hard examples (crowded and small objects). The object sizes are refined when their posterior probabilities are larger than the threshold of 0.6. To illustrate the impact of such a refinement scheme, we counted the percentage of examples whose probabilities exceed 0.6 during training. Specifically, we split examples into four categories corresponding to the crowdedness intervals $[0, 1]$, $(1, 4]$, $(4, 14]$, and $[14, +\infty)$ with 64k, 30k, 31k, and 30k training examples, respectively. Fig. 4.7 shows the percentage of examples with their posterior probabilities larger than 0.6 under each category. We can observe that the percentage of the revised low-crowdedness examples increases more rapidly than that of the high-crowdedness examples in the first 20 epochs. Throughout the entire training epochs, the percentage corresponding to lower-crowdedness reaches the highest, demonstrating the effectiveness of our refinement scheme.

4.4.1.3 Comparison with State-of-the-art Methods

Point-supervised face detection. We compared our approach with state-of-the-art point-supervised detection methods, including PSDDN [19] and LSC-CNN [21],

FIGURE 4.7: Percentage of refined training examples with respect to the training epoch.

and the counting-by-regression method, i.e., CSR-A-thr [21]. The CSR-A-thr is the detection version of CSRNet [18]. Table 4.5 shows the AP scores of different methods on the validation set of WiderFace [6]. From Table 4.5, we can see that our Crowd-DCNet outperforms the other point-supervised methods to a significant margin (i.e., more than 10% AP improvement). Especially on the hard subset, our method improves the AP score of the second best LSC-CNN method by 18.2%.

TABLE 4.5: Comparisons with state-of-the-art point-supervised detection methods on the validation set of WiderFace [6]. “*”: the AP results are provided by [19]. The larger the value of AP is, the better. The initialized pseudo boxes were used when training CSP with point supervision.

Method	Supervision	easy	medium	hard
LSC-CNN [21]	Box	57.3	70.1	68.9
Faceness-WIDER [129]	Box	71.3	63.4	34.5
Faster R-CNN* [83]	Box	84.0	72.4	34.7
CSP (anchor-free) [87]	Box	90.7	95.2	96.1
HR-TinyFace [130]	Box	92.5	91.0	80.6
CSP (anchor-free) [87]	Point	18.6	23.4	30.8
CSR-A-thr [21]	Point	30.2	41.9	33.5
PSDNN [19]	Point	60.5	60.5	39.6
LSC-CNN [21]	Point	40.5	62.1	46.2
Crowd-DCNet (ours)	Point	75.8	71.0	64.4

Besides, Table 4.5 provides the results of several box-supervised methods for comparison, where it can be seen that our method even outperforms box-supervised LSC-CNN and Faceness-WIDER on the easy subset and Faceness-WIDER and

Faster-RCNN on the hard subset. Although the AP score of the box-supervised CSP achieves above 90%, it significantly drops to 30.8% when trained with point-initialized pseudo boxes, which demonstrates the effectiveness of our point-supervised training scheme. Fig. 4.8 shows some visual results of bounding box predicted by our method.

FIGURE 4.8: Qualitative results on (a) WiderFace [6], (b) SHA [7], (c) SHB [7], (d) PUCPR+ [8], and (e) CARPK [8]. The top row shows the ground-truth boxes or points, and counts. The bottom row shows the bounding boxes and counts predicted by our Crowd-DCNet. *Zoom in on the figure for better viewing.*

Center point localization of crowds. (1) SHA and SHB datasets. To evaluate the localization ability for the datasets with only point-level annotations, we compared our method with PSDNN [19], CSR-A-thr [21], and LSC-CNN [21] on the SHA and SHB [7] datasets. Table 4.6 presents the results of the AP and MLE metrics. Our approach obtains the best results, showing 85.3% AP and 8.0 MLE for SHA, and 91.6% AP and 6.0 MLE for SHB.

TABLE 4.6: Comparisons with crowd localization methods on SHA and SHB [7] datasets. The larger the value of AP is and the lower the value of MLE is, the better.

Method	SHA		SHB	
	AP (%)	MLE	AP (%)	MLE
PSDDN [19]	73.7	-	75.9	-
CSR-A-thr [21]	-	16.8	-	12.3
LSC-CNN [21]	67.6	9.6	76.0	9.0
Crowd-DCNet (ours)	85.3	8.0	91.6	6.0

(2) NWPU-Crowd dataset. We also compared our approach with state-of-the-art crowd localization methods on the large-scale NWPU-Crowd [2] dataset. The results are shown in Table 4.7, where it can be observed that our method obtains the best F1-measure (i.e., 63.8%) and Recall (i.e., 62.4%). Fast R-CNN achieves 95.8% Precision but sacrifices the Recall to 3.5%, indicating that Fast R-CNN fails to

detect crowded people. Note that the proposed approach is the only one that can output bounding boxes when trained with point supervision.

TABLE 4.7: Comparison with state-of-the-art crowd localization methods on the NWPU-Crowd test set. The values of F1-measure, Precision, and Recall were calculated under the threshold σ_l [2].

Method	Supervision	Output	F1 (%)	Precision (%)	Recall (%)
Faster R-CNN [83]	Box	Box	6.7	95.8	3.5
TinyFaces [130]	Box	Box	56.7	52.9	61.1
VGG+GPR [131]	Point	Point	52.5	55.8	49.6
RAZ_Loc [132]	Point	Point	59.8	66.6	54.3
Crowd-DCNet (ours)	Point	Box	63.8	65.1	62.4

Crowd counting. In addition to detection, we evaluated our Crowd-DCNet in the crowd counting task. Table 4.8 shows the results on SHA and SHB [7], and WiderFace [6] datasets. For a fair comparison, we split the counting methods into two categories: counting-by-regression methods and counting-by-detection methods. Our method achieves the best performance when compared with state-of-the-art counting-by-detection methods. For SHA and SHB, the proposed method is comparable to state-of-the-art counting-by-regression methods, such as Shi *et al.* [20] and HA-CCN [81]. In addition, we show the results of our MSD regressor, which comparable to state-of-the-art methods. Note that the MSD regressor is designed for low-density surveillance datasets, and we will demonstrate its effectiveness in Section 4.4.2.

Our Crowd-DCNet not only provides the count but also estimates the bounding boxes for object instances. We qualitatively evaluated the proposed method by visualizing the predicted bounding boxes on the SHA, and SHB datasets in Fig. 4.8. Besides, Fig. 4.9 shows the predicted boxes by our method and the estimated density maps by CSRNet [18] on the SHA and SHB datasets. We argue that the box outputs of our method are more informative than the density maps of counting-by-regression methods because the boxes provide high-level understanding of crowds.

Vehicle counting from drone view. To evaluate the generalization ability in other domains, we trained our model on the CARPK and PUCPR+ datasets [8]. The datasets provide densely-packed car counting images from drone view. Table 4.9 reports the MAE and RMSE values of our method and state-of-the-art vehicle counting methods [8, 85, 88, 133, 134]. It can be seen that our method consistently

TABLE 4.8: Comparisons with state-of-the-art counting-by-regression methods (in the top part) and counting-by-detection methods (in the bottom part) on SHA and SHB [7], and WiderFace [6] (WF) datasets. “-”: the author does not provide the result. “*”: the results are provided by [21]. The lower the values of MAE and RMSE are, the better.

Method	SHA		SHB		WF
	MAE	RMSE	MAE	RMSE	MAE
Zhang <i>et al.</i> [7]	110.2	173.2	26.4	41.3	7.1
CSRNet [18]	68.2	115.0	10.6	16.0	4.3
Cao <i>et al.</i> [68]	67.0	104.5	8.4	13.6	8.5
PSDNN+ [19]	65.9	112.3	9.1	14.2	-
Shi <i>et al.</i> [20]	65.2	109.4	7.2	12.2	3.2
PSD+DCL [80]	65.0	108.0	8.1	13.3	-
HA-CCN [81]	62.9	94.9	8.1	13.4	-
MSD regressor (ours)	66.5	101.8	9.9	14.9	-
<hr/>					
TinyFace* [130]	237.8	422.8	-	-	-
LC-FCN8 [89]	-	-	13.1	-	-
PSDNN [19]	85.4	159.2	16.1	27.9	-
LSC-CNN [21]	66.4	117.0	8.1	15.7	-
Crowd-DCNet (ours)	65.1	104.4	7.8	12.6	2.2

FIGURE 4.9: Comparisons between the bounding boxes (at top row) produced by our Crowd-DCNet and the density maps (at bottom row) produced by CSRNet [18] on the SHA and SHB [7] datasets. The predicted count is shown at the bottom-left corner of an image. The ground-truth counts from left to right images are 1068, 584, 139, and 274. *Zoom in on the figure for better viewing.*

performs better than the other methods, demonstrating that our model is flexible for various detection and counting tasks. Some visual results on CARPK and PUCPR+ are shown in Fig. 4.8.

4.4.2 Evaluation of MSD Regressor

We evaluate the MSD regressor on surveillance datasets in the counting task.

TABLE 4.9: Comparisons with state-of-the-art vehicle counting methods on the CARPK and PUCPR+ benchmark [8]. The lower the values of MAE and RMSE are, the better.

Method	CARPK		PUCPR+	
	MAE	RMSE	MAE	RMSE
LPN Counting [8]	23.80	36.79	22.76	34.46
RetinaNet [85]	16.62	22.30	24.58	33.12
IEP Counting [133]	51.83	-	15.17	-
Goldman <i>et al.</i> [88]	6.77	8.52	7.16	12.00
Li <i>et al.</i> [134]	5.24	7.38	3.92	5.06
Crowd-DCNet (ours)	4.95	7.09	3.20	4.83

4.4.2.1 Comparison with State-of-the-art Methods

UCSD dataset. The UCSD dataset [119] was captured from a surveillance camera. It contains 2000 frames, sampled by video clips with the size of 158×238 and the framerate of 10. The region of interest (ROI) of images is provided. For a fair comparison, we employ the same configuration with [119]. We train our model by 601st to 1400th frames and test it by the remaining 1200 frames. According to [18, 124], we use ROI to mask the images and density maps.

TABLE 4.10: Comparison on UCSD dataset.

Method	MAE	RMSE
Traditional approaches		
Gaussian process regression [119]	2.24	7.97
Ridge regression [135]	2.25	7.82
Cumulative attribute regression [136]	2.07	7.90
Lempitskt's [79]	1.70	-
CNN-based approaches		
Zhang <i>et al.</i> [137]	1.60	3.31
CCNN [124]	1.51	-
MCNN [7]	1.07	1.35
FCN-MT [138]	1.67	3.41
CSRNet [18]	1.16	1.47
MSD regressor (ours)	1.04	1.33

The MAE and RMSE of the proposed method and compared methods are shown in Table 4.10. The traditional global count regression methods [79, 119, 135, 136] are inferior to CNN-based density map estimation methods [7, 18, 124, 137, 138]. Our MSD regressor obtains the best results compared with state-of-the-art approaches.

TRANCOS dataset. The TRANCOS dataset [124] is captured from traffic surveillance cameras in different road sections. It comprises 1244 images, in which 823 images are used for training and 421 images are used for testing. Each vehicle

is manually labeled with a dot. The ROI is also provided for evaluation. The Grid Average Mean Absolute Error (GAME) [124] is used as the evaluation metric for this dataset. The GAME (L) divides an image into 4^L non-overlapping regions and calculates the MAE of each region, and the overall error is the sum of all MAEs. Note that GAME (0) is equivalent to MAE.

TABLE 4.11: Comparison on TRANCOS dataset. ‘-’ indicates the method does not provide the results.

Method	MAE	GAME (1)	GAME (2)	GAME (3)
CCNN [124]	10.99	13.75	16.69	19.32
FCN-MT [138]	5.31	-	-	-
CSRNet [18]	3.56	5.49	8.57	15.04
MSD regressor (ours)	2.97	4.39	6.43	9.92

Results are shown in Table 4.11. Our method outperforms state-of-the-art methods for all GAMES. Compared with [18], the proposed MSD regressor decreases the MAE, GAME (1), GAME (2), and GAME (3) by 17%, 20%, 25%, and 34%, respectively. The low GAMES means the proposed method predicts the count precisely not only for the whole image but also for the sub-regions of the image. Some visual results predicted by the proposed method are shown in Fig. 4.10.

FIGURE 4.10: Qualitative results from the UCSD (first two columns) and TRANCOS (last two columns) dataset. The ROI is inside the green lines. The first row shows the input images. The second row shows the corresponding ground truth density map and count. The third row shows the estimated density map and count of our MSD regressor.

Remark. Our MSD regressor has a compact architecture with 0.372 million parameters compared with the CSRNet with 16.26 million parameters. Our light

model can perform well on video surveillance datasets, which is crucial for the embedded application.

4.4.2.2 Ablation Study

We investigate the effectiveness of the coarse density map regressor. It is composed of two settings: the MSD regressor with or without the coarse density map (CDM). The ablation study is conducted on UCSD and TRANCOS dataset. Table 4.12 shows the results of two settings. These results indicate that the coarse density map regressor can provide extra supervision to improve the performance and even avoid the divergence of the hourglass structure.

TABLE 4.12: Ablation study. ‘-’ indicates the model diverges.

Dataset	UCSD		TRANCOS	
CDM	×	✓	×	✓
MAE	1.17	1.04	-	2.97

4.5 Chapter Summary

In this chapter, we have presented a self-training approach to train a typical detector with only point-level annotations such that the detector can accurately detect and count objects in crowd scenes simultaneously. This is achieved by the locally-uniform distribution assumption, the crowdedness-aware loss, the confidence and order-aware refinement scheme, and the decoding method. Our approach promotes the detector to generate accurate bounding boxes in a coarse-to-fine and end-to-end manner. Moreover, bypassing object detection, we proposed a multi-scale density regression method for counting the number of crowds in video surveillance. Extensive experimental results over multiple benchmark datasets demonstrated that the proposed methods achieve outstanding performance in the crowd detection and counting tasks.

Chapter 5

License Plate Detection and Recognition

Chapter 3 introduced image restoration methods, while Chapter 4 introduced crowd detection and counting methods. Dense prediction is adopted in the computer vision task at different levels, i.e., low-level transmission estimation, medium-level ambient light region segmentation, and high-level crowd detection and counting. In this chapter, we focus on a high-level vision task, i.e., recognition. Based on small object detection, we present an automatic license plate recognition approach, in which the characters on a license plate can be densely predicted. This dense prediction method is demonstrated in Figure 5.1. Unlike the previous chapters, the task of this chapter is to predict a sequence of characters, which is represented by a 1D dense vector.

FIGURE 5.1: Automatic license plate recognition method.

5.1 Introduction

Automatic license plate recognition (ALPR) aims at exploiting image processing and pattern recognition techniques to extract and recognize the characters on license plates (LPs) from vehicle images or videos. ALPR approaches are crucial for intelligent transport systems (ITS) in a wide variety of applications, such as traffic surveillance and control, parking management, and toll station management [139]. However, ALPR methods often suffer from images captured in unconstrained scenarios, such as oblique views, uneven illumination, and different weather conditions, as shown in Figure 5.2.

FIGURE 5.2: Images captured in unconstrained scenarios, e.g., (a) oblique view, (b) uneven illumination, and (c) snow in the nighttime, labeled by ground-truth (GT) bounding boxes, four vertices, and characters. Our AOLP approach can handle these cases and generate correct predictions (Pred).

Benefiting from deep neural networks (DNNs), modern ALPR methods achieve a dramatic improvement. They either adopt a unified framework [91] where two typical components (license plate detection and character recognition) are performed simultaneously in an end-to-end model or design a cascaded framework [98] where each component is optimized for specific tasks and connected in a cascaded manner. Regardless of which framework is chosen, the speed/accuracy trade-off always accompanies ALPR’s design process. How to design an effective and efficient ALPR system is still an open question.

In unified frameworks, the region proposal network (RPN) [83] was commonly adopted. Li *et al.* [91] proposed a unified method, named TE2E, which uses an RPN for LP detection and recurrent neural networks (RNNs) for character recognition. This method is not efficient due to RNNs. Xu *et al.* [3] proposed a

CNN-based Roadside Parking net (RPnet) to improve efficiency. However, small-size images adversely affect recognition performance since the details of the characters are lost. Enlarging input size implies an increase of the inference time. Another issue is that task-specific techniques (e.g., LP rectification for input images) are unavailable in such unified networks. These drawbacks harm both detection and recognition performance.

On the other hand, a typical DNN-based cascaded framework employs an LP detection network followed by the character recognition network. Some methods [66, 98] utilize the YOLO [93] network to perform LP detection and use CR-NET [94] for character segmentation and recognition. A semantic segmentation network was proposed to segment characters on LPs by Zhuang *et al.* [103]. However, character segmentation needs laborious annotations of each character. The other methods [24, 100] adopt RNN-based structures for character recognition. However, RNN hinders a compact and efficient design. Additional character segmentation or RNNs decreases the inference speed compared with CNNs.

In view of these issues, we propose a high-performing ALPR approach, termed VSNet, with five contributions.

(1) We propose a resampling-based cascaded manner to combine the detection and recognition components, i.e., VertexNet for LP detection and Squeeze Character Recognition Network (SCR-Net) for LP recognition, as shown in Fig. 5.3. VertexNet is designed with small-resolution input (even losing character details), achieving high inference speed and decreasing memory usage. This would not affect recognition performance in SCR-Net since the LP is resampled from the finest input image and rectified to high resolution according to the predicted vertices by VertexNet.

FIGURE 5.3: Framework of the proposed VSNet. An input image is resized to a small resolution, i.e., 256×256 , for fast inference in VertexNet. The LP patch is then resampled from the finest input image and rectified to high resolution according to the predicted vertices by VertexNet. Finally, SCR-Net recognizes all characters in the LP.

- (2) Our network architecture adopts well-designed CNNs without additional character segmentation or RNNs. One forward pass of the networks can obtain detection and recognition results, which is more suitable for the real-time system with limited resources. The first two insights ensure fast inference speed.
- (3) We separately implement VertexNet and SCR-Net with novel vertex-based architecture designs. In VertexNet, we propose an integration block (IB) to effectively extract the spatial features of LPs. With vertex supervisory information, we propose a vertex-estimation branch for predicting the geometric shapes of LPs, which are further utilized for LP rectification in SCR-Net. Moreover, the vertices can be used to augment training images. In SCR-Net, we propose a horizontal encoding technique for left-to-right feature extraction and a weight-sharing classifier for character recognition.
- (4) We propose a weight-sharing classifier to balance the training samples and address the character recognition failure of CNNs on small-scale datasets. The parameters of our classifier account for only 4% of the fully-connected classifier. The third and fourth insights guarantee outstanding performance.
- (5) In addition, we propose a dynamic regularization method for SCR-Net, in which the regularization strength is adaptable to the change of the training loss. During training, the regularization strength is gradually increased with respect to the training status. Analogous to education, the regularizer is regarded as an instructor who gradually increases the difficulty of training examples in a feature perturbation manner corresponding to the student’s (SCR-Net) learning progress.

Table 5.1 illustrates the comparisons of the characteristics between our VSNet and state-of-the-art methods. VSNet employs the cascaded framework with the CNN structure only and adopts the vertex estimation and LP rectification, which outperforms state-of-the-art ALPR methods in both speed and accuracy. Experiments show that VertexNet improves the detection precision to 99.1% on the CCPD [3] dataset, surpassing the state-of-the-art method [100] by 60% relative improvement on error rate. SCR-Net increases the recognition accuracy to 99.5% and 99.7% on the CCPD and AOLP [101] datasets, respectively, achieving more than 50% relative improvement on error rate compared with the recent work [24]. Our method has the fastest inference speed among those ALPR methods, i.e., the

fast version of our approach can run at 149 FPS on a single NVIDIA GTX1080Ti GPU.

TABLE 5.1: Comparisons of the characteristics between state-of-the-art methods and VSNet. Performance is evaluated on CCPD [3].

Characteristic	[91]	[3]	[100]	[24]	VSNet
Unified framework	✓	✓			
Cascaded framework			✓	✓	✓
RNN structure	✓		✓	✓	
CNN structure only		✓			✓
Vertex estimation			✓		✓
LP rectification			✓		✓
Inference speed rank	5	2	3	4	1
Performance rank	5	4	3	2	1

The remainder of this chapter is organized as follows. The proposed VSNet is presented in detail in Section 5.2. Experimental results and ablation studies are discussed in Section 5.3. Finally, we conclude the chapter in Section 5.4.

5.2 Proposed VSNet for ALPR

The proposed VSNet is composed of two cascaded CNNs, i.e., VertexNet for LP detection and SCR-Net for LP recognition. Fig. 5.4 shows the architecture of VSNet. We decouple the design of VSNet into detection and recognition, improving overall performance. VertexNet is a compact yet high-performing one-stage detector implemented with small input size, narrow channel of high-level layers, and vertex estimation. SCR-Net is a robust and efficient classifier. We propose LP rectification to resample and normalize various LP detections. SCR-Net employs a horizontal encoding technique for left-to-right feature extraction and a weight-sharing classifier for character recognition. Besides, A dynamic regularization is proposed to avoid overfitting in SCR-Net. This section will describe VertexNet, SCR-Net, and dynamic regularization in detail.

5.2.1 VertexNet for LP Detection

The inference speed of a detector is considered as a bottleneck of VSNet. The critical insight is to design the detector with a small input size and compact architecture.

FIGURE 5.4: Architecture of VSNet. (1)-(4) are executed sequentially in inference. H and W are set to 256 for fast inference. VertexNet consists of the backbone, fusion, and head networks, which predicts the bounding boxes and vertices of LPs. In the fusion structure, “ $2 \times U$ ” is the bilinear upsampling with a factor of 2. “BR” is the batch normalization followed by the ReLU. “CBR” is the 1×1 -s1 convolution followed by the “BR”. $[\cdot]$ is the concatenation operation. SCR-Net resamples and rectifies LP images from the finest input images based on the predicted vertices (red points in the LP) and recognizes the characters on the LP.

However, the small input size leads to extremely small LPs, which poses a threat to state-of-the-art detectors. Some existing methods [66] applied two cascaded YOLO networks with the input size of 416×416 to detect vehicles and LPs sequentially, but two-step detection increases the inference time and may accumulate errors. In VertexNet, we instead introduce a single detector with the small size (256×256). To deal with small LPs, we propose a novel architecture as follows.

5.2.1.1 Architecture of VertexNet

The proposed VertexNet contains three modules, i.e., backbone, fusion, and head networks, as shown in Fig. 5.4. Specifically, instead of using off-the-shelf architectures (YOLO) as a backbone, we propose an integration block (IB) by leveraging the residual block [22] and enhancing the Squeeze-and-Excitation (SE) attention module [140]. In the head network, in addition to the box offsets estimation, we propose the vertex-estimation branch of LPs, which improves the detection performance and benefits to character recognition in SCR-Net.

Backbone network. The backbone is split into six stages (Stage-1 to 6), and the output size is half of the input size in each stage. We define the output feature maps

as $C_i, i \in \{1, 2, \dots, 6\}$. The input images are first fed to Stage-1, which employs a 7×7 -s2 Conv-BN-ReLU layer (a convolutional layer with the kernel of 7×7 and the stride of 2 followed by the batch normalization and the rectified linear unit). Then, the output features are fed to the last five stages successively, each of which is composed of two IBs. For Stage-2, the downsample of feature maps is achieved by a 2×2 -s2 max-pooling layer inserted before the first IB, while for the remaining stages, the downsample of feature maps is achieved by a 3×3 -s2 convolutional layer in the first IB.

The IB module is an effective building block in VertexNet, as shown in Fig. 5.5. Since LPs present a small scale, the multiple channels in top layers of ResNet [22] that handles large objects are unnecessary for small LP detection. We design the IB module with fixed-channel residual block [22]. The number of channels in each IB is set to 128. This naive version of IB contains fewer parameters, and its accuracy is comparable to ResNet-50 (see Section 5.3.3.2). Moreover, we enhance the SE attention module [140] by introducing the p -norm in the *squeeze* step, named p -norm SE. The *squeeze* step aggregates the feature maps across spatial dimensions to generate channel descriptor. In the original SE module, only the global average pooling is adopted. In the p -norm SE module, however, we calculate two channel-wise p -norms, i.e., 1-norm and ∞ -norm, and then fuse those features. The 1-norm feature encodes the global average information, while the ∞ -norm feature extracts the significant information. The p -norm SE enhances the channel-wise statistics compared with the original SE. With the p -norm SE, the IB module is superior to ResNet-50 in terms of accuracy and model size (see Section 5.3.3.2).

FIGURE 5.5: Implementation of the integration block. F_r , F_{sq} , and F_{ex} are the residual branch, squeeze step, and excitation step, respectively. $[\cdot]$, $\odot$, and $\oplus$ denote the concatenation operation, element-wise product, and element-wise addition, respectively. $\{\cdot\} \times x$ denotes performing the $\{\cdot\}$ structure by x times. The convolutional layers have the kernel size of 3×3 in F_r and 1×1 in F_{ex} .

Formally, let the input and output features of the residual branch (F_r) be $\mathbf{x}$ and $\mathbf{y}$, respectively, as shown in Fig. 5.5. We have $\mathbf{y} \in \mathbb{R}^{C \times H \times W}$ and $\mathbf{y} =$

$[y_1, y_2, \dots, y_C], y_c \in \mathbb{R}^{H \times W}$ where y_c is spatial features of the c -th channel ($C=128$ for all experiments). We first feed the features to the *squeeze* step (F_{sq}) of the p -norm SE and obtained two channel descriptors $\mathbf{z}^k \in \mathbb{R}^{C \times 1 \times 1} = [z_1^k, z_2^k, \dots, z_C^k], k = \{1, \infty\}$, which are calculated by Eq. (5.1) and Eq. (5.2), respectively:

$$z_c^1 = F_{1\text{-norm}}(y_c) = \sum_{i=1}^H \sum_{j=1}^W y_c(i, j), \quad (5.1)$$

and

$$z_c^\infty = F_{\infty\text{-norm}}(y_c) = \max_{i,j} y_c(i, j). \quad (5.2)$$

In practice, we adopt the global average-pooling layer and global max-pooling layer to implement the 1-norm and ∞ -norm squeeze operation, respectively. Then, the channel descriptor $\mathbf{z} \in \mathbb{R}^{2C \times 1 \times 1}$ is obtained by concatenating 1-norm and ∞ -norm features, i.e., $\mathbf{z} = [\mathbf{z}^1, \mathbf{z}^\infty]$. Next, we perform the *excitation* step (F_{ex}), like the one in [140], to capture channel-wise dependencies. Specifically, a 1×1 Conv-ReLU followed by a 1×1 Conv-Sigmoid is applied to process the channel descriptor:

$$\mathbf{s} = F_{ex}(\mathbf{z}, \mathbf{W}) = \sigma(\mathbf{W}_2 \times \delta(\mathbf{W}_1 \times \mathbf{z})), \quad (5.3)$$

where $\mathbf{W}_1 \in \mathbb{R}^{\frac{C}{r} \times 2C}$ and $\mathbf{W}_2 \in \mathbb{R}^{C \times \frac{C}{r}}$ are the weights of the first and second convolutional layers, δ and σ denote ReLU and sigmoid activation, respectively, and $\mathbf{s} \in \mathbb{R}^{C \times 1 \times 1}$ is the scalar to control the channel attention of the features $\mathbf{y}$. r is the reduction ratio and set to 4 for all experiments. The output of the p -norm SE is calculated by scaling $\mathbf{y}$ with $\mathbf{s}$:

$$\tilde{\mathbf{y}} = \mathbf{s}' \odot \mathbf{y}, \quad (5.4)$$

where $\tilde{\mathbf{y}} \in \mathbb{R}^{C \times H \times W}$ is the scaled features, $\odot$ denotes the element-wise product, and $\mathbf{s}'$ is the repeated $\mathbf{s}$ in H and W dimensions, i.e., $\mathbb{R}^{C \times 1 \times 1} \rightarrow \mathbb{R}^{C \times H \times W}$. Finally, the output features of IB is defined by:

$$\tilde{\mathbf{x}} = \tilde{\mathbf{y}} \oplus \mathbf{x}, \quad (5.5)$$

where $\oplus$ is the element-wise addition between two tensors.

Fusion network. The feature pyramid network (FPN) [141] is modified to fuse the features of the backbone network. We employ a top-down route to fuse multi-scale

features. As shown in the *Fusion* part of Fig 5.4, the basic structure fuses the output features from two consecutive stages. To match the feature sizes, the bilinear upsampling is first used for upsizing the lower-resolution features by a factor of 2. The higher-resolution features are passed through BN-ReLU. Then, the two feature maps are concatenated and produced by a 1×1 -s1 Conv-BN-ReLU. With the concatenation and Conv-BN-ReLU, the structure becomes a weighted fusion manner, and the weights are learnable. We fuse the features from Stage-6 to Stage-3, producing the fused feature maps $P_i, i \in \{5, 4, 3\}$. The fused multi-scale features have a good representation ability compared with the original ones.

Head network. In the previous works [3, 91], the head network performs LP classification and box offset regression. In addition, we add a vertex-estimation branch to predict four 2D vertices of LPs, as shown in the *Head* part of Fig 5.4. Specifically, the head network takes three fused feature maps ($P_i, i \in \{3, 4, 5\}$) and one top-level feature maps (C_6) as the inputs and produces 4-scale predictions using two convolutional layers. Unlike SSD [84], the layers in our head network share the parameters across different-scale features, i.e., two convolutional layers process four feature maps. Large-scale feature maps P_3 and P_4 are processed by a 3×3 -s1 convolutional filter (Shared-head A), while small-scale feature maps P_5 and C_6 are processed by another 3×3 -s1 convolutional filter (Shared-head B). Sharing parameters not only allow the filter to fit different-scale objects but also reduces the number of network parameters.

The channel number of output predictions are based on the number of the default boxes (or anchor boxes). Following [141], we use 9 anchor boxes for each location in the output feature maps. Since we predict 2 class scores (LP or background), 4 box offsets, 8 vertex values (4×2 D vertex), the channel number is equal to $(2 + 4 + 8) \times 9 = 126$.

VertexNet-Fast. To achieve faster inference, we further simplify VertexNet with shallow architecture. In particular, VertexNet-Fast adopts the naive IB in the backbone network, i.e., six naive IB-based stages without the p -norm SE module. Although the SE module brings a few parameter increases, it doubles the network depth and increases the inference time.

5.2.1.2 Training of VertexNet

We define the anchor boxes whose Jaccard index (or intersection over union, IoU) with the ground-truth bounding boxes larger than 0.25 as positive anchors. Other anchor boxes are defined as negative anchors. We choose such a small IoU threshold because it guarantees that most LPs can match the anchor boxes. Following SSD, we use hard negative mining to significantly reduce the number of negative anchors. We choose negative anchors by the highest losses, and their number is 3 times larger than the positive ones. The overall loss consists of three aspects, i.e., classification loss for LP classification (L_{cls}), localization loss for box offset regression (L_{box}), and vertex estimation loss for vertex regression (L_{vert}). Formally, we minimise the overall loss for any anchor box j :

$$L = \lambda_1 L_{cls}(p_j, \hat{p}_j) + \lambda_2 p_j L_{box}(t_j, \hat{t}_j) + \lambda_3 p_j L_{vert}(l_j, \hat{l}_j), \quad (5.6)$$

where L_{cls} is implemented by Softmax layer followed by the cross-entropy loss. p_j is the ground-truth label of anchor j with 1 for LP and 0 for background, and $\hat{p}_j$ is the corresponding predicted probability. L_{box} and L_{vert} are achieved by Smooth-L1 loss [67, 84]. $t_j = \{t_x, t_y, t_w, t_h\}$ and $l_j = \{l_{x_1}, l_{y_1}, l_{x_2}, l_{y_2}, l_{x_3}, l_{y_3}, l_{x_4}, l_{y_4}\}$ are the ground-truth box coordinates and 2D vertices associated with the positive anchor, respectively. $\hat{t}_j$ and $\hat{l}_j$ are the predicted ones. λ_1 , λ_2 , and λ_3 are the hyper-parameters to balance these losses, and we empirically set to 4, 0.8, and 0.1, respectively.

Remark. Although our VertexNet predicts both bounding boxes and vertices of LPs, it can work smoothly with only vertex annotations since the boxes can be generated according to the vertices. The vertices imply the geometric information of LPs captured from oblique views, and the time cost of vertex annotations is equivalent to that of bounding boxes. Therefore, we think that vertex annotations are more suitable for LP detection and recognition.

5.2.2 SCR-Net for LP Recognition

Different from YOLO-based detection methods [66, 94] or semantic segmentation method [103] that requires character-level annotations, we consider LP recognition as a classification problem for the predefined characters. First, the LP is resampled

from the finest input image and rectified to high resolution (64×256) according to its vertices. This step implicitly lays out and normalizes the locations of characters. Then, the rectified LP image is fed to SCR-Net. SCR-Net predicts the characters via a forward pass. In this section, we describe LP rectification, architecture, and training method.

5.2.2.1 License Plate Rectification

We resample and rectify LPs according to the predicted vertices from VertexNet. To be specific, the rectification process is achieved by perspective transformation which generates a bird's-eye view of the LPs regardless of the original camera angles. Suppose there is a vertex in an image with coordinates $[u, v, 1]$. Thus we can project the original vertex into the target 3D space by:

$$[\rho', \eta', \xi'] = [u, v, 1] \times M = [u, v, 1] \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & 1 \end{bmatrix}, \quad (5.7)$$

where M denotes the transformation matrix with eight unknown variables. Next, a squeeze operation is used in order to express the target vertex in a plane which is perpendicular to camera lens. Specifically, the target vertex coordinates $[\rho, \eta]$ are $\rho = \rho' / \xi'$ and $\eta = \eta' / \xi'$.

The projection between four vertices in the original plane and the target plane constructs eight equations through Eq. (5.7). It can be solved with linear algebra. Finally, we apply the same projection to all the pixels in the original plane. To ensure all characters are included in rectification, we enlarge the sampling height and width to $1.25h_l$ and $1.25w_l$, where h_l and w_l denote the height and width of an LP, respectively. Note that the LP rectification not only rectifies the view angle but also normalize the resampled LP to a predefined size, as shown in the step (3) of Fig. 5.4.

5.2.2.2 Architecture of SCR-Net

Taking efficiency into account, we perform a forward-pass CNN rather than use character segmentation or RNNs. The architecture of SCR-Net is shown in Fig. 5.4.

The detailed descriptions of SCR-Net are shown in Table 5.2. We employ the *BN-Conv3-BN-ReLU-Conv3-BN-Dynamic* structure in the residual branch of S-Stage-1, 2, and 3, which effectively decreases overfitting, in which the dynamic regularization will be presented in Section 5.2.3. Moreover, to specialize in the left-to-right format of LPs, we propose a horizontal encoding technique to squeeze features maps. In particular, we implement it in the S-Conv-2 and S-Stage-4 structures. In S-Conv-2, the output feature maps of S-Stage-3 (with the size of 8×32) are encoded by an 8×1 -s1 convolutional filter, compressing the feature maps to the size of 1×32 . In S-Stage-4, we adopt two residual blocks [22] followed by Conv-BN-ReLU to extract LP features where all convolutional filters use a 1×3 kernel. By doing so, the feature maps are vertically squeezed and encoded to 1D horizontal features with the size of 1×32 , fitting the left-to-right format.

TABLE 5.2: Descriptions of SCR-Net. “Size” and “CH” are the output size and channel of feature maps, respectively. Conv3: 3×3 convolution. BN: batch normalization. Conv4: 4×4 -s2 convolution for downsampling the feature maps by a factor of 2. ReLU: rectified linear unit. Dy: dynamic regularization. Conv 8×1 : 8×1 -s1 convolution for a horizontal encoding. Conv 1×3 : 1×3 -s1 convolution for 1×32 features encoding. Res-block: residual block [22] with the branch of two Conv 1×3 -BN-ReLUs. $\{ \cdot \} \times x$ denotes performing the $\{ \cdot \}$ structure by x times.

Name	Size	Ch	Descriptions
Input	64×256	3	Input image
S-Conv-1	32×128	16	Conv4-BN-ReLU
S-Stage-1	32×128	93	$\{\text{BN-Conv3-BN-ReLU-Conv3-BN-Dy}\} \times 4$
S-Stage-2	16×64	176	$\{\text{BN-Conv3-BN-ReLU-Conv3-BN-Dy}\} \times 4$
S-Stage-3	8×32	256	$\{\text{BN-Conv3-BN-ReLU-Conv3-BN-Dy}\} \times 4$
S-Conv-2	1×32	256	Conv3-BN-ReLU-Conv 8×1 -BN-ReLU
S-Stage-4	1×32	256	$\{\{\text{Res-block}\} \times 2\text{-Conv}1 \times 3\text{-BN-ReLU}\} \times 2$
Classifiers	1×1	N	N is the output size of the classifier.

Moreover, a weight-sharing classifier is proposed to balance training samples and address the issue of small-scale datasets, such as AOLP [101]. Usually, CNN-based models employ fully-connected layers as a classifier to recognize the characters on the predefined positions of LPs [3]. However, this manner cannot handle small-scale datasets since the probability of a specific character appearing in a specific position is low, and even the character never appears in that position. Our weight-sharing classifier utilizes the shared weights of convolutional layers such that the classifier can spot all character instances in the dataset regardless of their positions. Fig. 5.6 shows three weight-sharing classifiers for three types of characters on the CCPD [3] dataset. The output features of S-Stage4 are fed to the weight-sharing classifier, which produces probabilities of each character. Each classifier is implemented by a

FIGURE 5.6: Implementation of the weight-sharing classifiers.

convolutional layer with the kernel size of 1×8 and the stride of 1×4 . For small-scale datasets, like AOLP [101], only one classifier can handle six-position characters. This structure also decreases parameter consumption compared with fully-connected layers (see Section 5.3.4.2). Meanwhile, the weight-sharing classifier avoids applying additional character segmentation or RNNs for the small-scale datasets.

SCR-Net-Fast. Like VertexNet-Fast, we also propose a fast version of SCR-Net, named SCR-Net-Fast. Specifically, referring to Table 5.2, we change the number of channels of S-Stage-1, 2, 3, S-Conv-2, and S-Stage-4 to 53, 91, 128, 128, and 128, respectively. Besides, we change 4 blocks in S-Stage-1, 2, and 3, to 2 blocks. This shallow architecture improves inference speed with a little performance decrease (see 5.3.4.1).

5.2.2.3 Training of SCR-Net

We generate pairs of training examples according to the labeled LP vertices and the corresponding characters. The LP image is normalized to the size of 64×256 . We predefine the output layout for CCPD [3] and AOLP [101] datasets. For CCPD, there are three character types arranged on seven places in order, i.e., 34 Chinese characters (provinces) and 25 English characters (letters) in the first and second places of LPs, respectively, followed by 35 English+Number characters in the last five places. Therefore, the output size of the classifier (N in Table 5.2) is $34 + 25 + 5 \times 35 = 234$, as shown in Fig. 5.6. Similarly, we define 35 English+Number characters arranged on six places for AOLP, so the output size is 210.

The cross-entropy loss is employed for predicting the characters for each character type, i.e.,

$$L = -y_i \cdot \log(F(x_i, \Omega)), \quad (5.8)$$

where y_i is the ground-truth one-hot vector, $F(x_i, \Omega)$ is the prediction of SCR-Net, and “ $\cdot$ ” is the inner product. In $F(x_i, \Omega)$, x_i is an input image and Ω is a set of parameters of SCR-Net.

5.2.3 Dynamic Regularization

Fig. 5.7 shows the proposed dynamic regularization in SCR-Net. The training loss is not only used to perform backpropagation but also exploited to update the amplitude of the regularization. The features are multiplied by the regularizer in the residual branch. The regularizer works as a perturbation which introduces an augmentation in feature space, so CNNs are trained by the diversity of augmented features. Additionally, the regularization amplitude is changeable with respect to the change of the training loss.

FIGURE 5.7: Proposed dynamic regularization in S-Stage-1, 2, and 3 of SCR-Net. S-Conv-2 and S-Stage-4 are omitted for simplification. F_r denotes the residual function, i.e., $BN-Conv3-BN-ReLU-Conv3-BN$. $\nabla f(loss)$ denotes a backward difference of the training loss. The dynamic regularization aims to make a self-adaptive schedule during training by adjusting the strength of the random perturbation θ . As a manner of feature augmentation, the θ introduces noises for the residual branch in the forward and backward process.

5.2.3.1 SCR-Net with Dynamic Regularization

The fixed regularization strength in the existing regularization methods, such as DropPath [142], Stochastic depth [143], Shake-Shake [144], and Shakedrop [145], ignore the human learning paradigm (e.g., the curriculum learning [146, 147] or

self-paced learning [148]). A naive way is to predefine the schedule for updating the regularization strength, such as the linear increment scheme in [149, 150], which linearly increases the regularization strength from low to high. We argue that the predefined schedule is not flexible enough to reveal the learning process.

Based on the fact that the loss of the learning system can fully provide the learning status, we propose a dynamic regularization, which is capable of adjusting the regularization strength adaptively. Our dynamic regularization for CNNs leverages the dynamics of the training loss. That is, at the beginning of training, both the training and testing losses keep decreasing. Through a certain number of iterations, the network overfits the training data, resulting in that the training loss decreases more rapidly than the testing loss. We design a regularization strategy to follow this dynamics. If the training loss drops in an iteration, the regularization strength should increase against the overfitting in the next iteration; otherwise, the regularization strength should decrease against the underfitting. In what follows, we first introduce the dynamic regularization in SCR-Net and then propose the update of the regularization strength in each iteration of the training process.

Training phase. The dynamic regularization adopted in the residual block of SCR-Net is shown in Figures 5.8(a) and (b). Specifically, a dynamic regularization unit (or random perturbation) is embedded into the residual branch. The random perturbation θ is achieved by

$$\theta = A + s_i \cdot r, \quad (5.9)$$

where A is the basic constant amplitude, s_i is the dynamic factor at the i^{th} iteration, and $r \in [-R, R]$ is the uniform random noise with the expected value $E(r) = 0$. The value of s_i is updated via the backward difference of the training loss (see Section 5.2.3.2). The regularization amplitude is proportional to $A + s_i \cdot R$. In the forward pass, the output of the $(l + 1)^{th}$ block can be expressed as:

$$x_{l+1} = x_l + (A + s_i \cdot r)F_r(x_l), \quad (5.10)$$

where F_r is the residual function. In the backward pass, θ has a different value (represented by μ in Fig. 5.8(b)) due to the random noise r .

Random noise. The range of r , i.e., R , is a hyper-parameter in the training phase. A straightforward way is to set R to be uniform inside all residual blocks. According to [143], the features of the bottom blocks should remain more than those of the

FIGURE 5.8: Residual block of SCR-Net with dynamic regularization.

top blocks. Hence, we propose a linear enhancement rule to configure this range inside the blocks. For the l^{th} block, the range denoted as R_l is given by

$$R_l = l/L, \quad (5.11)$$

where L is the total number of blocks. With the linearly increased R , the regularization strength gradually increases from the bottom layers to the top layers.

Inference phase. As shown in Fig. 5.8(c), we calculate the expected value of θ as

$$E(\theta) = E(A + s_i \cdot r) = A, \quad (5.12)$$

and obtain a forward pass for inference:

$$x_{l+1} = x_l + A \cdot F(x_l). \quad (5.13)$$

Since A is a constant, Eq. (5.13) is equivalent to the standard residual block [22].

5.2.3.2 Update of Regularization Strength

The proposed updating solution for the dynamic regularization strength is achieved by the dynamics of the training loss. In particular, the dynamic characteristic of the training loss can be model as the backward difference between the training

losses at successive iterations:

$$\nabla loss_i = loss_i - loss_{i-1}, \quad (5.14)$$

where $loss_i$ denotes the training loss at the i^{th} iteration. Although the training loss shows a downtrend in training, large fluctuations appear when sequential mini-batches are fed. To eliminate the fluctuations and obtain the overall trend of the loss, we apply a Gaussian filter to smooth it. The filtered backward difference can be rewritten as

$$\nabla f(loss_i) = f(loss_i) - f(loss_{i-1}), \quad (5.15)$$

where $f(\cdot)$ is the filtering operation defined as

$$f(loss_i) = \sum_{n=0}^N w[n] \cdot loss_{i-n}. \quad (5.16)$$

The filter length is $N + 1$. Here we use the normalized Gaussian window and formulate $w[n]$ as

$$w[n] = \frac{1}{\sqrt{2\pi}(\sigma N/2)} e^{-\frac{1}{2}\left(\frac{n-N/2}{\sigma N/2}\right)^2}, \quad (5.17)$$

where $\sigma = 0.4$, and $0 \leq n \leq N$. The standard deviation is determined by $\sigma \cdot N/2$. The dynamic factor in Eq. (5.10) is updated with respect to $\nabla f(loss_i)$, i.e.,

$$s_{i+1} = \begin{cases} s_i + \Delta s, & \nabla f(loss_i) \leq 0 \\ s_i - \Delta s, & \nabla f(loss_i) > 0 \end{cases} \quad (5.18)$$

where Δs is a small constant step for changing the regularization amplitude. In Eq. (5.18), it can be observed that if the training loss decreases ($\nabla f(loss_i) \leq 0$), the regularization amplitude increases to avoid overfitting; otherwise, it decreases to prevent underfitting. The dynamic factor keeps updating to reflect the dynamics of the training loss.

Remark. Some methods have been proposed to change the regularization strength. For instance, Zoph *et al.* [150] introduced a linear increment scheme, Scheduled-DropPath, to regularize NASNets. The probability of dropping out a path is increased linearly throughout training. Following this, DropBlock [149] employs a linearly-increased dropping rate. However, the constant or linear scheme is still a predefined rule, which cannot adapt to the training procedure and different model

size. Different from them, our dynamic scheduling exploits the dynamics of the training loss, which is applicable to different network architectures. Here we adopt it in SCR-Net, and more detailed experiments for image classification can refer to our previous work [151].

5.3 Experiments

In this section, we first introduce datasets, evaluation metrics, and implementation details. Then, we evaluate the proposed VSNet compared with state-of-the-art methods in terms of LP detection and LP recognition. Finally, we conduct ablation studies to investigate the effectiveness of the proposed modules.

5.3.1 Datasets and Evaluation Metrics

5.3.1.1 Dataset

We evaluate our methods on two benchmark datasets, i.e., Chinese city parking dataset (CCPD) [3] and application-oriented license plate dataset (AOLP) [101]. CCPD is a large-scale comprehensive LP detection and recognition benchmark on unconstrained scenes, while AOLP is a small-scale LP benchmark on constrained scenes. The detailed introduction can refer to Section 2.2.3.

CCPD. Following [3, 24], we train our model on 100k examples of CCPD-Base and test it on the rest 100k examples of CCPD-Base and the 80k examples of sub-datasets including CCPD-DB, CCPD-FN, CCPD-Rotate, CCPD-Tilt, CCPD-Weather, and CCPD-Challenge. The AOLP only contains 2,049 images, which is challenging for DNN-based classifier.

AOLP. Like [24, 91], we train our model on any two subsets and test on the remaining one. We compared our method with state-of-the-art methods in the LP recognition task.

5.3.1.2 Evaluation Metrics

For the CCPD dataset, we follow the detection and recognition metrics in [3], where the detection precision is calculated without considering the recall since each image only contains one single license plate. The detector is allowed to predict one bounding box for each image. The predicted box is true positive when its IoU with ground truth is larger than 0.7. We report the precision over the testing set (overall precision) and all subsets of CCPD. In the LP recognition task, the predicted LP is correct when its IoU is more than 0.6, and all characters on the LP are correct. The recognition accuracy is calculated by the correct LP predictions over all LPs. Similarly, for the AOLP dataset, we calculate the recognition accuracy based on the percentage of true positives. The true positive is that all characters on the LP are correct. We follow [24] to test the inference time (ms/image) on an NVIDIA GTX1080Ti GPU.

5.3.2 Implementation Details

Since VertexNet and SCR-Net employ the specific architecture, there are no off-the-shelf pre-trained weights on ImageNet, so we trained them from scratch. We trained VertexNet by 30 epochs with the batch size of 128 in a GPU, which costs only 4 hours. The learning rate was initialized to 0 and warmed up to 0.08 at the first 2000 steps. Then, we applied the cosine learning schedule to gradually reduce the learning rate to 0 at the end of the training. We used the SGD optimizer with the weight decay of 0.0005 and the momentum of 0.9. We employed vertex-based affine transformation, randomly flipping, and AutoAugment [152].

For SCR-Net, we employed the SGD optimizer with the momentum of 0.9. The learning rate was initialized to 0 and warmed up to 0.6 at the first 2000 steps. The cosine learning schedule was used to decrease the learning rate. We set the training epoch to 100 and the batch size to 128. The dynamic regularization was applied to each block of S-Stage-1, 2, and 3 (see Table 5.2). In order to improve view invariance, we augment an LP patch with the 3D transformation by vertices. We randomly enlarge the sampling height to $\bar{h}_l \in [h_l, 1.5 \times h_l]$ and the sampling width to $\bar{w}_l \in [w_l, 1.5 \times w_l]$, where h_l and w_l denote the height width of an LP, respectively. This operation makes SCR-Net robust to the imprecise LP detections of VertexNet. Moreover, we used AutoAugment [152] to diverse LP images.

5.3.3 Evaluation of License Plate Detection

5.3.3.1 Comparison with State-of-the-art License Plate Detection Methods

We first compare the proposed VertexNet with state-of-the-art LP detection methods. Since the CCPD [3] dataset contains diverse environments that other datasets do not have, the results of CCPD can be considered as a key indicator of LP detection in unconstrained environments. The precision is calculated based on a high IoU threshold of 0.7, which requires the detector to produce high-quality bounding boxes.

The results are reported in Table 5.3, which shows that generic object detectors, such as SSD300 [84], YOLO9000 [93], and Faster RCNN [83], generally perform worse than the LP-specific detection methods, such as TE2E[91], PRnet [3], MTLPR [100], and VertexNet. Among LP-specific methods, the unified frameworks, TE2E and PRnet, cannot achieve top performance. Due to the detection-oriented optimization and well-designed architecture, the detectors of the cascaded frameworks, MTLPR and VertexNet, outperform TE2E and PRnet. The proposed VertexNet improves the precision by 1.4% compared with the state-of-the-art MTLPR, achieving the best precision (99.1%). Our method obtains the best results on all subsets, especially for CCPD-FN and CCPD-Weather with more than 10% improvement compared with PRnet. Note that 1% improvement means that the method can have 1,800 true-positive detections improvement over 180k testing images. VertexNet runs at 5.7 ms per image. Moreover, with naive IB, VertexNet-Fast obtains the fastest speed at 3.4 ms per image, and its 98.8% accuracy is better than state-of-the-art detection methods.

TABLE 5.3: Comparisons of LP detection methods on the CCPD testing set. “-” means the result is not provided. “Time” means the inference time (ms/image).

Method	Overall Precision	Base (100k)	DB (20k)	FN (20k)	Rotate (10k)	Title (10k)	Weather (10k)	Challenge (10k)	Time ms
Faster RCNN [83]	92.9	98.1	92.1	83.7	91.8	89.4	81.8	83.9	56.7
YOLO9000 [93]	93.1	98.8	89.6	77.3	93.3	91.8	84.2	88.6	22.8
SSD300 [84]	94.4	99.1	89.2	84.7	95.6	94.9	83.4	93.1	24.6
TE2E [91]	94.2	98.5	91.7	83.8	95.1	94.5	83.6	93.1	310
RPnet [3]	94.5	99.3	89.5	85.3	94.7	93.2	84.1	92.8	11.7
MTLPR [100]	97.7	-	-	-	-	-	-	-	15.4
VertexNet-Fast (ours)	98.8	99.2	99.1	97.0	99.9	99.6	99.7	95.1	3.4
VertexNet (ours)	99.1	99.6	99.4	97.0	99.9	99.8	99.9	95.6	5.7

FIGURE 5.9: Qualitative results of VertexNet. Green, blue, and red bounding boxes represent ground truth, truth positive detections, and failure detections, respectively. White points denote the predicted vertices. (a)-(g) show truth positive detections with the $\text{IoU} \in [0.7, 1.0]$. (h)-(o) shows failure detections with the $\text{IoU} \in [0.0, 0.7]$.

We show some qualitative results of VertexNet in Fig. 5.9. We can observe that VertexNet is robust for unconstrained environments, such as uneven light and oblique views (see Fig. 5.9(a)-(g)). VertexNet fails in only 1,563 images over 180k testing images, i.e., 0.9% error rate, suppressing 2.3% error rate of MTLPR by a relative improvement of 60%. Some failure cases are illustrated in Fig. 5.9(h)-(o). The failure cases are divided into three aspects: 1) detections with $\text{IoU} \in [0.6, 0.7]$, 2) detections with $\text{IoU} \in [0.5, 0.6)$, and 3) detections with $\text{IoU} \in [0.0, 0.5)$, and the number of them is 1,386, 132, and 45, respectively, as shown in Fig. 5.10. Most of the failure cases have the IoU between 0.6 and 0.7. If we set the IoU threshold to 0.6 (the green line of Fig. 5.10) when calculating precision, the precision of VertexNet can achieve 99.9%. In LP recognition, we will show that most characters of the failure cases can be still recognized by SCR-Net (see Section 5.3.4.1).

5.3.3.2 Ablation Study of VertexNet

We conduct ablation studies on the CCPD dataset to evaluate the effectiveness of the proposed modules in VertexNet. The modules include IB, vertex-based augmentation, and vertex estimation. Table 5.4 shows the results of VertexNet with different settings.

TABLE 5.4: Ablation studies of VertexNet on the CCPD testing set. “Augment”: vertex-based data augmentation. “Est.”: vertex estimation. “#Params”: the number of network’s parameters. “√” means using the corresponding technique.

Backbone	IB setting	Augment	Est.	Precision	#Params
ResNet-50	-			97.5	27.2M
	w/o p -norm SE			97.4	3.19M
	w/ SE			97.5	3.27M
IB (ours)	w/ p -norm SE			98.3	3.31M
	w/ p -norm SE	√		98.8	3.31M
	w/ p -norm SE	√	√	99.1	3.48M

Integration block. As the building block of the backbone network, we first evaluated the effectiveness of IB by three settings, as shown in the 2-4th entries of Table 5.4. We also set up a strong baseline backbone, ResNet-50 [22], as shown in the 1st entry. ResNet-50 obtains 97.5% precision, but the multiple channels in top layers lead to a large number of parameters. From the 2nd entry of Table 5.4, the IB module without the p -norm SE that employs the narrow channels and shallow structure is comparable to ResNet-50 (97.4 *vs.* 97.5), which shows that the multiple channels are unnecessary for small object detection. With the SE [140] module (the 3rd entry), the performance of the IB module improves slightly. Furthermore, the IB module with the p -norm SE (the 4th entry), i.e., the whole structure of the IB module, achieves 98.3% precision. This version outperforms ResNet-50 by a significant margin of 0.8% and overtakes the state-of-the-art MTLPR [100].

Vertex-based augmentation. Vertex-based augmentation is to generate diverse views of LPs by affine transformation according to vertex supervisory information. From the 4th and 5th entries of Table 5.4, we can observe that this component boosts the detection precision from 98.3% to 98.8%. The improvement illustrates that the vertex-based augmentation is helpful for LP detection.

Vertex estimation. We compare VertexNet without/with the vertex estimation branch, as shown in the last two entries of Table 5.4. With only 5% relative parameter increases (0.17M/3.31M), the vertex estimation branch improves the

detection precision from 98.8% to 99.1%. Moreover, the estimated vertices are crucial for the LP rectification in SCR-Net. The rectified LPs can significantly enhance character recognition. See Section 5.3.4.2 for more detailed comparisons.

5.3.4 Evaluation of License Plate Recognition

5.3.4.1 Comparison with State-of-the-art License Plate Recognition Methods

We compare the proposed SCR-Net with state-of-the-art LP recognition methods on the CCPD [3] and AOLP [101] datasets.

Results on CCPD. Table 5.5 shows the comparisons of LP recognition methods on the CCPD dataset. Instead of using ground-truth vertices, the proposed SCR-Net takes the predicted vertices by VertexNet as the inputs, i.e., VSNet. The results show that VSNet is superior to state-of-the-art LP recognition methods, including RNN-based MTLPR [100] and Attentional Net [24]. VSNet obtains the best recognition accuracy (99.5%) on the overall testing set and all subsets, especially for CCPD-Challenge with more than 5% accuracy improvement. Our approach are even better than Attentional Net trained with additional synthetic data (99.5% *vs.* 98.9%). For inference time, the fast version of our approach (VertexNet-Fast + SCR-Net-Fast, i.e., VSNet-Fast) obtains the fastest inference speed at 6.7 ms per image, 43% relative improvement compared with 11.7 ms per image of RPnet [3]. This improvement comes from the resampling-based cascaded framework and compact network architectures.

We detail the result of SCR-Net by partitioning the detections by IoU. The accuracy of SCR-Net and the count of detections of VertexNet with respect to IoU interval are shown in Fig. 5.10. It can be observed that 1) more than 99% LPs can be recognized when the IoU is large than 0.7, 2) SCR-Net can obtain 95% recognition accuracy for $\text{IoU} \in [0.6, 0.7)$, and 3) SCR-Net can also correctly recognize part of detections in the case of $\text{IoU} < 0.6$.

Some successful and failure cases in challenging environments are presented in Fig. 5.11. The environments include strong light, uneven light, nighttime, occlusion, oblique view, and blur. SCR-Net handles most of such conditions and contains

TABLE 5.5: Comparisons of LP recognition methods on the CCPD testing set. “-” means the result is not provided. “Time” means the inference time (ms/image). “*” means the recognition results of HC [23]. “**” means the method trained with synthetic data. In our method, the inference time of VertexNet+SCR-Net (i.e., VSNet) and only SCR-Net are shown outside and inside the parentheses, respectively.

Method	Overall Accuracy	Base (100k)	DB (20k)	FN (20k)	Rotate (10k)	Title (10k)	Weather (10k)	Challenge (10k)	Time ms
Faster RCNN* [83]	92.8	97.2	94.4	90.9	82.9	87.3	85.5	76.3	57.6
YOLO9000* [93]	93.7	98.1	96.0	88.2	84.5	88.5	87.0	80.5	23.8
SSD300* [84]	95.2	98.3	96.6	95.9	88.4	91.5	87.3	83.8	25.6
TE2E [91]	94.4	97.8	94.8	94.5	87.9	92.1	86.8	81.2	310
RPnet [3]	95.5	98.5	96.9	94.3	90.8	92.5	87.9	85.1	11.7
DAN [153]	96.6	98.9	96.1	96.4	91.9	93.7	95.4	83.1	19.3
MORAN [33]	98.3	99.5	98.1	98.6	98.1	98.6	97.6	86.5	18.2
Attentional Net [24]	98.5	99.6	98.8	98.8	96.4	97.6	98.5	88.9	24.9
MTLPR [100]	98.8	-	-	-	-	-	-	-	15.6
Attentional Net** [24]	98.9	99.8	99.2	99.1	98.1	98.8	98.6	89.7	24.9
VSNet-Fast (ours)	99.4	99.9	99.6	99.3	99.8	99.8	99.1	93.9	6.7 (3.3)
VSNet (ours)	99.5	99.9	99.7	99.4	99.9	99.9	99.4	94.8	11.4 (5.7)

FIGURE 5.10: The accuracy of SCR-Net and the count of detections of VertexNet with respect to IoU interval.

very few failure cases. There are only 864 failure cases over 180k testing images, i.e., 0.5% error rate, defeating 1.1% error rate of Attentional Net by a relative improvement of 55%. Among failure cases, human eyes cannot even recognize some characters, like “5” recognized as “F” in uneven light. Moreover, SCR-Net can filter out false detections of VertexNet by training the randomly sampled background images. The false detections in Fig. 5.11(c) are recognized as “background”.

Results on AOLP. Table 5.6 shows the results of LP recognition methods on three subsets of the AOLP dataset. For a fair comparison, we employed the detections by [24] as the input images for SCR-Net and did not adopt LP rectification. Our SCR-Net (box) achieves the best accuracy (98.7%) among state-of-the-art LP recognition

FIGURE 5.11: Qualitative results of our SCR-Net. Error characters are marked by the red color. False detections in (c) cause false recognition, but SCR-Net can still recognize these images as the “background” rather than LPs.

methods. Especially on the RP subset, SCR-Net improve the accuracy by 7.6% when compared with Attentional Net [24]. With vertex information, SCR-Net (vertex) improves the accuracy to 99.7%.

TABLE 5.6: Comparisons of LP recognition methods on the AOLP dataset. “box” means the predicted boxes by Attentional Net [24]. “vertex” is the ground truth vertex. “FC” is the fully-connected classifier.

Method	Overall	AC	LE	RP
#Images	(2049)	(681)	(757)	(611)
TE2E [91]	94.5	95.3	96.6	83.7
Attentional Net [24]	96.1	97.3	98.3	91.9
SCR-Net (box)	98.7	97.9	98.9	99.5
SCR-Net (vertex)	99.7	99.4	99.9	99.7
SCR-Net w/ FC	81.3	97.1	53.9	97.6

5.3.4.2 Ablation Study of SCR-Net

We conduct ablation studies to evaluate and analyze the proposed modules in SCR-Net, i.e., resampling-based cascaded framework, vertex-based LP rectification, robustness of SCR-Net, and weight-sharing classifier.

Resampling-based cascaded framework. In TE2E [91] and RPnet [3], the unified framework only processes one input size. However, designing the small-size input may lose the details of characters, while designing the large-size input will increase the inference time. In contrast, our resampling-based cascaded framework decouples the input-size design of VertexNet and SCR-Net, i.e., VertexNet processes small-size images for fast inference regardless of the details of characters, and SCR-Net resamples LP patches from the finest input images. Here we evaluated the importance of the resampling process. If the resampling is disabled (like the unified manner), SCR-Net and VertexNet can only access input images with the size of 256×256 , and the LP patches are sampled from those images, named SCR-Net-Small. This setting causes a dramatic drop in the recognition accuracy (94.0%), compared with 99.5% accuracy of SCR-Net. Some sampled LP patches with their predictions by SCR-Net and SCR-Net-Small are shown in Fig. 5.12. We can observe that due to the small-size input of SCR-Net-Small, the sampled LPs lose the details on some characters. Therefore, SCR-Net-Small makes many mistakes when SCR-Net generates correct predictions. The result illustrates that our resampling process is important to SCR-Net.

FIGURE 5.12: Some sampled LP patches with their predictions by SCR-Net and SCR-Net-Small. Error characters are marked by the red color.

Vertex-based LP rectification. LP rectification is crucial for LP recognition in our CNN-based architecture because the rectified LPs imply the character's layout.

TABLE 5.7: Ablation studies of SCR-Net with different input types. “Box” and “Vertex” mean the LPs are sampled according to bounding boxes and vertices, respectively. “Predicted” and “GT” mean the boxes (or vertices) are generated by VertexNet and ground truth, respectively. “revised metric” in the parentheses means removing the condition of IoU larger than 0.6.

Input type	Predicted Box (revised metric)	Predicted Vertex (revised metric)	GT Box	GT Vertex
Accuracy	98.51 (98.56)	99.52 (99.58)	98.79	99.65

The more accurate the rectified LPs is the best character recognition. The LP rectification is based on the vertex estimation in VertexNet. We compared two types in SCR-Net, which were separately trained by the rectified LPs (resampled LP images by ground-truth vertices) and non-rectified LPs (resampled LP images by ground-truth bounding boxes). In inference, the first model performs LP rectification according to the predicted vertices by VertexNet, while the second one directly sample LPs by the predicted bounding boxes. We evaluated our method on the CCPD testing set. As illustrated in the first two entries (columns) of Table 5.7, the rectified LPs by vertices improve the accuracy from 98.51% to 99.52%, which validates the effectiveness of LP rectification. Additionally, the results in Table 5.6 show that SCR-Net (vertex) also benefits from the vertices on the AOLP dataset.

Robustness of SCR-Net. SCR-Net may be affected by the biased results of VertexNet. Here we modify the recognition metric of CCPD in Section 5.3.1.2 by removing the condition of IoU larger than 0.6. The increased accuracy in the parentheses of Table 5.7 shows that even though the predicted boxes have low IoU, SCR-Net can make correct recognition for those inaccurate boxes. Then, we conducted two comparative experiments for the predicted vertices (resp. boxes) and ground-truth vertices (resp. boxes) on the CCPD testing set. From Table 5.7, we can observe that the predicted vertices (resp. boxes) only cause $99.65\% - 99.58\% = 0.07\%$ (resp. $98.79\% - 98.56\% = 0.23\%$) more errors compared with the ground-truth vertices (resp. boxes). The results illustrate that SCR-Net is robust to the output of VertexNet, especially for the predicted vertices.

Weight-sharing classifier. The weight-sharing classifier aims to increase training samples for the predefined position of LP images. In previous classification works [3, 23], the fully-connected (FC) layer is commonly used as a classifier. We compared our classifier with the FC layer on the small-scale dataset, i.e., AOLP. Specifically, we replace the weight-sharing classifier with the FC layer. The last two entries of Table

5.6 illustrate that the weight-sharing classifier is superior to the FC counterpart by a significant margin, i.e., 99.7% *vs.* 81.1%. For the LE subset, the sharp drop in performance is caused by the lack of training samples for the FC layer. For example, the “S” appears 20 times in the first place of LPs on the LE subset, but there is only one case on the training set, leading to 20 errors in testing. Regardless of the “S” position, there are 41 cases in the training set when the weight-sharing classifier is used. With the weight-sharing classifier, our CNN-based model can handle small-scale datasets rather than seek help from RNNs or character segmentation. Moreover, the weight-sharing classifier only requires 4% parameters of the FC layer (63k *vs.* 1,721k).

5.4 Chapter Summary

This chapter has investigated the insights in designing high-performing ALPR systems and proposed the novel VSNet. In VSNet, the proposed VertexNet and SCR-Net are connected in a resample-based cascaded manner, thereby improving the inference speed. With a novel architecture design, such as integration block, vertex estimation, horizontal encoding, dynamic regularization, weight-sharing classifier, and dynamic regularization, our VSNet achieves the best performance on the CCPD and AOLP datasets. We validated that taking advantage of vertex information and the weight-sharing classifier is crucial for the CNN model to perform character recognition rather than adopt additional character segmentation modules or RNNs.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

This thesis has presented three dense prediction tasks in complex visual scenes, i.e., image restoration, crowd detection and counting, and license plate recognition.

First, aiming at light scattering scenes, we propose two methods to estimate transmission and ambient light for image restoration, respectively. (1) We introduce a novel adaptive attenuation-curve prior, where the pixels from the same cluster can form a power-function curve in RGB space, which models the light attenuation process in various scattering scenes. With the prior, we cluster the pixels of an image into the curves and estimate each pixel’s transmission, thereby restoring the image. We exploit the saturation constraints to adjust three color channels’ transmission to reduce the over-saturation and noise in the restored images. (2) We propose a novel deep learning-based approach for ambient light estimation based on the fact that ambient light exists in the infinite distance. We synthesize hazy outdoor images and train an encoder-decoder to predict the distant region such that ambient light is calculated by the pixel values inside the distant region. The qualitative and quantitative results on the benchmarks demonstrate that our proposed methods outperform state-of-the-art image restoration methods. The proposed approaches exhibit a good generalization capability for natural outdoor images. In addition, the integration of the two methods further improves image restoration performance.

Second, for detecting and counting crowds, we present two CNNs trained with only point-level annotations, i.e., Crowd-DCNet for crowd detection and MSD regressor for crowd counting. (1) We introduce a novel self-training approach to train Crowd-DCNet such that it can accurately detect and count the objects in dense crowds. Our training method starts with the pseudo labels generated by the locally-uniform distribution assumption. We then iteratively train the network by the crowdedness-aware loss and updates those pseudo labels by the confidence and order-aware refinement scheme. Besides, to address densely crowded scenes, we propose an effective decoding method to improve the detector’s representation ability. (2) Bypassing object detection, we propose the MSD regressor to estimate the multi-scale density maps of crowds, then calculate the count over the full-resolution density map. Both of the CNNs follow a coarse-to-fine and end-to-end manner to predict the discrete or continuous pixel-wise labels. The extensive experiments on six benchmark datasets show that Crowd-DCNet obtains superior performance among point-supervised methods, and the MSD regressor yields state-of-the-art counting results in low-resolution surveillance images.

Third, for small characters in an image, we introduce a novel VSNet for automatic license plate recognition. There are four improvements in VSNet. (1) VSNet is composed of VertexNet for license plate detection and SCR-Net for license plate recognition, in which they are cascaded in an efficient resampling-based manner. (2) Without additional character segmentation module or RNNs, the proposed VSNet adopts a CNN-based structure only, which is more efficient for the real-time system. (3) The detection performance of VertexNet is improved significantly by the proposed integration block and vertex-estimation branch. (4) The recognition performance of SCR-Net is boosted by the proposed license plate rectification module, horizontal encoding technique, weight-sharing classifier, and dynamic regularization. The experimental results show that VSNet outperforms state-of-the-art methods in both speed and accuracy. VSNet can handle complex unconstrained scenes, such as oblique views, uneven illumination, and different weather conditions.

6.2 Future Work

6.2.1 High-level Vision Tasks in Poor Weather

Despite the remarkable improvement of image restoration made in recent years (as shown in Chapter 3), dehazing does not introduce new information to an image. State-of-the-art dehazing methods still have a huge gap to boost high-level vision tasks in unconstrained environments [108]. Other poor weather scenes, like rain streaks [154], also introduce various blurring effects and noises in an image, which poses a challenge for object detection and semantic segmentation. Therefore, we should further investigate some techniques to improve the performance of high-level vision tasks, such as object classification, detection, and segmentation, in poor weather scenes. We believe that the main problem is the domain shift between training images and hazy images. Most of the large-scale datasets mainly contains clear images without degradation [108], e.g., ImageNet [34] and COCO [155]. There are two potential directions for future studies.

Poor weather image generation. It is difficult to collect and label hazy images for training deep learning models due to burdensome handle-labeling. The poor weather image generator is to synthesize hazy images from large-scale clear images with labels. With the success of GANs [156], Image-to-Image translation [62], and Cycle GAN [75], we can use GANs to generate high-resolution real hazy images. The image generation would resolve the bottleneck of handle-labeling and provide abundant annotated training data.

Domain adaptation. Domain adaptation aims to tackle the domain shift problems by transferring pre-trained models from label-rich source domains to label-limited or label-free target domains. It is reliant on domain-invariant alignment of the image or feature distributions. Recently, a detection method [157] has been proposed to address this cross-domain problem. Domain adaptation would be used and improved for the image recognition tasks in unconstrained environments.

6.2.2 Learning from Less Labeled Data

Chapter 4 has shown the potential of weakly-supervised learning for dense object detection, i.e., applying point annotations to learn the object's bounding box,

by which laborious labeling works can be dramatically reduced. However, the performance of weakly-supervised learning is not on par with state-of-the-art fully-supervised learning. Researchers still need efforts to narrow the gap. Besides, due to small-scale labeled data, crowd detection and crowd counting methods only perform well on similar domain data. It is necessary to make use of the massive unlabeled data to improve the model’s generalizability. We propose two potential research directions as follows.

Point-supervised dense object detection. In addition to training a network by exploring the properties of incomplete and inaccurate labeled data (see Chapter 4), other learning methods, like active learning [158] with human intervention, curriculum learning [146], and transductive learning [159], are promising to point-supervised dense object detection with a great potential.

Self-training on both labeled and unlabeled data. Xie *et al.* [160] proposed a noisy student training method for image classification, which uses the self-training and distillation idea. The method employs a teacher network initialized by ImageNet to predict “pseudo labels” for large-scale unlabeled data. It then iteratively trains a student network by labeled and unlabeled examples with noise added. Such a self-training method can be extended to dense prediction tasks, like object detection and crowd counting, though the extension may be nontrivial.

6.2.3 Multi-task Learning for Dense Prediction

In our proposed methods, the task-related dense prediction can solve a single task, such as image restoration, crowd detection, crowd counting, and character recognition. We adopted some post-processing techniques or separate networks to tackle them in an isolation manner when facing multiple tasks. For example, we proposed a crowd detection method to detect object instances, but we need to use a post-processing method (like thresholding) to count those detections. Conversely, regression-based methods predict the count directly, but they cannot provide objects’ locations. The post-processing methods are usually rule-based and sensitive to some hyper-parameter settings, limiting the model’s generalization ability. Besides, we proposed separate networks for license plate detection and recognition. Although they are better than the recent unified network, separate networks show accumulated inference time. Like humans, the machine is required to

solve multi-tasks concurrently. Vandenhende *et al.* [161] reviewed recent multi-task learning (MTL) works and showed their potential for improving performance with shared complementary information. Therefore, it is necessary to investigate multi-task learning for dense prediction tasks. We introduce several potential research directions as follows.

Taxonomy of MTL methods. The research is ongoing in MTL methods' taxonomy, and the highly efficient network architectures are improving the performance and reducing memory footprint. From FCN [5] to Multi-Scale Task Interaction Networks [162], it is always meaningful to explore novel network architectures and the reasons behind the improvement.

Task balancing. MTL methods are required to balance the tasks such that all tasks can be performed smoothly. During training, the optimization process should be investigated in case of one or more tasks dominating the network's parameters [161]. It is a trend of integrating task balancing methods into the whole network's training process.

Multi-domain learning. Current MTL methods focused on similar visual domain tasks, such as semantic segmentation and depth estimation, both of which are the scene understanding task. The multi-domain learning performed in a unified manner (like [163]) is valuable for research. For example, a video surveillance system employs a unified framework to execute object detection, multi-object tracking, and crowd counting. This may be a future research direction.

List of Author's Publications

Journal Articles

- **Yi Wang**, Hui Liu and Lap-Pui Chau. Single underwater image restoration using adaptive attenuation-curve prior. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 65(3): 992-1002, 2018.
- **Yi Wang**, Zhen-Peng Bian, Junhui Hou, and Lap-Pui Chau. Convolutional neural networks with dynamic regularization. *IEEE Transactions on Neural Networks and Learning Systems*, 32(5): 2299-2304, 2021.
- **Yi Wang**, Junhui Hou, Xinyu Hou, and Lap-Pui Chau. A self-training approach for point-supervised object detection and counting in crowds. *IEEE Transactions on Image Processing*, 30: 2876-2887, 2021.
- **Yi Wang**, Zhen-Peng Bian, Yunhao Zhou, and Lap-Pui Chau. Rethinking and designing a high-performing automatic license plate recognition approach. *IEEE Transactions on Intelligent Transport System*, 2021. (Accepted)

Conference Proceedings

- **Yi Wang**, Hui Liu, and Lap-Pui Chau. Single underwater image restoration using attenuation-curve prior. *In Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1-4, 2017.
- **Yi Wang**, Lap-Pui Chau, and Xiaoxi Ma. Airlight estimation based on distant region segmentation. *In Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1-5, 2019.

- **Yi Wang**, Junhui Hou, and Lap-Pui Chau. Object counting in video surveillance using multi-scale density map regression. *In Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2422-2426, 2019.
- **Yi Wang**, Xinyu Hou, and Lap-Pui Chau. Dense point prediction: a simple baseline for crowd counting and localization. *IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, 2021. (Accepted)

Bibliography

- [1] Yuval Bahat and Michal Irani. Blind dehazing using internal patch recurrence. In *IEEE International Conference on Computational Photography*, pages 1–9. IEEE, 2016. [xix](#), [xx](#), [2](#), [50](#), [51](#), [52](#)
- [2] Qi Wang, Junyu Gao, Wei Lin, and Xuelong Li. Nwpu-crowd: A large-scale benchmark for crowd counting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. [xix](#), [xxvi](#), [22](#), [59](#), [71](#), [72](#), [78](#), [79](#)
- [3] Zhenbo Xu, Wei Yang, Ajin Meng, Nanxue Lu, Huan Huang, Changchun Ying, and Liusheng Huang. Towards end-to-end license plate detection and recognition: A large dataset and baseline. In *Proceedings of the European Conference on Computer Vision*, pages 255–271, 2018. [xix](#), [xxvi](#), [2](#), [19](#), [20](#), [23](#), [86](#), [88](#), [89](#), [93](#), [96](#), [97](#), [102](#), [103](#), [104](#), [107](#), [108](#), [110](#), [111](#)
- [4] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86 (11):2278–2324, 1998. [xix](#), [13](#), [14](#)
- [5] 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*, pages 3431–3440, 2015. [xix](#), [14](#), [15](#), [117](#)
- [6] Shuo Yang, Ping Luo, Chen-Change Loy, and Xiaoou Tang. Wider face: A face detection benchmark. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5525–5533, 2016. [xix](#), [xxi](#), [xxv](#), [xxvi](#), [22](#), [56](#), [58](#), [59](#), [71](#), [72](#), [73](#), [75](#), [77](#), [78](#), [79](#), [80](#)
- [7] Yingying Zhang, Desen Zhou, Siqin Chen, Shenghua Gao, and Yi Ma. Single-image crowd counting via multi-column convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 589–597. IEEE, 2016. [xix](#), [xxi](#), [xxii](#), [xxv](#), [xxvi](#), [3](#), [17](#), [19](#), [22](#), [57](#), [58](#), [59](#), [62](#), [63](#), [71](#), [72](#), [73](#), [75](#), [78](#), [79](#), [80](#), [81](#)
- [8] Meng-Ru Hsieh, Yen-Liang Lin, and Winston H Hsu. Drone-based object counting by spatially regularized regional proposal network. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4145–4153, 2017. [xix](#), [xxi](#), [xxvi](#), [22](#), [23](#), [59](#), [71](#), [72](#), [78](#), [79](#), [81](#)

- [9] Nicholas Carlevaris-Bianco, Anush Mohan, and Ryan M Eustice. Initial results in underwater single image dehazing. In *IEEE OCEANS*, pages 1–8, 2010. [xx](#), [13](#), [21](#), [43](#), [44](#), [45](#), [46](#), [48](#)
- [10] P Drews, Erickson Nascimento, F Moraes, S Botelho, and M Campos. Transmission estimation in underwater single images. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 825–830, 2013. [xx](#), [12](#), [43](#), [44](#), [45](#), [46](#), [48](#)
- [11] Yan-Tsung Peng and Pamela C Cosman. Underwater image restoration based on image blurriness and light absorption. *IEEE Transactions on Image Processing*, 26(4):1579–1594, 2017. [xx](#), [13](#), [43](#), [44](#), [45](#), [46](#), [47](#), [48](#)
- [12] Yi Wang, Hui Liu, and Lap-Pui Chau. Single underwater image restoration using attenuation-curve prior. In *IEEE International Symposium on Circuits and Systems*, pages 1–4, 2017. [xx](#), [37](#), [43](#), [45](#), [46](#), [48](#)
- [13] Pascal Getreuer. Automatic color enhancement (ACE) and its fast implementation. *Image Processing On Line*, 2:266–277, 2012. [xx](#), [28](#), [46](#), [47](#)
- [14] Kaiming He, Jian Sun, and Xiaoou Tang. Single image haze removal using dark channel prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(12):2341–2353, 2011. [xx](#), [2](#), [5](#), [10](#), [11](#), [28](#), [29](#), [39](#), [49](#), [50](#), [51](#), [52](#)
- [15] Dana Berman, Shai Avidan, et al. Non-local image dehazing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1674–1682, 2016. [xx](#), [13](#), [31](#), [37](#), [39](#), [44](#), [48](#), [49](#), [50](#)
- [16] Dana Berman, Tali Treibitz, and Shai Avidan. Air-light estimation using haze-lines. In *IEEE International Conference on Computational Photography*, pages 1–9. IEEE, 2017. [xx](#), [13](#), [50](#), [51](#), [52](#)
- [17] He Zhang and Vishal M Patel. Densely connected pyramid dehazing network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. [xx](#), [16](#), [41](#), [42](#), [51](#), [52](#), [53](#)
- [18] Yuhong Li, Xiaofan Zhang, and Deming Chen. Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1091–1100. IEEE, 2018. [xxii](#), [17](#), [19](#), [56](#), [77](#), [79](#), [80](#), [81](#), [82](#)
- [19] Yuting Liu, Miaoqing Shi, Qijun Zhao, and Xiaofang Wang. Point in, box out: Beyond counting persons in crowds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6469–6478, 2019. [xxv](#), [18](#), [19](#), [22](#), [56](#), [65](#), [71](#), [72](#), [73](#), [75](#), [76](#), [77](#), [78](#), [80](#)
- [20] Zenglin Shi, Pascal Mettes, and Cees GM Snoek. Counting with focus for free. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4200–4209, 2019. [xxv](#), [17](#), [56](#), [63](#), [72](#), [73](#), [75](#), [76](#), [79](#), [80](#)

- [21] Deepak Babu Sam, Skand Vishwanath Peri, Mukuntha Narayanan Sundararaman, Amogh Kamath, and R. Venkatesh Babu. Locate, size and count: Accurately resolving people in dense crowds via detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. [xxvi](#), [18](#), [19](#), [22](#), [56](#), [57](#), [71](#), [72](#), [76](#), [77](#), [78](#), [80](#)
- [22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. [xxvi](#), [14](#), [60](#), [90](#), [91](#), [96](#), [100](#), [106](#)
- [23] Jakub Špaňhel, Jakub Sochor, Roman Juránek, Adam Herout, Lukáš Maršík, and Pavel Zemčík. Holistic recognition of low quality license plates by cnn using track annotated data. In *IEEE International Conference on Advanced Video and Signal Based Surveillance*, pages 1–6. IEEE, 2019. [xxvi](#), [108](#), [111](#)
- [24] Linjiang Zhang, Peng Wang, Hui Li, Zhen Li, Chunhua Shen, and Yanning Zhang. A robust attentional framework for license plate recognition in the wild. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–10, 2020. [xxvi](#), [20](#), [87](#), [88](#), [89](#), [102](#), [103](#), [107](#), [108](#), [109](#)
- [25] Milan Sonka, Vaclav Hlavac, and Roger Boyle. *Image processing, analysis, and machine vision*. Nelson Education, 2014. [1](#), [9](#)
- [26] Antonio Brunetti, Domenico Buongiorno, Gianpaolo Francesco Trotta, and Vitoantonio Bevilacqua. Computer vision and deep learning techniques for pedestrian detection and tracking: A survey. *Neurocomputing*, 300:17–33, 2018. [1](#)
- [27] Joel Janai, Fatma Güney, Aseem Behl, Andreas Geiger, et al. Computer vision for autonomous vehicles: Problems, datasets and state of the art. *Foundations and Trends® in Computer Graphics and Vision*, 12(1–3):1–308, 2020. [1](#)
- [28] M Sami Zitouni, Harish Bhaskar, J Dias, and Mohammed E Al-Mualla. Advances and trends in visual crowd analysis: A systematic survey and evaluation of crowdmodelling techniques. *Neurocomputing*, 186:139–159, 2016. [1](#)
- [29] Harold C Burger, Christian J Schuler, and Stefan Harmeling. Image denoising: Can plain neural networks compete with bm3d? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2392–2399. IEEE, 2012. [1](#), [10](#)
- [30] Miao Yang, Jintong Hu, Chongyi Li, Gustavo Rohde, Yixiang Du, and Ke Hu. An in-depth survey of underwater image enhancement and restoration. *IEEE Access*, 7:123638–123657, 2019. [2](#), [3](#)
- [31] Zhong-Qiu Zhao, Peng Zheng, Shou-tao Xu, and Xindong Wu. Object detection with deep learning: A review. *IEEE Transactions on Neural Networks and Learning Systems*, 30(11):3212–3232, 2019. [2](#), [56](#)

- [32] Yu Xiang, Alexandre Alahi, and Silvio Savarese. Learning to track: Online multi-object tracking by decision making. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4705–4713, 2015. [2](#), [56](#)
- [33] Canjie Luo, Lianwen Jin, and Zenghui Sun. Moran: A multi-object rectified attention network for scene text recognition. *Pattern Recognition*, 90:109–118, 2019. [2](#), [108](#)
- [34] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Conference on Neural Information Processing Systems*, pages 1097–1105, 2012. [3](#), [14](#), [71](#), [115](#)
- [35] Iacopo Masi, Yue Wu, Tal Hassner, and Prem Natarajan. Deep face recognition: A survey. In *SIBGRAPI Conference on Graphics, Patterns and Images*, pages 471–478. IEEE, 2018. [3](#)
- [36] Zhihao Wang, Jian Chen, and Steven CH Hoi. Deep learning for image super-resolution: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. [3](#)
- [37] Rene Ranftl, Vibhav Vineet, Qifeng Chen, and Vladlen Koltun. Dense monocular depth estimation in complex dynamic scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4058–4066, 2016. [3](#), [15](#)
- [38] Jifeng Dai, Kaiming He, and Jian Sun. Instance-aware semantic segmentation via multi-task network cascades. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3150–3158. IEEE, 2016. [3](#), [58](#), [67](#)
- [39] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019. [3](#), [15](#), [18](#), [61](#)
- [40] Yoav Y Schechner and Nir Karpel. Clear underwater vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 536–543, 2004. [5](#), [26](#)
- [41] Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Semantic foggy scene understanding with synthetic data. *International Journal of Computer Vision*, 126(9):973–992, 2018. [5](#), [26](#), [42](#)
- [42] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgb-d images. In *Proceedings of the European Conference on Computer Vision*, pages 746–760. Springer, 2012. [6](#), [42](#)
- [43] Raanan Fattal. Single image dehazing. *ACM Transactions on Graphics*, 27(3), 2008. [10](#), [11](#), [21](#), [44](#)

- [44] Robby T Tan. Visibility in bad weather from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008. [11](#)
- [45] Jean-Philippe Tarel and Nicolas Hautiere. Fast visibility restoration from a single color or gray level image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2201–2208. IEEE, 2009. [11](#), [12](#)
- [46] Gaofeng Meng, Ying Wang, Jiangyong Duan, Shiming Xiang, and Chunhong Pan. Efficient image dehazing with boundary constraint and contextual regularization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 617–624, 2013. [12](#)
- [47] Alona Golts, Daniel Freedman, and Michael Elad. Unsupervised single image dehazing using dark channel prior loss. *IEEE Transactions on Image Processing*, 29:2692–2701, 2019. [12](#)
- [48] John Y Chiang and Ying-Ching Chen. Underwater image enhancement by wavelength compensation and dehazing. *IEEE Transactions on Image Processing*, 21(4):1756–1769, 2012. [12](#)
- [49] Adrian Galdran, David Pardo, Artzai Picón, and Aitor Alvarez-Gila. Automatic red-channel underwater image restoration. *Journal of Visual Communication and Image Representation*, 26:132–145, 2015. [12](#), [21](#), [30](#), [44](#)
- [50] Guojia Hou, Jingming Li, Guodong Wang, Huan Yang, Baoxiang Huang, and Zhenkuan Pan. A novel dark channel prior guided variational framework for underwater image restoration. *Journal of Visual Communication and Image Representation*, 66:102732, 2020. [12](#)
- [51] Adrian Galdran, Aitor Alvarez-Gila, Alessandro Bria, Javier Vazquez-Corral, and Marcelo Bertalmío. On the duality between retinex and image dehazing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8212–8221, 2018. [12](#), [28](#)
- [52] Raanan Fattal. Dehazing using color-lines. *ACM Transaction on Graphics*, 34(13), 2014. [12](#), [21](#), [32](#), [49](#)
- [53] Zeev Farbman, Raanan Fattal, Dani Lischinski, and Richard Szeliski. Edge-preserving decompositions for multi-scale tone and detail manipulation. *ACM Transactions on Graphics*, 27(3), 2008. [13](#), [37](#)
- [54] Manjit Kaur, Dilbag Singh, Vijay Kumar, and Kehui Sun. Color image dehazing using gradient channel prior and guided l0 filter. *Information Sciences*, 521:326–342, 2020. [13](#)
- [55] Jingchun Zhou, Zhenzhen Liu, Weidong Zhang, Dehuan Zhang, and Weishi Zhang. Underwater image restoration based on secondary guided transmission map. *Multimedia Tools and Applications*, 80(5):7771–7788, 2021. [13](#)

- [56] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016. [13](#)
- [57] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2014. [14](#), [17](#)
- [58] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015. [14](#)
- [59] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015. [14](#)
- [60] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241. Springer, 2015. [14](#), [15](#), [28](#), [41](#)
- [61] Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. Deep layer aggregation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2403–2412, 2018. [14](#)
- [62] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1125–1134, 2017. [14](#), [16](#), [25](#), [115](#)
- [63] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Minghui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. [14](#)
- [64] Bolun Cai, Xiangmin Xu, Kui Jia, Chunmei Qing, and Dacheng Tao. Dehazenet: An end-to-end system for single image haze removal. *IEEE Transactions on Image Processing*, 25(11):5187–5198, 2016. [15](#), [16](#), [28](#), [42](#)
- [65] Vishwanath A Sindagi and Vishal M Patel. A survey of recent advances in cnn-based single image crowd counting and density estimation. *Pattern Recognition Letters*, 107:3–16, 2018. [15](#), [56](#)
- [66] Rayson Laroca, Luiz A Zanlorensi, Gabriel R Gonçalves, Eduardo Todt, William Robson Schwartz, and David Menotti. An efficient and layout-independent automatic license plate recognition system based on the yolo detector. *arXiv preprint arXiv:1909.01754*, 2019. [15](#), [19](#), [87](#), [90](#), [94](#)
- [67] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448, 2015. [15](#), [62](#), [64](#), [69](#), [94](#)

- [68] Xinkun Cao, Zhipeng Wang, Yanyun Zhao, and Fei Su. Scale aggregation network for accurate and efficient crowd counting. In *Proceedings of the European Conference on Computer Vision*, pages 734–750, 2018. [16](#), [17](#), [80](#)
- [69] Wenqi Ren, Si Liu, Hua Zhang, Jinshan Pan, Xiaochun Cao, and Ming-Hsuan Yang. Single image dehazing via multi-scale convolutional neural networks. In *Proceedings of the European Conference on Computer Vision*, pages 154–169. Springer, 2016. [16](#), [28](#)
- [70] Yang Liu, Jinshan Pan, Jimmy Ren, and Zhixun Su. Learning deep priors for image dehazing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2492–2500, 2019. [16](#)
- [71] Boyi Li, Xiulian Peng, Zhangyang Wang, Jizheng Xu, and Dan Feng. Aod-net: All-in-one dehazing network. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4770–4778, 2017. [16](#), [26](#), [28](#)
- [72] Xu Qin, Zhilin Wang, Yuanchao Bai, Xiaodong Xie, and Huizhu Jia. Ffa-net: Feature fusion attention network for single image dehazing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11908–11915, 2020. [16](#)
- [73] Yan Hu, Keyan Wang, Xi Zhao, Hui Wang, and Yunsong Li. Underwater image restoration based on convolutional neural network. In *Asian Conference on Machine Learning*, pages 296–311, 2018. [17](#)
- [74] Daniel Scharstein and Richard Szeliski. High-accuracy stereo depth maps using structured light. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages I–I. IEEE, 2003. [17](#)
- [75] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2223–2232, 2017. [17](#), [115](#)
- [76] Cameron Fabbri, Md Jahidul Islam, and Junaed Sattar. Enhancing underwater imagery using generative adversarial networks. In *IEEE International Conference on Robotics and Automation*, pages 7159–7165. IEEE, 2018. [17](#)
- [77] Akshay Dudhane, Praful Hambarde, Prashant Patil, and Subrahmanyam Murala. Deep underwater image restoration and beyond. *IEEE Signal Processing Letters*, 27:675–679, 2020. [17](#)
- [78] Younggun Cho, Hyesu Jang, Ramavtar Malav, Gaurav Pandey, and Ayoung Kim. Underwater image dehazing via unpaired image-to-image translation. *International Journal of Control, Automation and Systems*, 18(3):605–614, 2020. [17](#)

- [79] Victor Lempitsky and Andrew Zisserman. Learning to count objects in images. In *Conference on Neural Information Processing Systems*, pages 1324–1332, 2010. [17](#), [81](#)
- [80] Qi Wang, Wei Lin, Junyu Gao, and Xuelong Li. Density-aware curriculum learning for crowd counting. *IEEE Transactions on Cybernetics*, pages 1–13, 2020. doi: 10.1109/TCYB.2020.3033428. [17](#), [18](#), [80](#)
- [81] Vishwanath A Sindagi and Vishal M Patel. Ha-ccn: Hierarchical attention-based crowd counting network. *IEEE Transactions on Image Processing*, 29: 323–335, 2019. [17](#), [79](#), [80](#)
- [82] Guangshuai Gao, Junyu Gao, Qingjie Liu, Qi Wang, and Yunhong Wang. Cnn-based density estimation and crowd counting: A survey. *arXiv preprint arXiv:2003.12783*, 2020. [18](#), [56](#)
- [83] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, pages 91–99, 2015. [18](#), [19](#), [56](#), [77](#), [79](#), [86](#), [104](#), [108](#)
- [84] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *Proceedings of the European Conference on Computer Vision*, pages 21–37. Springer, 2016. [18](#), [93](#), [94](#), [104](#), [108](#)
- [85] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2980–2988, Oct 2017. [18](#), [61](#), [79](#), [81](#)
- [86] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Keypoint triplets for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6569–6578, 2019. [18](#)
- [87] Wei Liu, Shengcai Liao, Weiqiang Ren, Weidong Hu, and Yinan Yu. High-level semantic feature detection: A new perspective for pedestrian detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5187–5196, June 2019. [18](#), [57](#), [59](#), [61](#), [71](#), [77](#)
- [88] Eran Goldman, Roei Herzig, Aviv Eisenschtat, Jacob Goldberger, and Tal Hassner. Precise detection in densely packed scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5227–5236, 2019. [18](#), [79](#), [81](#)
- [89] Issam H Laradji, Negar Rostamzadeh, Pedro O Pinheiro, David Vazquez, and Mark Schmidt. Where are the blobs: Counting by localization with point supervision. In *Proceedings of the European Conference on Computer Vision*, pages 547–562, 2018. [18](#), [19](#), [80](#)

- [90] Haroon Idrees, Muhammad Tayyab, Kishan Athrey, Dong Zhang, Somaya Al-Maadeed, Nasir Rajpoot, and Mubarak Shah. Composition loss for counting, density map estimation and localization in dense crowds. In *Proceedings of the European Conference on Computer Vision*, pages 532–546, 2018. [19](#), [57](#)
- [91] Hui Li, Peng Wang, and Chunhua Shen. Toward end-to-end car license plate detection and recognition with deep neural networks. *IEEE Transactions on Intelligent Transportation Systems*, 20(3):1126–1136, 2018. [19](#), [20](#), [86](#), [89](#), [93](#), [102](#), [104](#), [108](#), [109](#), [110](#)
- [92] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. [19](#)
- [93] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7263–7271, 2017. [19](#), [87](#), [104](#), [108](#)
- [94] Sergio Montazzolli Silva and Claudio Rosito Jung. Real-time brazilian license plate detection and recognition using deep convolutional neural networks. In *SIBGRAPI Conference on Graphics, Patterns and Images*, pages 55–62. IEEE, 2017. [19](#), [87](#), [94](#)
- [95] Gee-Sern Hsu, ArulMurugan Ambikapathi, Sheng-Luen Chung, and Cheng-Po Su. Robust license plate detection in the wild. In *IEEE International Conference on Advanced Video and Signal Based Surveillance*, pages 1–6. IEEE, 2017. [19](#)
- [96] Lele Xie, Tasweer Ahmad, Lianwen Jin, Yuliang Liu, and Sheng Zhang. A new cnn-based method for multi-directional car license plate detection. *IEEE Transactions on Intelligent Transportation Systems*, 19(2):507–517, 2018.
- [97] Rayson Laroca, Evair Severo, Luiz A Zanlorensi, Luiz S Oliveira, Gabriel Resende Gonçalves, William Robson Schwartz, and David Menotti. A robust real-time automatic license plate recognition based on the yolo detector. In *International Joint Conference on Neural Networks*, pages 1–10. IEEE, 2018.
- [98] Sergio Montazzolli Silva and Claudio Rosito Jung. License plate detection and recognition in unconstrained scenarios. In *Proceedings of the European Conference on Computer Vision*, pages 580–596, 2018. [19](#), [86](#), [87](#)
- [99] Hongwei Qin, Junjie Yan, Xiu Li, and Xiaolin Hu. Joint training of cascaded cnn for face detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3456–3465, 2016. [19](#)
- [100] Wanwei Wang, Jun Yang, Min Chen, and Peng Wang. A light cnn for end-to-end car license plates detection and recognition. *IEEE Access*, 7: 173875–173883, 2019. [19](#), [20](#), [87](#), [88](#), [89](#), [104](#), [106](#), [107](#), [108](#)

- [101] Gee-Sern Hsu, Jiun-Chang Chen, and Yu-Zu Chung. Application-oriented license plate recognition. *IEEE Transactions on Vehicular Technology*, 62(2): 552–561, 2012. [20](#), [23](#), [88](#), [96](#), [97](#), [102](#), [107](#)
- [102] Chao Gou, Kunfeng Wang, Yanjie Yao, and Zhengxi Li. Vehicle license plate recognition based on extremal regions and restricted boltzmann machines. *IEEE Transactions on Intelligent Transportation Systems*, 17(4):1096–1107, 2015. [20](#)
- [103] Jiafan Zhuang, Saihui Hou, Zilei Wang, and Zheng-Jun Zha. Towards human-level license plate recognition. In *Proceedings of the European Conference on Computer Vision*, pages 306–321, 2018. [20](#), [87](#), [94](#)
- [104] Alex Graves, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. A novel connectionist system for unconstrained handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(5):855–868, 2008. [20](#)
- [105] Cosmin Ancuti, Codruta Orniana Ancuti, Tom Haber, and Philippe Bekaert. Enhancing underwater images and videos by fusion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 81–88, 2012. [21](#), [44](#)
- [106] Simon Emberton, Lars Chittka, and Andrea Cavallaro. Hierarchical rank-based veiling light estimation for underwater dehazing. In *British Machine Vision Conference*, pages 125.1—125.12, 2015. [21](#), [44](#)
- [107] Yi Wang, Hui Liu, and Lap-Pui Chau. Single underwater image restoration using adaptive attenuation-curve prior. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 65(3):992–1002, 2018. [26](#)
- [108] Yanting Pei, Yaping Huang, Qi Zou, Yuhang Lu, and Song Wang. Does haze removal help cnn-based image classification? In *Proceedings of the European Conference on Computer Vision*, pages 682–697. Springer, 2018. [26](#), [115](#)
- [109] John P Oakley and Brenda L Satherley. Improving image quality in poor visibility conditions using a physical model for contrast degradation. *IEEE Transactions on Image Processing*, 7(2):167–179, 1998. [26](#)
- [110] Wencheng Wang and Xiaohui Yuan. Recent advances in image dehazing. *IEEE/CAA Journal of Automatica Sinica*, 4(3):410–436, 2017. [28](#)
- [111] Richard W Gould, Robert A Arnone, and Paul M Martinolich. Spectral dependence of the scattering coefficient in case 1 and case 2 waters. *Applied Optics*, 38(12):2377–2383, 1999. [30](#)
- [112] Xinwei Zhao, Tao Jin, and Song Qu. Deriving inherent optical properties from background color and underwater image enhancement. *Ocean Engineering*, 94:163–172, 2015. [31](#)

- [113] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016. [42](#)
- [114] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, Sabine Süsstrunk, et al. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, 2012. [42](#)
- [115] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3485–3492, 2010. [44](#)
- [116] Karen Panetta, Chen Gao, and Sos Agaian. Human-visual-system-inspired underwater image quality measures. *IEEE Journal of Oceanic Engineering*, 41(3):541–551, 2016. [46](#)
- [117] Teng Li, Huan Chang, Meng Wang, Bingbing Ni, Richang Hong, and Shuicheng Yan. Crowded scene analysis: A survey. *IEEE Transactions on Circuits and Systems for Video Technology*, 25(3):367–386, 2015. [56](#)
- [118] Di Kang, Zheng Ma, and Antoni B Chan. Beyond counting: comparisons of density maps for crowd analysis tasks-counting, detection, and tracking. *IEEE Transactions on Circuits and Systems for Video Technology*, 29(5):1408–1422, 2019. [56](#)
- [119] Antoni B Chan, Zhang-Sheng John Liang, and Nuno Vasconcelos. Privacy preserving crowd monitoring: Counting people without people models or tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–7. IEEE, 2008. [56](#), [58](#), [81](#)
- [120] Yen-Lin Chen, Bing-Fei Wu, Hao-Yu Huang, and Chung-Jui Fan. A real-time vision system for nighttime vehicle detection and traffic surveillance. *IEEE Transactions on Industrial Electronics*, 58(5):2030–2044, 2011. [56](#)
- [121] Mikel Rodriguez, Ivan Laptev, Josef Sivic, and Jean-Yves Audibert. Density-aware person detection and tracking in crowds. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2423–2430. IEEE, 2011. [56](#)
- [122] Matthew A Turk and Alex P Pentland. Face recognition using eigenfaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 586–591. IEEE, 1991. [56](#)
- [123] Wei Li, Rui Zhao, Tong Xiao, and Xiaogang Wang. Deepreid: Deep filter pairing neural network for person re-identification. In *Proceedings of the IEEE*

- Conference on Computer Vision and Pattern Recognition*, pages 152–159, 2014. [56](#)
- [124] Daniel Oñoro-Rubio and Roberto J López-Sastre. Towards perspective-free object counting with deep learning. In *Proceedings of the European Conference on Computer Vision*, pages 615–629. Springer, 2016. [58](#), [81](#), [82](#)
- [125] Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 761–769, 2016. [66](#)
- [126] Aruni RoyChowdhury, Prithvijit Chakrabarty, Ashish Singh, SouYoung Jin, Huaizu Jiang, Liangliang Cao, and Erik Learned-Miller. Automatic adaptation of object detectors to new domains using self-training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 780–790, 2019. [66](#)
- [127] Vishwanath A Sindagi and Vishal M Patel. Cnn-based cascaded multi-task learning of high-level prior and density estimation for crowd counting. In *IEEE International Conference on Advanced Video and Signal Based Surveillance*, pages 1–6. IEEE, 2017. [67](#), [68](#)
- [128] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. [71](#)
- [129] Shuo Yang, Ping Luo, Chen-Change Loy, and Xiaoou Tang. From facial parts responses to face detection: A deep learning approach. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3676–3684, 2015. [77](#)
- [130] Peiyun Hu and Deva Ramanan. Finding tiny faces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 951–959, 2017. [77](#), [79](#), [80](#)
- [131] Junyu Gao, Tao Han, Qi Wang, and Yuan Yuan. Domain-adaptive crowd counting via inter-domain features segregation and gaussian-prior reconstruction. *arXiv preprint arXiv:1912.03677*, 2019. [79](#)
- [132] Chenchen Liu, Xinyu Weng, and Yadong Mu. Recurrent attentive zooming for joint crowd counting and precise localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1217–1226, 2019. doi: 10.1109/CVPR.2019.00131. [79](#)
- [133] Tobias Stahl, Silvia L Pintea, and Jan C van Gemert. Divide and count: Generic object counting by image divisions. *IEEE Transactions on Image Processing*, 28(2):1035–1044, 2018. [79](#), [81](#)

- [134] Wei Li, Hongliang Li, Qingbo Wu, Xiaoyu Chen, and King Ng Ngan. Simultaneously detecting and counting dense vehicles from drone images. *IEEE Transactions on Industrial Electronics*, 66(12):9651–9662, 2019. [79](#), [81](#)
- [135] Ke Chen, Chen Change Loy, Shaogang Gong, and Tony Xiang. Feature mining for localised crowd counting. In *British Machine Vision Conference*, volume 1, page 3, 2012. [81](#)
- [136] Ke Chen, Shaogang Gong, Tao Xiang, and Chen Change Loy. Cumulative attribute space for age and crowd density estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2467–2474. IEEE, June 2013. [81](#)
- [137] Cong Zhang, Hongsheng Li, Xiaogang Wang, and Xiaokang Yang. Cross-scene crowd counting via deep convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 833–841. IEEE, 2015. [81](#)
- [138] Shanghang Zhang, Guanhang Wu, Joao P Costeira, and José MF Moura. Understanding traffic density from large-scale web camera data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4264–4273. IEEE, 2017. [81](#), [82](#)
- [139] Christos Nikolaos E Anagnostopoulos, Ioannis E Anagnostopoulos, Vassilis Loumos, and Eleftherios Kayafas. A license plate-recognition algorithm for intelligent transportation system applications. *IEEE Transactions on Intelligent Transportation Systems*, 7(3):377–392, 2006. [86](#)
- [140] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7132–7141, 2018. [90](#), [91](#), [92](#), [106](#)
- [141] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2117–2125, 2017. [92](#), [93](#)
- [142] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Fractalnet: Ultra-deep neural networks without residuals. In *International Conference on Learning Representations*, 2017. [98](#)
- [143] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *Proceedings of the European Conference on Computer Vision*, pages 646–661. Springer, 2016. [98](#), [99](#)
- [144] Xavier Gastaldi. Shake-shake regularization. *arXiv preprint arXiv:1705.07485*, 2017. [98](#)

- [145] Yoshihiro Yamada, Masakazu Iwamura, Takuya Akiba, and Koichi Kise. Shakedrop regularization for deep residual learning. *IEEE Access*, 7:186126–186136, 2019. ISSN 2169-3536. doi: 10.1109/ACCESS.2019.2960566. [98](#)
- [146] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the Annual International Conference on Machine Learning*, pages 41–48. ACM, 2009. [98](#), [116](#)
- [147] Pietro Morerio, Jacopo Cavazza, Riccardo Volpi, René Vidal, and Vittorio Murino. Curriculum dropout. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3544–3552, 2017. [98](#)
- [148] M Pawan Kumar, Benjamin Packer, and Daphne Koller. Self-paced learning for latent variable models. In *Advances in Neural Information Processing Systems*, pages 1189–1197, 2010. [99](#)
- [149] Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V Le. Dropblock: A regularization method for convolutional networks. In *Advances in Neural Information Processing Systems*, pages 10727–10737, 2018. [99](#), [101](#)
- [150] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8697–8710, 2018. [99](#), [101](#)
- [151] Yi Wang, Zhen-Peng Bian, Junhui Hou, and Lap-Pui Chau. Convolutional neural networks with dynamic regularization. *IEEE Transactions on Neural Networks and Learning Systems*, 2020. [102](#)
- [152] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 113–123, 2019. [103](#)
- [153] Tianwei Wang, Yuanzhi Zhu, Lianwen Jin, Canjie Luo, Xiaoxue Chen, Yaqiang Wu, Qianying Wang, and Mingxiang Cai. Decoupled attention network for text recognition. In *AAAI*, 2020. [108](#)
- [154] Kui Jiang, Zhongyuan Wang, Peng Yi, Chen Chen, Baojin Huang, Yimin Luo, Jiayi Ma, and Junjun Jiang. Multi-scale progressive fusion network for single image deraining. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8346–8355, 2020. [115](#)
- [155] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Proceedings of the European Conference on Computer Vision*, pages 740–755. Springer, 2014. [115](#)

- [156] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014. [115](#)
- [157] Yuhua Chen, Wen Li, Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Domain adaptive faster r-cnn for object detection in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3339–3348, 2018. [115](#)
- [158] Burr Settles. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2009. [116](#)
- [159] Weiwei Shi, Yihong Gong, Chris Ding, Zhiheng MaXiaoyu Tao, and Nanning Zheng. Transductive semi-supervised deep learning using min-max features. In *Proceedings of the European Conference on Computer Vision*, pages 299–315, 2018. [116](#)
- [160] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10687–10698, 2020. [116](#)
- [161] Simon Vandenhende, Stamatios Georgoulis, Wouter Van Gansbeke, Marc Proesmans, Dengxin Dai, and Luc Van Gool. Multi-task learning for dense prediction tasks: A survey. *arXiv preprint arXiv:2004.13379*, 2020. [117](#)
- [162] Simon Vandenhende, Stamatios Georgoulis, and Luc Van Gool. Mti-net: Multi-scale task interaction networks for multi-task learning. *Proceedings of the European Conference on Computer Vision*, 2020. [117](#)
- [163] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Efficient parametrization of multi-domain deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8119–8127, 2018. [117](#)