THE UNIVERSITY of ADELAIDE

Efficient Fully Convolutional Networks for Dense Prediction Tasks

Yifan Liu

A thesis submitted for the degree of
DOCTOR OF PHILOSOPHY
The University of Adelaide

October 14, 2021

Contents

List of Figures

List of Tables

We develop efficient fully convolutional networks for dense prediction tasks, which learns a mapping from input images to complex output structures. To get a better trade-off between performance and the efficiency, we mainly focus on better training off-the-shelf efficient convolutional networks. Extra training constraints from larger models, temporal information, auxiliary model and unlabeled data are discussed and employed in this thesis. The performance of the efficient fully convolutional network can be boosted without introduce any extra computational cost during the. inference process. The effectiveness has been verified on various tasks, including semantic segmentation, depth estimation and object detection.

University of Adelaide

Abstract

Efficient Fully Convolutional Networks for Dense Prediction Tasks

by Yifan Liu

Dense prediction is a family of fundamental problems in computer vision, which learns a mapping from input images to complex output structures, including semantic segmentation, depth estimation, and object detection, among many others. Pixel-level labeling is required in such tasks. Deep neural networks have been the dominant solution since the invention of fully-convolutional neural networks (FCNs). Well-designed complicated network structures achieve state-of-the-art performance on benchmark datasets, but often with a high computational cost. The cost will be more expensive when extending to the video sequence. It is important to design efficient fully convolutional networks for dense prediction tasks so that the models can be used on mobile devices in many real-world applications. Light-weight models have drawn much attention recently. Most compact models try to obtain higher accuracy with lower computational cost, but usually, they need to make the trade-off between accuracy and efficiency. Besides, it is hard to train a compact model properly with limited model capacity. Thus, we target improving the performance of fully convolutional networks by using extra constraints during the training process to keep the efficiency of the inference. Our study starts with knowledge distillation, which has been verified valid in classification tasks. The compact models are trained with the help of large models. We design several new distillation methods for capturing the structure information, taking into account the fact that dense prediction is a structured prediction problem. Moreover, we extend the distillation methods to the video sequence and design temporal knowledge distillation. Both the temporal consistency and the accuracy of the compact models can be improved. Except for knowledge distillation, we employ auxiliary modules to provide extra gradients or supervisions in training compact models. Through our training methods, we can improve the performance of compact models without any extra computational costs during inference. The proposed training methods are general and can be applied to various network structures, datasets, and tasks. We mainly conduct our experiments on typical dense prediction tasks, e.g., semantic segmentation with both images and video sequences. We also extend our methods to object detection, depth estimation, and the multi-task learning system. We outperform previous works with a better trade-off between accuracy and efficiency for various dense prediction tasks.

Declaration of Authorship

I certify that this work contains no material which has been accepted for the award of any other degree or diploma in my name, in any university or other tertiary institution and, to the best of my knowledge and belief, contains no material previously published or written by another person, except where due reference has been made in the text. In addition, I certify that no part of this work will, in the future, be used in a submission in my name, for any other degree or diploma in any university or other tertiary institution without the prior approval of the University of Adelaide and where applicable, any partner institution responsible for the joint-award of this degree. I acknowledge that copyright of published works contained within this thesis resides with the copyright holder(s) of those works. I also give permission for the digital version of my thesis to be made available on the web, via the University's digital research repository, the Library Search and also through web search engines, unless permission has been granted by the University to restrict access for a period of time. I acknowledge the support I have received for my research through the provision of an Australian Government Research Training Program Scholarship.

Yifan Liu

May 2021

Statement of Authorship

Title of Paper	Auxiliary Learning for Deep Multi-task Learning
Publication Status	☐ Published ☐ Accepted for Publication ☐ Submitted for Publication ☑ Unpublished and Unsubmitted w ork w ritten in manuscript style
Publication Details	This paper is an arxiv preprint.

Principal Author

Name of Principal Author (Candidate)	Yifan Liu		
Contribution to the Paper	Conducted experiments and wrote the paper.		
Overall percentage (%)	70		
Certification:	This paper reports on original research I conducted during the period of my Higher Degree by Research candidature and is not subject to any obligations or contractual agreements with a third party that would constrain its inclusion in this thesis. I am the primary author of this paper.		
Signature		Date	10/05/2021

Co-Author Contributions

By signing the Statement of Authorship, each author certifies that:

i. the candidate's stated contribution to the publication is accurate (as detailed above);

ii. permission is granted for the candidate in include the publication in the thesis; and

iii. the sum of all co-author contributions is equal to 100% less the candidate's stated contribution.

Name of Co-Author	Bohan Zhuang		
Contribution to the Paper	Discussion and writing the revision.		
Signature		Date	10/05/2021

Name of Co-Author	Chunhua Shen		
Contribution to the Paper	Discussion and writing the revision.		
Signature		Date	10/05/2021

Please cut and paste additional co-author panels here as required.

Name of Co-Author	Hao Chen		
Contribution to the Paper	Discussion and writing the revision.		
Signature		Date	10/05/2021

Name of Co-Author	Wei Yin		
Contribution to the Paper	Discussion and writing the revision.		
Signature		Date	10/05/2021

Please cut and paste additional co-author panels here as required.

Statement of Authorship

Title of Paper	Channel Distribution Distillation for Dense Prediction
Publication Status	☐ Published ☐ Accepted for Publication ☑ Submitted for Publication ☐ Unpublished and Unsubmitted w ork w ritten in manuscript style
Publication Details	This paper is submited to ICCV2021.

Principal Author

Name of Principal Author (Candidate)	Yifan Liu		
Contribution to the Paper	Proposed the ideas, conducted experiments and wrote the paper.		
Overall percentage (%)	70		
Certification:	This paper reports on original research I conducted during the period of my Higher Degree by Research candidature and is not subject to any obligations or contractual agreements with a third party that would constrain its inclusion in this thesis. I am the primary author of this paper.		
Signature		Date	10/05/2021

Co-Author Contributions

By signing the Statement of Authorship, each author certifies that:

i. the candidate's stated contribution to the publication is accurate (as detailed above);

ii. permission is granted for the candidate in include the publication in the thesis; and

iii. the sum of all co-author contributions is equal to 100% less the candidate's stated contribution.

Name of Co-Author	Changyong Shu		
Contribution to the Paper	Conducted part of the experiments and wrote the paper.		
Signature		Date	10/05/2021

Name of Co-Author	Jianfei Gao		
Contribution to the Paper	Conducted part of the experiments.		
Signature		Date	10/05/2021

Please cut and paste additional co-author panels here as required.

Name of Co-Author	Zhen Yan		
Contribution to the Paper	Discussion and writing the revision.		
Signature		Date	10/05/2021

Name of Co-Author	Chunhua Shen		
Contribution to the Paper	Discussion and writing the revision.		
Signature		Date	10/05/2021

Please cut and paste additional co-author panels here as required.

Statement of Authorship

Title of Paper	Efficient Semantic Video Segmentation with Per-frame Inference
Publication Status	☑ Published ☐ Accepted for Publication ☐ Submitted for Publication ☐ Unpublished and Unsubmitted w ork w ritten in manuscript style
Publication Details	Published in ECCV 2020

Principal Author

Name of Principal Author (Candidate)	Yifan Liu
Contribution to the Paper	Proposed the ideas, conducted the experiments and wrote the paper.
Overall percentage (%)	70
Certification:	This paper reports on original research I conducted during the period of my Higher Degree by Research candidature and is not subject to any obligations or contractual agreements with a third party that would constrain its inclusion in this thesis. I am the primary author of this paper.
Signature	Date: 10/05/2021

Co-Author Contributions

By signing the Statement of Authorship, each author certifies that:

i. the candidate's stated contribution to the publication is accurate (as detailed above);

ii. permission is granted for the candidate in include the publication in the thesis; and

iii. the sum of all co-author contributions is equal to 100% less the candidate's stated contribution.

Name of Co-Author	Changqian Yu
Contribution to the Paper	Discussion and writing the revision.
Signature	Date: 10/05/2021

Name of Co-Author	Jingdong Wang
Contribution to the Paper	Discussion and writing the revision.
Signature	Date: 10/05/2021

Name of Co-Author	Chunhua Shen
Contribution to the Paper	Discussion and writing the revision.
Signature	Date: 10/05/2021

Statement of Authorship

Title of Paper	Structured Knowledge Distillation for Semantic Segmentation
Publication Status	☑ Published ☐ Accepted for Publication ☐ Submitted for Publication ☐ Unpublished and Unsubmitted w ork w ritten in manuscript style
Publication Details	Published in CVPR 2019

Principal Author

Name of Principal Author (Candidate)	Yifan Liu
Contribution to the Paper	Proposed the ideas, conducted experiments and wrote the paper.
Overall percentage (%)	70
Certification:	This paper reports on original research I conducted during the period of my Higher Degree by Research candidature and is not subject to any obligations or contractual agreements with a third party that would constrain its inclusion in this thesis. I am the primary author of this paper.
Signature	Date: 10/05/2021

Co-Author Contributions

By signing the Statement of Authorship, each author certifies that:

i. the candidate's stated contribution to the publication is accurate (as detailed above);

ii. permission is granted for the candidate in include the publication in the thesis; and

iii. the sum of all co-author contributions is equal to 100% less the candidate's stated contribution.

Name of Co-Author	Ke Chen
Contribution to the Paper	Discussion and writing the revision.
Signature	Date: 10/05/2021

Name of Co-Author	Chris Liu
Contribution to the Paper	Discussion and writing the revision.
Signature	Date: 10/05/2021

Please cut and paste additional co-author panels here as required.

Name of Co-Author	Zengchang Qin		
Contribution to the Paper	Discussion and writing the revision.		
Signature		Date	10/05/2021

Name of Co-Author	Zhenbo Luo		
Contribution to the Paper	Discussion and writing the revision.		
Signature		Date	10/05/2021

Name of Co-Author	Jingdong Wang		
Contribution to the Paper	Discussion and writing the revision.		
Signature		Date	10/05/2021

Statement of Authorship

Title of Paper	Structured Knowledge Distillation for Dense Prediction
Publication Status	☑ Published ☐ Accepted for Publication ☐ Submitted for Publication ☐ Unpublished and Unsubmitted w ork w ritten in manuscript style
Publication Details	Published in TPAMI 2020

Principal Author

Name of Principal Author (Candidate)	Yifan Liu
Contribution to the Paper	Proposed the ideas, conducted the main experiments and wrote the paper.
Overall percentage (%)	70
Certification:	This paper reports on original research I conducted during the period of my Higher Degree by Research candidature and is not subject to any obligations or contractual agreements with a t constrain its inclusion in this thesis. I am the primary author of this paper.
Signature	Date: 10/05/2021

Co-Author Contributions

By signing the Statement of Authorship, each author certifies that:

i. the candidate's stated contribution to the publication is accurate (as detailed above);

ii. permission is granted for the candidate in include the publication in the thesis; and

iii. the sum of all co-author contributions is equal to 100% less the candidate's stated contribution.

Name of Co-Author	Changyong Shu
Contribution to the Paper	Conducted part of the experiments.
Signature	Date: 10/05/2021

Name of Co-Author	Jingdong Wang
Contribution to the Paper	Discussion and writing the revision.
Signature	Date: 10/05/2021

Name of Co-Author	Chunhua Shen
Contribution to the Paper	Discussion and writing the revision.
Signature	Date: 10/05/2021

Acknowledgements

The life of studying at the University of Adelaide is full of unforgettable memories. I had a wonderful experience, enriched my knowledge, expanded my research field, found a career plan, and made many good friends. In the process of research, I have encountered many difficulties and challenges, and have also received help from many people.

First of all, I want to thank my Ph.D. supervisor, Chunhua Shen, who gave me a chance to enter the field of computer vision. He is knowledgeable, creative, and full of passion. He taught us solid basic skills in research. When we are stuck in research difficulties, he usually shares some insights based on his awesome research taste. He is also a kind, warm and helpful friend. He often encourages us to think about our future and make our career plan. Without his help, I can not get succeed in my Ph.D. career.

Besides, I also want to thank Prof. Jingdong Wang and Prof. Wanli Ouyang. They have helped me a lot during my Ph.D. career with my research.

I am also grateful to my co-authors. We collaborate to have great ideas and work as a team. They are Wei Yin, Hao Chen, Zhi Tian, Changqian Yu, Yang Zhao, Yuanzhouhan Cao Tong He, Bohan Zhuang, Xinlong Wang, and Yutong Dai, Jianlong Yuan, and Changyong Shu. It is a great pleasure to work with you. I also want to thank Google, LLC for their generous financial support with the Google Ph.D. fellowship.

I want to thank other colleagues in my lab and friends outside my lab. We usually discuss with each other during the workdays and sometimes have road trips together. They are Hui Li, Hu Wang, Qichang Hu, Libo Sun, Jiawang Bian, Xinyu Zhang, Weian Mao, Bowen Zhang, Yue Xie, Deb, Jon, Andy, and Ifan.

Thank our wonderful basketball team of AIML! The players include CK, Daqi Liu, Fengyi, Elaine, Ming Cai, Huangying, Binghong Liang Liu, Hao Lu, Yuliang Liu, and Fangchao Tian. Playing basketball can exercise our body and relieve stress.

Finally, thank my family for giving me support during my study.

Publications

This thesis contains the following works that have been published or prepared for publication:

- Structured Knowledge Distillation for Semantic Segmentation.
  Yifan Liu, Ke Chen, Chris Liu, Zengchang Qin, Zhenbo Luo, Jingdong Wang.
  Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

- Structured Knowledge Distillation for Dense Prediction.
  Yifan Liu, Changyong Shu, Jingdong Wang, Chunhua Shen.
  IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2020.

- Channel Distribution Distillation for Dense Prediction.
  Yifan Liu*, Changyong Shu*, Jianfei Gao, Zhen Yan, Chunhua Shen.
  International Conference on Computer Vision (ICCV), 2021.

- Efficient Semantic Video Segmentation with Per-frame Inference.
  Yifan Liu, Chunhua Shen, Changqian Yu, and Jingdong Wang.
  European Conference on Computer Vision (ECCV), 2020

- Auxiliary Learning for Deep Multi-task Learning.
  Yifan Liu, Bohan Zhuang, Chunhua Shen, Hao Chen, Wei Yin
  ArXiv Preprint, 2020.

In addition, I have the following papers not included in this thesis:

- Generic Perceptual Loss for Modeling Structured Output Dependencies.
  Yifan Liu, Hao Chen, Yu Chen, Wei Yin, Chunhua Shen
  Conference on Computer Vision and Pattern Recognition (CVPR), 2021.

- Enforcing geometric constraints of virtual normal for depth prediction.
  Wei Yin, **Yifan Liu**, Chunhua Shen, Youliang Yan
  International Conference on Computer Vision (ICCV), 2019.

- MobileFAN: transferring deep hidden representation for face alignment.
  Yang Zhao, **Yifan Liu**, Chunhua Shen, Yongsheng Gao, Shengwu Xiong.
  Pattern Recognition (PR) 2020.

- Representative Graph Neural Network.
  Changqian Yu, **Yifan Liu**, Changxin Gao, Chunhua Shen, Nong Sang
  European Conference on Computer Vision (ECCV), 2020

- Instance-Aware Embedding for Point Cloud Instance Segmentation.
  Tong He, **Yifan Liu**, Chunhua Shen, Xinlong Wang, Changming Sun
  European Conference on Computer Vision (ECCV), 2020

Chapter 1

Introduction

Scene understanding is a core topic for computer vision. Developing algorithms to analyze the location, layout, and geometry of a specific scene derives several fundamental vision tasks, such as semantic segmentation [90], depth estimation [143], object detection [131] and so on. These tasks require pixel-level labeling for an input image and belong to the family of dense prediction tasks. Dense prediction tasks have a lot of applications such as self-driving, robotics, photography, and augmented reality. It is crucial to consider the accuracy as well as the efficiency when designing algorithms.

Due to the high representation and generalization ability, deep learning methods have been the dominant approach since the invention of fully-convolutional neural networks (FCNs) [90]. New state-of-the-art performance has been achieved on public benchmarks continuously. To obtain higher performance, two factors are usually considered. The first one is to involve massive labeled data. Taking semantic segmentation as an example, to compare with state-of-the-art performance on a popular self-driving benchmark, i.e., Cityscapes [25], recent works [126, 18, 159] usually pretrain their models on a larger self-driving dataset, *e.g.*, Apollo [55]. Thus, superior performance can be achieved. However, pixel-level labels are hard to obtain and will require huge human efforts. The labeling cost will be more expensive if consistent labels are required across a video sequence. The second direction to improve the performance is to design high-computational task-specific modules. A complicated model with a powerful backbone and a well-designed task-specific module is helpful to achieve higher performance, but it is hard to apply to mobile devices. The computational cost will be even higher if we apply the complicated networks to the video sequence. For example, the high-performance OCRNet [159] can only run 4 fps with input images with the shape of 1024×2048 on the public benchmark [25], which can not meet the requirement of real-time applications.

Recently, efficient fully convolutional networks have drawn much attention. Some works [117, 170] employ lightweight backbones from the image classification tasks for the feature extraction of dense prediction tasks. Other works [116, 2] design lightweight structures by accelerating the convolution operations, like depth-wise or channel-wise convolutions. There are also some works [154, 173] pay attention to reducing the input resolution to get a higher inference speed, but details will be lost with reduced input images. Compact models can significantly reduce the parameters

and the computational costs for the deep models. However, due to the limited model capacity, training the compact models is harder than the larger ones. Besides, directly applying the compact models to each frame in a video sequence will lead to inconsistent results as the model is hard to capture the temporal information among frames.

1.1 Motivation

As has been stated, massive labeled data are hard to obtain, large models have low efficiency, and compact models are hard to train properly to get a promising performance. In this thesis, we focus on achieving a better trade-off between performance and efficiency on images and videos for different dense prediction tasks. The core idea is to help the training of efficient convolutional networks with extra constraints and unlabeled data. Although being challenging, we find properly training compact models for dense prediction tasks is important for both empirical and theoretical reasons. We consider adding extra constraints from various sources, including larger models (Chapter 3 and 4), temporal information (Chapter 5), and auxiliary models (Chapter 6). The proposed methods in Chapter 3, 4 and 5 can also be applied to unlabeled data.

Previous works [49, 164, 3] have explored training a compact model with the help of large models in the classification task, also known as knowledge distillation. The pioneering work employs a KL divergence to minimize the difference between the soft logits from the teacher nets (large modes) and the student nets (compact models). The correlation among different classes can help the student networks to learn better. Following works also pay attention to the inner feature maps [3, 165], and try to align pixel-level feature maps between the teacher and student nets. Most dense prediction tasks can be seen as pixel-level classification problems with different targets. The knowledge distillation methods for classification tasks can be directly applied to each pixel in dense prediction tasks. However, pixels in dense prediction tasks are not independent of each other, and they usually formulate structure outputs. It will be helpful if the structure information can be considered during designing the knowledge distillation frameworks for dense prediction tasks.

We first propose the structural knowledge distillation frameworks for dense prediction tasks, including semantic segmentation, depth estimation, and object detection. The pair-wise distillation and holistic distillation are proposed to capture the correlation among pixels explicitly and implicitly. Based on this work, we further find out that the activation values in each channel also contain structure information. Besides, making use of the channel distribution is more efficient than calculating the pair-wise correlation. By aligning the salient region in each channel, the student nets can focus more on meaningful regions in spatial space.

When applying the compact models to video sequences, new challenges occur. Except for the low per-frame accuracy, the compact models are hard to produce temporal consistent results when the labeling images are sparse across the frames [63]. To deal

with this, training constraints from the pre-trained motion network are considered in this thesis. The prediction logits between adjacent frames are forced to be the same on the matching pixels. Furthermore, new temporal consistent knowledge distillation methods are proposed based on the single frame structural knowledge distillation.

The large teacher model and the temporal information may be hard to obtain under some circumstances, e.g, in a compact multi-task learning system. Dealing with multiple tasks in one network is hard to train as different features are needed in one compact backbone. An easy and useful auxiliary module is designed for multi-task learning. Benefit from the auxiliary module, the performance for a specific task or all tasks can be boosted.

All the proposed methods are training schemes with extra constraints. We can improve the performance of an off-the-shelf efficient fully convolutional network without any extra computational costs during the inference process. Although more training memory and longer training time are required, the empirical and theoretical values of the proposed methods are still non-negligible.

1.2 Contribution

We train efficient fully convolutional networks for various dense perception tasks with extra constraints and unlabeled data. This thesis aims to further improve the performance (*e.g.* accuracy and stability) of a well-designed efficient fully convolutional network without introducing any extra computational cost during the inference process. The contributions of this thesis are:

- Inspired by that training with extra constraints can help the compact model converge better, we design several training constraints in this thesis. The training constraints can come from large networks (i.e. knowledge distillation), temporal correlation, and overparameterization with auxiliary modules.
- The proposed training methods can also be applied to the unlabeled data in a semi-supervised way.
- We propose several knowledge distillation methods for 2D images, including the structural knowledge distillation, and the channel-wise distillation for training compact models for semantic segmentation. The state-of-the-art performance on various datasets has been achieved.
- The distillation methods can be extended to other tasks, including object detection, and depth estimation. The performance of various strong baselines has been boosted.
- The distillation methods can be extended to video sequences. To obtain an accurate, efficient, and stable model, we devise new temporal consistent knowledge distillation methods. The temporal consistency and the accuracy can be improved without any extra computational cost during the inference process.

- To make use of unlabeled video frames and temporal correlations, a motion loss is proposed. The temporal consistency can be improved by aligning the predictions from two adjacent frames with the help of a pre-trained flow estimation network.
- An auxiliary module is designed to help the training of a multi-task learning system. Extra constraints come from the supervision from the auxiliary branch.

Chapter 2 gives the details about the related literature. We introduce two types of compact model learning systems. Then, the development of dense prediction tasks, including image/video semantic segmentation, depth estimation, object detection, is covered. Finally, we summarize several previous training methods as adding auxiliary constraints. The relation and difference are discussed.

Chapter 3 describes the basic structural knowledge distillation framework for semantic segmentation. The pair-wise distillation and holistic distillation are proposed to capture structure information between the teacher and student networks. The training methods are further applied to depth estimation and object detection tasks.

Chapter 4 analyzes the shortages of previous structural knowledge distillation, i.e. the large training memory caused by the pair-wise distillation. To solve this, a new distillation framework regarding the channel-relation is proposed, namely channel distillation. Superior performance is achieved and less training memory is required.

Chapter 5 focuses on building an efficient fully convolutional network for semantic video segmentation. Accuracy, temporal consistency, and efficiency are considered during designing the model. Two training methods are proposed, including a motion loss and the temporal consistent knowledge distillation.

Chapter 6 aims to develop efficient fully convolutional networks for the multi-task learning system through hard parameter sharing. The proposed auxiliary model can provide extra gradients for specific tasks. Thus, the overall performance of the multi-task learning system can be boosted.

In **Chapter 7**, we summarize the methods and the contribution of this thesis and discuss the future work in developing efficient fully convolutional networks for dense prediction tasks.

Chapter 2

Literature Review

2.1 Efficient Fully Convolutional Networks

First of all, we introduce two categories of problems we term efficient network learning (ENL). The first one is the lightweight network structure designed for a single specific task, which forms an efficient structure. The efficient classification networks, *e.g.*, MobileNet [52, 117], EfficientNet [130], ShuffleNet [170], and IGCNet [169] are the fundamental works in this subarea, which design basic blocks with depth-wise convolutions or group convolutions to reduce the computation cost and are widely used to accelerate other dense prediction problems in computer vision, such as semantic segmentation, object detection, depth prediction and so on. Besides, the subsequent approaches, Enet [106], ESPNet [92], YOLO [111, 109] focus on lightweight network designed by accelerating the convolution operations with factorization techniques. Moreover, Nekrasov et al. [98] employed the neural architecture search to sample the decoder for depth prediction problems and achieve promising results. Most of these methods are trying to find a better trade-off between accuracy and efficiency. However, lightweight structures may lead to optimization difficulties. Another type of problem we define as ENL is the hard parameter sharing multi-task learning. As shown in Figure 5.2-(a) and (c), there are two typical ways of building an MTL system, including hard parameters sharing [9, 96, 62] and soft parameters sharing [102, 150]. The former one builds an efficient way to handle the MTL by sharing the backbone encoder and design task-specific decoders for each task. The latter one utilizes an individual network for each task with only passing the variables among tasks, and the complexity will grow linearly with the number of tasks.

The hard parameters sharing system shares the backbone network for different tasks, which can be treated as an efficient network. It is challenging to be optimized properly because of the competing task objectives [121]. And some approaches [148, 95, 172] focus on designing complex task-specific decoders to improve the performance.

As we mainly focus on dense prediction tasks, including semantic segmentation, object detection, and depth estimation, our efficient networks are built with convolutional layers and can be feed with arbitrary input sizes. Different from previous methods for ENL, we do not focus on designing an efficient structure, but designing training modules, which can be removed during inference. Therefore we can improve

the performance of the compact model without increasing any complexity during inference. We propose to assist the training of the first kind of efficient networks through newly-designed knowledge distillation methods in Chapter 3 and 4. Besides, we propose training efficient networks for video sequence through knowledge distillation and auxiliary losses in Chapter 5. Finally, we propose an auxiliary module to help the training of the multi-task models in Chapter 6.

2.2 Dense Prediction

2.2.1 Semantic Image/video Segmentation

Semantic segmentation is a pixel classification problem, which requires an semantic understanding of the whole scene. Deep convolutional neural networks have been the dominant solution to semantic segmentation since the pioneering work, fully-convolutional networks [122]. Various schemes have been developed for improving the network capability and accordingly the segmentation performance. For example, stronger backbone networks such as ResNets [42] and DenseNets [54], have shown improved segmentation performance. Retaining the spatial resolution through dilated convolutions [17] or multi-path refine networks [74] leads to significant performance gain. Exploiting multi-scale context using dilated convolutions [156], or pyramid pooling modules in PSPNet [176], also benefits the segmentation. Lin et al. [73] combine deep models with structured output learning for semantic segmentation.

Recently, highly efficient segmentation networks have been attracting increasingly more interests due to the need for mobile applications. Most works focus on lightweight network design by accelerating the convolution operations with techniques such as factorization techniques. ENet [106], inspired by [129], integrates several acceleration factors, including multi-branch modules, early feature map resolution downsampling, small decoder size, filter tensor factorization, and so on. SQ [134] adopts the SqueezeNet [56] fire modules and parallel dilated convolution layers for efficient segmentation. ESPNet [92] proposes an efficient spatial pyramid, which is based on filter factorization techniques: point-wise convolutions and spatial pyramid of dilated convolutions, to replace the standard convolution. The efficient classification networks such as MobileNet [52] and ShuffleNet [170] and IGCNet [169], are also applied to accelerate segmentation. In addition, ICNet (image cascade network) [174] exploits the efficiency of processing low-resolution images and high inference quality of high-resolution ones, achieving a trade-off between efficiency and accuracy.

Semantic video segmentation requires dense labeling for all pixels in each frame of a video sequence. It is different from video object detection [135] and video object segmentation [24], which only focus on the recognition of foreground objects. Previous work can be summarized into two streams.

The first one focuses on improving the accuracy by exploiting the temporal relations and the unlabelled data in the video sequence. Nilsson and Sminchiesescu [103] employ a gated recurrent unit to propagate semantic labels to unlabeled frames. Other

works like NetWarp [37], STFCN [33], and SVP [82] also employ optical-flow or recurrent units to fuse the results of several frames during inferring to improve the segmentation accuracy. Recently, Zhu *et al.* [184] propose to use a motion estimation network to propagate ground truth labels to unlabeled frames as data augmentation and achieve state-of-the-art performance with segmentation accuracy. These methods can achieve significant performance but are hard to apply to mobile devices.

Another stream pays attention to reduce the computational cost by re-using the feature maps in the neighboring frames. ClockNet [123] proposes to copy the feature map to the next frame directly, therefore, can reduce the computational cost. DFF [182] employs the optical flow to warp the feature map between the keyframe and other frames. Xu *et al.* [149] further propose to use an adaptive keyframe selection policy while Zhu *et al.* [183] find out that propagating partial region in the feature map can get better performance. Li *et al.* [71] propose a low-latency video segmentation network by optimizing both the keyframe selection and the adaptive feature propagation. Accel [57] proposes a network fusion policy to use a large model to predict the keyframe and use a compact one in other frames. They also employ optical flow to propagate results of the keyframe for results fusion. Keyframe-based methods may produce different quantity results between keyframes and other frames. Besides, the keyframe-based methods need to refer to previous prediction results during the inference process, which may cause unbalanced latency.

2.2.2 Depth Estimation

Depth estimation from a monocular image is essentially an ill-posed problem, which requires an expressive model with high reasoning ability. Previous works depend on hand-crafted features [119].

Since Eigen et al. [31] proposed to use deep learning to predict depth maps, following works [78, 79, 69, 36] benefit from the increasing ability of deep models and achieve good results. Besides, Fei [34] proposed a semantically informed geometric loss while Yin et al. [143] introduced a virtual normal loss to exploit the structure information. As in semantic segmentation, some works try to replace the encoder with efficiency backbones [143, 125, 144] to decrease the computational cost, but often suffer from the training problem limited by the ability of the compact network. FastDepth [144] also pays attention to training the compact networks for depth estimation. They propose an efficient and lightweight encoder-decoder network architecture and apply network pruning to further reduce computational complexity and latency. Here we focus on adding extra constraints during training to improve the performance of lightweight networks.

2.2.3 Object Detection

Object detection is a fundamental task in computer vision, in which one needs to regress a bounding box as well as predict a category label for each instance of interest in

an image. Early works [38, 112] achieved good performance by first predict proposals and then refine the bounding box as well as predict a category label. Effort was also spent on improving detection efficiency such as Yolo [111], and SSD [83]. They use a one-stage method and design lightweight network structures. RetinaNet [76] solves the problem of unbalance samples to some extent by proposing the focal loss, which makes the results of one-stage methods comparable to two-stage ones. Most of the above detectors rely on a set of pre-defined anchor boxes, which decreases the training samples and makes the detection network sensitive to hyperparameters. Recently, anchor free methods show promises, *e.g.*, FCOS [133]. FCOS employs a fully convolutional framework, and predict bounding box based on every pixel like in semantic segmentation, which solves the object detection task as a dense prediction problem. In this work, we apply the structured knowledge distillation method with the FCOS framework, as it is simple and can achieve good performance.

2.3 Auxiliary supervision

2.3.1 Knowledge Distillation

Knowledge distillation [50] is a way of transferring knowledge from a heavy model to a compact model to improve the performance of compact networks. It has been applied to image classification by using the class probabilities produced from the cumbersome model as "soft targets" for training the compact model [7, 50, 136] or transferring the intermediate feature maps [115, 166]. There are also other applications, including object detection [67], pedestrian re-identification [23] and so on. The MIMIC [67] method distills a compact object detection network by making use of a two-stage Faster-RCNN [112]. They align the feature map at pixel level and do not make use of the structure information among pixels.

In [146], a local similarity map is constructed to minimize the discrepancy of segmented boundary information between the teacher and student network, where the Euclidean distance between the center pixel and the 8-neighborhood pixels is used as knowledge for transferring. The work of [147] may be seen as a special case of our proposed pair-wise distillation.

The work in [142] focuses on the intra-class feature variation among the pixels with the same label, where the set of cosine distance between each pixel's feature and its corresponding class-wise prototype is constructed to transfer the structural knowledge. Besides, an auto-encoder is used to compress features [46], and the feature adaptor is employed to mitigate the feature mismatching between the teacher and student networks.

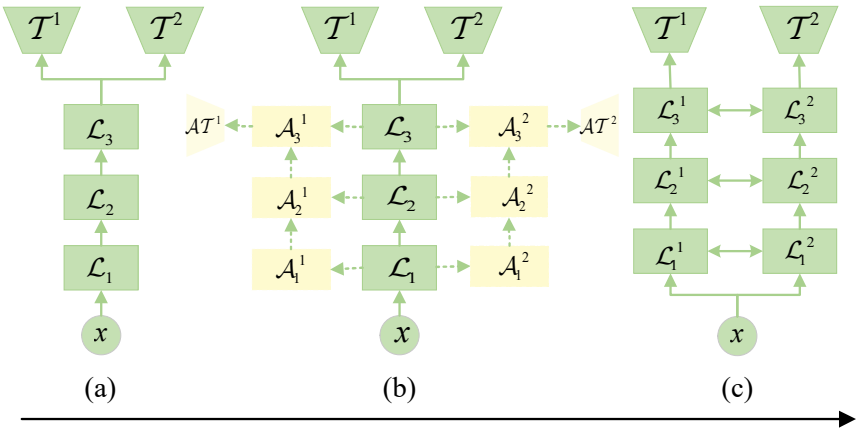


FIGURE 2.1. Comparison of two representative MTL methods and the proposed approach. We take two tasks as example. The two MTL categories take two extremes. (a) Hard parameter sharing: two tasks share the same layers to extract the features, and task-specific layers to handle different tasks. (c) Soft parameter sharing: each task employs an independent network, but apply message passing between specific layers. (b) The proposed auxiliary learning framework: The auxiliary modules provide extra inductive bias for specific tasks, and improve the training for the shared layers in the main network.

2.3.2 Auxiliary Loss

One straightforward way of adding auxiliary supervision is introducing additional losses in the intermediate layers, which serve to combat the vanishing gradient problem while providing regularization. The effectiveness of additional losses has been demonstrated in classification networks, like GoogLeNet [128], DSN [65], etc. Moreover, Zhao *et al.* [176] employ the additional loss added on top of the fourth layer of the encoder for the semantic segmentation task. Nekrasov *et al.* [98] use the auxiliary cells to accelerate the training of the decoders by adding multiple additional losses. These methods are usually sensitive to the positions and scales of the guidance signals. Knowledge distillation [50, 115, 164, 186, 86] can also be treated as adding auxiliary supervisions. The supervision signals are generated by the large model acting as a 'teacher'. The pioneering work [50] proposes to align the output logits between the teacher and the student. And the following works [115, 164] also introduce the idea of aligning the hidden feature maps. Different from the knowledge distillation methods, the proposed auxiliary module is easy to develop and does not need to pre-train a teacher network which is usually much deeper and maybe the upper bound of the performance.

Chapter 3

Structured Knowledge Distillation

3.1 Introduction

In this chapter, we investigate the knowledge distillation strategy for training efficient dense prediction networks by making use of large networks. We start from the straightforward scheme, pixel-wise distillation, which applies the distillation scheme adopted for image classification and performs knowledge distillation for each pixel *separately*.

Here we further propose to distill *structured* knowledge from large networks to compact networks, taking into account the fact that dense prediction is a structured prediction problem. Specifically, we study two structured distillation schemes: *i*) *pair-wise* distillation that distills the pairwise similarities by building a static graph; and *ii*) *holistic* distillation that uses adversarial training to distill holistic knowledge. The effectiveness of our knowledge distillation approaches is demonstrated by experiments on three dense prediction tasks: semantic segmentation, depth estimation, and object detection.

3.2 Background

Dense prediction is a family of fundamental problems in computer vision, which learns a mapping from input images to complex output structures, including semantic segmentation, depth estimation, and object detection, among many others. One needs to assign category labels or regress specific values for each pixel given an input image to form the structured outputs.

In general, these tasks are significantly more challenging to solve than image-level prediction problems, thus often requiring networks with large capacity to achieve satisfactory accuracy. On the other hand, compact models are desirable for enabling computing edge devices with limited computation resources.

Deep neural networks have been the dominant solutions since the invention of fully-convolutional neural networks (FCNs) [122]. Subsequent approaches, *e.g.*, DeepLab [17], PSPNet [176], RefineNet [74], and FCOS [133] follow the design of FCNs to optimize energy-based objective functions related to different tasks, having achieved significant improvement in accuracy, often with cumbersome models and expensive computation.

Recently, the design of neural networks with compact model sizes, light computation cost and high performance has attracted much attention due to the need for applications on mobile devices. Most current efforts have been devoted to designing lightweight networks especially for dense prediction tasks or borrowing the design from classification networks, *e.g.*, ENet [106], ESPNet [106] and ICNet [174] for semantic segmentation, YOLO [111] and SSD [83] for object detection, and FastDepth [144] for depth estimation.

Strategies such as pruning [144], knowledge distillation [67, 147] are applied to helping the training of compact networks by making use of cumbersome networks.

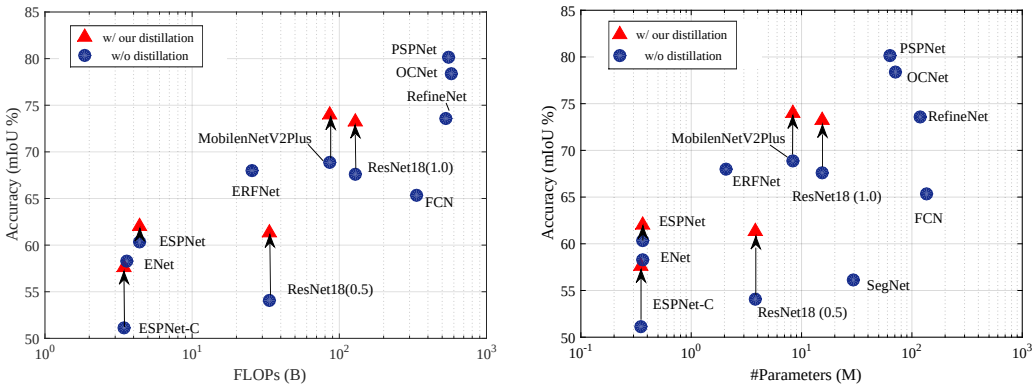


FIGURE 3.1. An example on the semantic segmentation task shows comparisons in terms of computation complexity, number of parameters and mIoU for different networks on the Cityscapes test set. The FLOPs is calculated with the resolution of 512×1024. The red triangles are the results of our distillation method while others are without distillation. Blue circles are collected from FCN* [122], RefineNet [74], SegNet [8], ENet [106], PSPNet [176], ERFNet [114], ESPNet [92], MobileNetV2Plus [81], and OCNet [162]. With our proposed distillation method, we can achieve a higher mIoU, with no extra FLOPs and parameters.

Knowledge distillation has proven effective in training compact models for classification tasks [50, 115]. Most previous works [67, 147] directly apply distillation at each pixel separately to transfer the class probability or extracted feature embedding of the corresponding pixel produced from the cumbersome network (teacher) to the compact network (student) for dense prediction tasks. Note that, such a pixel-wise distillation scheme neglects the important structural information.

Considering the characteristic of dense prediction problem, here we present structured knowledge distillation and transfer the structure information with two schemes, *pair-wise distillation* and *holistic distillation*. The *pair-wise distillation* scheme is motivated by the widely-studied pair-wise Markov random field framework [70] for enforcing spatial labeling consistency. The goal is to align a static affinity graph which is computed to capture both short and long-range structure information among different locations from the compact network and the teacher network.

The *holistic distillation* scheme aims to align higher-order consistencies, which are

not characterized in the pixel-wise and pair-wise distillation, between output structures produced from the compact network and the teacher network. We adopt the adversarial training scheme, and a fully convolutional network, a.k.a. the discriminator, considers both the input image and the output structures to produce a holistic embedding that represents the quality of the structure. The compact network is encouraged to generate structures with similar embeddings as the teacher network. We distill the structure knowledge into the weights of discriminators.

Generative adversarial networks (GANs) have been widely studied in text generation [138, 157] and image synthesis [39, 60]. The conditional version [93] is successfully applied to image-to-image translation, such as style transfer [58] and image coloring [84]. The idea of adversarial learning is also employed in pose estimation [22], encouraging the human pose estimation result not to be distinguished from the ground-truth; and semantic segmentation [91, 94], encouraging the estimated segmentation map not to be distinguished from the ground-truth map. One challenge in [91, 94] is the mismatch between the generator's continuous output and the discrete true labels, making the discriminator in GAN be of very limited success. Different from [91, 94], in our approach, the employed GAN does not face this issue as the ground truth for the discriminator is the teacher network's logits, which are real-valued. We use adversarial learning to encourage the alignment between the output maps produced from the cumbersome network and the compact network. However, in the depth prediction task, the ground truth maps are not discrete labels. In [41], the authors use the ground truth maps as the real samples. Different from theirs, our distillation methods are trying to align the output of the cumbersome network and that of the compact network. The task loss calculated with ground truth is optional. When the labeled data is limited, given a well-trained teacher, our method can be applied to unlabelled data and may further improve the accuracy.

To this end, we optimize an objective function that combines a conventional task loss with the distillation terms. The main contributions of this paper can be summarized as follows.

- We study the knowledge distillation strategy for training accurate and efficient networks for dense prediction.
- We present two structured knowledge distillation schemes, pair-wise distillation, and holistic distillation, enforcing pair-wise and high-order consistency between the outputs of the compact and teacher networks.
- We demonstrate the effectiveness of our approach by improving recently-developed state-of-the-art compact networks on three different dense prediction tasks: semantic segmentation, depth estimation, and object detection. Taking semantic segmentation as an example, the performance gain is illustrated in Figure 3.1.

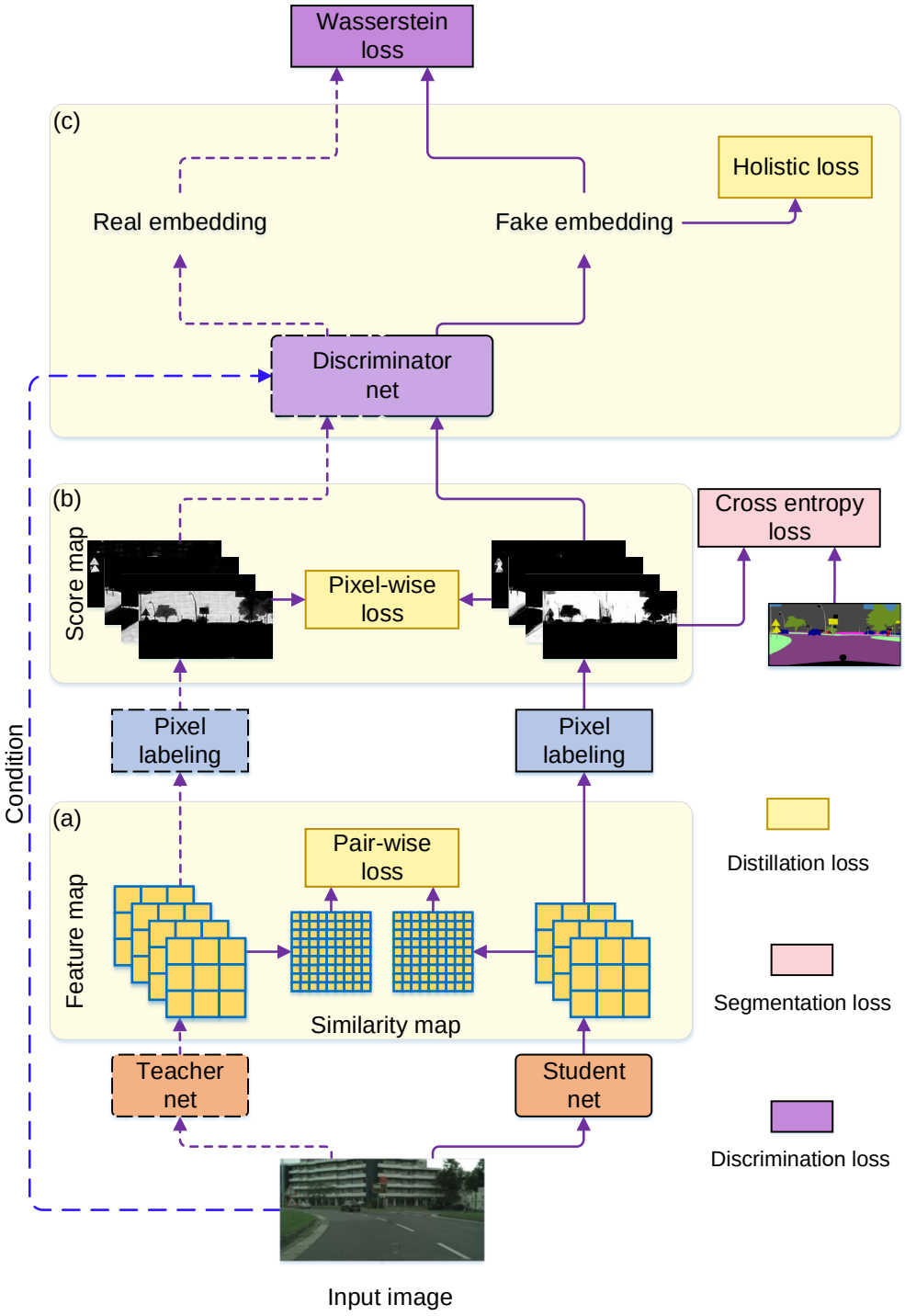


FIGURE 3.2. Our distillation framework with the semantic segmentation task as an example. (a) Pair-wise distillation; (b) Pixel-wise distillation; (c) Holistic distillation. In the training process, we keep the cumbersome network fixed as our teacher net, and only the student net and the discriminator net are optimized. The student net with a compact architecture is trained with three distillation terms and a task-specific loss, *e.g.*, the cross-entropy loss for semantic segmentation.

3.3 Method

In this section, we first introduce the structured knowledge distillation method for semantic segmentation, a task of assigning a category label to each pixel in the image from C categories. A segmentation network takes a RGB image $\mathbf{I}$ of size $W \times H \times 3$ as the input; then it computes a feature map $\mathbf{F}$ of size $W' \times H' \times N$, where N is the number of channels. Then, a classifier is applied to compute the segmentation map $\mathbf{Q}$ of size $W' \times H' \times C$ from $\mathbf{F}$, which is upsampled to the spatial size $W \times H$ of the input image to obtain the segmentation results. We extend our method to other two dense prediction tasks: depth estimation and object detection.

Pixel-wise distillation. We apply the knowledge distillation strategy [50] to transfer the knowledge of the large teacher segmentation network T to a compact segmentation network S for better training the compact segmentation network. We view the segmentation problem as a collection of separate pixel labeling problems, and directly use knowledge distillation to align the class probability of each pixel produced from the compact network. We follow [50] and use the class probabilities produced from the teacher model as soft targets for training the compact network.

The loss function is given as follows,

$$\ell_{pi}(\mathsf{S}) = \frac{1}{W' \times H'} \sum_{i \in \mathcal{R}} \mathrm{KL}(\mathbf{q}_i^s \| \mathbf{q}_i^t), \tag{3.1}$$

where $\mathbf{q}_i^s$ represents the class probabilities of the ith pixel produced from the compact network S. $\mathbf{q}_i^t$ represents the class probabilities of the ith pixel produced from the cumbersome network T. $\mathrm{KL}(\cdot)$ is the Kullback-Leibler divergence between two probabilities, and $\mathcal{R} = \{1, 2, \ldots, W' \times H'\}$ denotes all the pixels.

3.3.1 Structured Knowledge Distillation

In addition to the above straightforward pixel-wise distillation, we present two structured knowledge distillation schemes—pair-wise distillation and holistic distillation—to transfer structured knowledge from the teacher network to the compact network. The pipeline is illustrated in Figure 3.2.

Pair-wise distillation. Inspired by the pair-wise Markov random field framework that is widely adopted for improving spatial labeling contiguity, we propose to transfer the pair-wise relations, specifically pair-wise similarities in our approach, among spatial locations.

We build an affinity graph to denote the spatial pair-wise relations, in which, the nodes represent different spatial locations and the connection between two nodes represents the similarity. We denote the connection range α and the granularity β of each node to control the size of the static affinity graph. For each node, we only consider the similarities with top-α near nodes according to spatial distance (here we use the Chebyshev distance) and aggregate β pixels in a spatial local patch to represent the feature of this node as illustrate in Figure 3.3.

Here for a $W' \times H' \times C$ feature map, $W' \times H'$ is the spatial resolution. With the granularity β and the connection range α, the affinity graph contains $\frac{W' \times H'}{\beta}$ nodes with $\frac{W' \times H'}{\beta} \times \alpha$ connections.

Let a_{ij}^t denote the similarity between the ith node and the jth node the produced from the teacher network T and a_{ij}^s denote the similarity between the ith node and the jth node produced from the compact network S. We adopt the squared difference to formulate the pair-wise similarity distillation loss,

$$\ell_{pa}(\mathsf{S}) = \frac{\beta}{W' \times H' \times \alpha} \sum_{i \in \mathcal{R}'} \sum_{j \in \alpha} (a_{ij}^s - a_{ij}^t)^2. \tag{3.2}$$

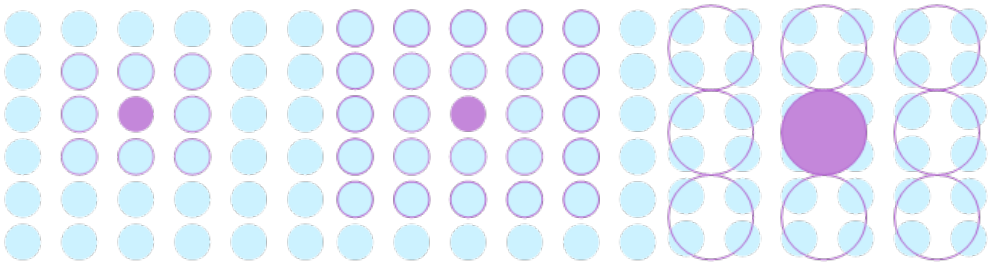
where $\mathcal{R}' = \{1, 2, \ldots, \frac{W' \times H'}{\beta}\}$ denotes all the nodes. In our implementation, we use an average pool to aggregate $\beta \times C$ features in one node to be $1 \times C$, and the similarity between two nodes is simply computed from the aggregated features $\mathbf{f}_i$ and $\mathbf{f}_j$ as

$$a_{ij} = \mathbf{f}_i^\top \mathbf{f}_j / (\|\mathbf{f}_i\|_2 \|\mathbf{f}_j\|_2),$$

which empirically works well.

Holistic distillation. We align the high-order relations between the segmentation maps produced from the teacher and student networks. The holistic embeddings of the segmentation maps are computed as the representations.

We adopt conditional generative adversarial learning [93] for formulating the holistic distillation problem. The compact net is regarded as a generator conditioned on the input RGB image $\mathbf{I}$, and the predicted segmentation map $\mathbf{Q}^s$ is regarded as a fake sample. We expect that $\mathbf{Q}^s$ is similar to $\mathbf{Q}^t$, which is the segmentation map predicted by the teacher and is regarded as the real sample. The GAN is usually suffering from the unstable gradient in training the generator due to the discontinuous *Jensen-Shannon* (JS) divergence, along with other common distance and divergence. The Wasserstein distance [40] (also known as the Earth Mover distance) can be used to measure the difference between two distributions. The Wasserstein distance is defined as the minimum cost to converge the model distribution $p_s(\mathbf{Q}^s)$ to the real distribution $p_t(\mathbf{Q}^t)$.



(A) (B) (C)

FIGURE 3.3. Illustrations of the connection range α and the granularity β of each node. (A) $\alpha = 9$, $\beta = 1$, (B) $\alpha = 25$, $\beta = 1$, (C)$\alpha = 9$, $\beta = 4$.

This can be written as follows:

$$\ell_{ho}(\mathsf{S},\mathsf{D}) = \mathbb{E}_{\mathbf{Q}^s\sim p_s(\mathbf{Q}^s)}[\mathsf{D}(\mathbf{Q}^s|\mathbf{I})] - \mathbb{E}_{\mathbf{Q}^t\sim p_t(\mathbf{Q}^t)}[\mathsf{D}(\mathbf{Q}^t|\mathbf{I})], \tag{3.3}$$

where $\mathbb{E}[\cdot]$ is the expectation operator, and $\mathsf{D}(\cdot)$ is an embedding network, acting as the discriminator in GAN, which projects $\mathbf{Q}$ and $\mathbf{I}$ together into a holistic embedding score. The Lipschitz requirement is satisfied by the gradient penalty.

The segmentation map and the conditional RGB image are concatenated as the input of the embedding network D. D is a fully convolutional neural network with five convolutions. Two self-attention modules are inserted between the final three layers to capture the structure information [167]. A batch normalization layer is added before the concatenation input to handle the different scales of the RGB image and the logits.

Such a discriminator can produce a holistic embedding representing how well the input image and the segmentation map match. We further add a pooling layer to pool the holistic embedding into a score. As we employ the Wasserstein distance in the adversarial training, the discriminator is trained to produce a higher score in terms of the output segmentation map from the teacher net and produce lower scores in terms of the ones from the student. In this processing, we encode the knowledge of evaluating the quality of a segmentation map into the discriminator. The student is trained with the regularization of achieving a higher score using the discriminator, improving the performance of the student.

3.3.2 Optimization

The overall objective function consists of a standard multi-class cross-entropy loss $\ell_{mc}(\mathsf{S})$ with pixel-wise and structured distillation terms[1]

$$\ell(\mathsf{S},\mathsf{D}) = \ell_{mc}(\mathsf{S}) + \lambda_1(\ell_{pi}(\mathsf{S}) + \ell_{pa}(\mathsf{S})) - \lambda_2\ell_{ho}(\mathsf{S},\mathsf{D}), \tag{3.4}$$

where λ_1 and λ_2 are hyper-parameters. Note that we have not carefully tuned these hyper-parameters. Preliminary results show that the final performance is not sensitive to these hyper-parameters in a wide range. For simplicity, we set λ_1 and λ_2 to be 10 and 0.1, making these loss value ranges comparable.

We minimize the objective function with respect to the parameters of the compact segmentation network S, while maximizing it with respect to the parameters of the discriminator D, which is implemented by iterating the following two steps:

- **Train the discriminator** D. Training the discriminator is equivalent to minimizing $\ell_{ho}(\mathsf{S},\mathsf{D})$. D aims to give a high embedding score for the real samples from the teacher net and low embedding scores for the fake samples from the student net.

[1] The objective function is the summation of the losses over the mini-batch of training samples. For ease of exposition, we omit the summation operation.

- **Train the compact segmentation network** S. Given the discriminator network, the goal is to minimize the multi-class cross-entropy loss and the distillation loss relevant to the compact segmentation network:

$$\ell_{mc}(\mathsf{S}) + \lambda_1(\ell_{pi}(\mathsf{S}) + \ell_{pa}(\mathsf{S})) - \lambda_2 \ell^s_{ho}(\mathsf{S}),$$

where

$$\ell^s_{ho}(\mathsf{S}) = \mathbb{E}_{\mathbf{Q}^s \sim p_s(\mathbf{Q}^s)}[\mathsf{D}(\mathbf{Q}^s|\mathbf{I})]$$

is a part of $\ell_{ho}(\mathsf{S}, \mathsf{D})$ given in Equation (3.3), and we expect S to achieve a higher score under the evaluation of D.

3.3.3 Extension to Other Dense Prediction Tasks

Dense prediction learns a mapping from an input RGB image $\mathbf{I}$ with size $W \times H \times 3$ to a per-pixel output $\mathbf{Q}$ with size $W \times H \times C$. In the semantic segmentation, the number of the output channels is C, which is equal to the number of semantic classes.

For the object detection task, for each pixel, we predict the c^* classes with one background class, as well as a 4D tensor $t^* = (l, t, r, b)$ representing the distance from the location to the four sides of the bounding box. We follow the task loss in FCOS [133], and combine with the distillation terms as regularization.

In the depth estimation task, the depth estimation task can be solved as a classification task, as the continuous depth values can be divided into C discrete categories [13]. After we get the predicted probability, we apply a weighted cross-entropy loss following [143]. The pair-wise distillation can be directly applied to the intermediate feature map. The holistic distillation uses the depth map as input. We can use the ground truth as the real samples of the GAN in the depth estimation task because it is a continuous map. However, in order to apply our method to unlabelled data, we still consider the depth map from the teacher as our real samples.

3.4 Experiments

In this section, we choose the typical structured output prediction task- semantic segmentation as an example to verify the effectiveness of the structured knowledge distillation. We discuss and explore how does structured knowledge distillation work and how well does structured knowledge distillation work in Section 3.4.1 and Section 3.4.1.

The structured knowledge distillation can be applied to other structured output prediction tasks under the FCN framework. In Section 3.4.3 and Section 3.4.2, we apply our distillation method to strong baselines in object detection and depth estimation tasks with minor modifications.

3.4.1 Semantic Segmentation

Implementation Details

Network structures. We use the segmentation architecture PSPNet [176] with a ResNet101 [42] as the teacher network T.

We study recent public compact networks and employ several different architectures to verify the effectiveness of the distillation framework. We first use ResNet18 as a basic student network and conduct ablation studies on it. Then, we employ the MobileNetV2Plus [81], which is based on a pretrained MobileNetV2 [117] model on the ImageNet dataset. We also test the ESPNet-C [92] and ESPNet [92] models which are very lightweight.

Training setup. Most segmentation networks in this chapter are trained using stochastic gradient descent (SGD) with the momentum (0.9) and the weight decay (0.0005) for 40000 iterations. The learning rate is initialized to be 0.01 and is multiplied by $(1 - \frac{iter}{max-iter})^{0.9}$. We random cut the the images into 512×512 as the training input. Standard data augmentation methods are applied during training, such as random scaling (from 0.5 to 2.1) and random flipping. We follow the settings in the corresponding publications [92] to reproduce the results of ESPNet and ESPNet-C, and train the compact networks under our distillation framework.

Dataset

Cityscapes. The Cityscapes dataset [25] is collected for urban scene understanding and contains 30 classes with only 19 classes used for evaluation. The dataset contains $5,000$ high-quality pixel-level finely annotated images and $20,000$ coarsely annotated images. The finely annotated $5,000$ images are divided into $2,975$, 500, $1,525$ images for training, validation and testing. We only use the finely annotated dataset in our experiments.

CamVid. The CamVid dataset [11] contains 367 training and 233 testing images. We evaluate the performance over 11 different classes such as building, tree, sky, car, road, etc. and ignore the 12th class that contains unlabeled data.

ADE20K. The ADE20K dataset [179] contains 150 classes of diverse scenes. The dataset is divided into $20K/2K/3K$ images for training, validation and testing.

Evaluation Metrics

We use the following metrics to evaluate the segmentation accuracy, as well as the model size and the efficiency.

The *Intersection over Union (IoU)* score is calculated as the ratio of interval and union between the ground truth mask and the predicted segmentation mask for each class. We use the mean IoU of all classes (mIoU) to study the distillation effectiveness. We also report the class IoU to study the effect of distillation on different classes. *Pixel accuracy* is the ratio of the pixels with the correct semantic labels to the overall pixels.

The *model size* is represented by the number of network parameters. and the *Complexity* is evaluated by the sum of floating-point operations (FLOPs) in one forward on the fixed input size.

Ablation Study

The effectiveness of distillations. We examine the effect of enabling and disabling different components of our distillation system. The experiments are conducted on ResNet18 with its variant ResNet18 (0.5) representing a width-halved version of ResNet18 on the Cityscapes dataset. In Table 3.1, the results of different settings for the student net are the average results of the final epoch from three runs.

From Table 3.1, we can see that distillation can improve the performance of the student network, and distilling the structure information helps the student learn better. With the three distillation terms, the improvements for ResNet18 (0.5), ResNet18 (1.0) and ResNet18 (1.0) with weights pre-trained from the ImageNet dataset are 6.26%, 5.74% and 2.9%, respectively, which indicates that the effect of distillation is more pronounced for the smaller student network and networks without initialization with the weight pre-trained from the ImageNet. Such an initialization method is also a way to transfer the knowledge from other sources (ImageNet). The best mIoU of the holistic distillation for ResNet18 (0.5) reaches 62.7% on the validation set.

On the other hand, one can see that each distillation scheme lead to higher mIoU score. This implies that the three distillation schemes make complementary contributions for better training the compact network.

The affinity graph in pair-wise distillation. In this section, we discuss the impact of the connection range α and the granularity of each node β in building the affinity

TABLE 3.1. The effect of different components of the loss in the proposed method. PI: pixel-wise distillation; PA: pair-wise distillation; HO: holistic distillation; ImN: initial from the pre-trained weight on the ImageNet.

Method	Validation mIoU (%)	Training mIoU (%)
Teacher	78.56	86.09
ResNet18 (0.5)	55.37 ± 0.25	60.67 ± 0.37
+ PI	57.07 ± 0.69	62.33 ± 0.66
+ PI + PA	61.52 ± 0.09	66.03 ± 0.07
+ PI + PA + HO	$\mathbf{62.35 \pm 0.12}$	$\mathbf{66.72 \pm 0.04}$
ResNet18 (1.0)	57.50 ± 0.49	62.98 ± 0.45
+ PI	58.63 ± 0.31	64.32 ± 0.32
+ PI + PA	62.97 ± 0.06	68.97 ± 0.03
+ PI + PA + HO	$\mathbf{64.68 \pm 0.11}$	$\mathbf{70.04 \pm 0.06}$
ResNet18 (1.0)	69.10 ± 0.21	74.12 ± 0.19
+ PI +ImN	70.51 ± 0.37	75.10 ± 0.37
+ PI + PA +ImN	71.78 ± 0.03	77.66 ± 0.02
+ PI + PA + HO +ImN	$\mathbf{74.08 \pm 0.13}$	$\mathbf{78.75 \pm 0.02}$

graph. Calculating the pair-wise similarity among each pixel in the feature map will form the most complete affinity graph, but lead to high computational complexity. We fix the node to be one pixel and change the connection range α from the fully connected graph to the local sub-graph. Then, we keep the connection range α to be fully connected and use a local patch to denote each node to change the granularity β from fine to coarse. The result are shown in Table 3.2. The results of different settings for the pair-wise distillation are the average results from three runs. We employ a ResNet18 (1.0) with the weight pre-trained from ImageNet as the student network. All the experiments are performed with both pixel-wise distillation and pair-wise distillation, but the sizes of the affinity graph in pair-wise distillation vary.

From Table 3.2, we can see increasing the connection range can help improve the distillation performance. With the fully connected graph, the student can achieve around 71.37% mIoU. The best β is 2×2, which is slightly better than the finest affinity graph, but the connections are significantly decreased. Using a small local patch to denote a node and calculate the affinity graph may form a more stable correlation between different locations. One can choose to use the local patch to cut off the number of the nodes, instead of decreasing the connection range for a better trade-off between efficiency and accuracy.

To include more structure information, we fuse pair-wise distillation items with different affinity graphs. Three pair-wise fusion ways are introduced: alpha-fusion, beta-fusion, and different scale feature fusion. The details can be seen in Table 3.2. We can see that combining more affinity graphs may slightly improve the performance, but also introduces extra computational cost during training.

The adversarial training in holistic distillation. In this section, we illustrate that GAN can distill holistic knowledge. Firstly, we compare the results between conditional GAN and unconditional GAN. Then we conduct ablation studies on the structure of the discriminator. As described in Section 3.3.1, we concatenate the input image with the segmentation map as the input of the discriminator. To deal with the different scales of the RGB image and the segmentation map, we add a BN layer on the concatenation input. The results are shown in Table 3.3. From this table, we could know that the conditional discriminator outperforms the unconditional discriminator. As the RGB images and the segmentation maps have different data scales, the batch normalization layer is essential.

As described in Section 3.3.1, we employ a fully convolutional network with five convolution blocks as our discriminator. Each convolution block has ReLU and BN layers, except for the final output convolution layer. We also insert two self-attention blocks in the discriminator to capture the structure information. The ability of the discriminator will affect the adversarial training, and we conduct experiments to discuss the impact of the discriminator's architecture. The results are shown in Table 3.4, and we use $AnLm$ to represent the architecture of the discriminator with n self-attention

TABLE 3.2. The impact of the connection range and node granularity. The shape of the output feature map is $H' \times W'$. We can see that keeping a fully connected graph is more helpful in pair-wise distillation.

Method	Validation mIoU(%)	Connections
Teacher	78.56	–
Resnet18 (1.0)	69.10 ± 0.21	–
$\beta = 1 \times 1$, $\alpha =$		
$W'/16 \times H'/16$	70.83 ± 0.12	$(W' \times H')^2/2^8$
$W'/8 \times H'/8$	70.94 ± 0.11	$(W' \times H')^2/2^6$
$W'/4 \times H'/4$	71.09 ± 0.07	$(W' \times H')^2/2^4$
$W'/2 \times H'/2$	71.15 ± 0.01	$(W' \times H')^2/4$
$W \times H$	71.37 ± 0.12	$(W' \times H')^2$
$\alpha = W' \times H'/\beta$, $\beta =$		
2×2	$\mathbf{71.78 \pm 0.03}$	$(W' \times H')^2/2^4$
4×4	71.24 ± 0.18	$(W' \times H')^2/2^8$
8×8	71.10 ± 0.36	$(W' \times H')^2/2^{12}$
16×16	71.11 ± 0.14	$(W' \times H')^2/2^{16}$
32×32	70.94 ± 0.23	$(W' \times H')^2/2^{20}$
Multi-scale pair-wise distillations		
alpha-fusion[1]	72.03 ± 0.26	$21 * (W' \times H')^2/2^8$
Beta-fusion[2]	71.91 ± 0.17	$273 * (W \times H)^2/2^{12}$
Feature-fusion[3]	$\mathbf{72.18 \pm 0.12}$	$3 * (W' \times H')^2/2^4$

[1] Different connection ranges fusion, with output feature size $H' \times W'$, $\beta = 2 \times 2$ and α as $W'/2 \times H'/2$, $W'/4 \times H'/4$ and $W'/8 \times H'/8$, respectively.
[2] Different node granulates fusion, with output feature size $H' \times W'$, β as 2×2, 4×4 and 8×8, respectively, and α as $W' \times H'/\beta$.
[3] Different feature levels fusion, with feature size $H' \times W'$, $2H' \times 2W'$, $4H' \times 4W'$, β as 2×2, 4×4 and 8×8, respectively, and α as maximum.

layers and m convolution blocks with BN layers. The detailed structure can be seen in Figure 3.4, and the red arrow represents a self-attention layer.

From Table 3.4, we can see adding self-attention layers can improve the average mean of mIoU, and adding more self-attention layers does not change the results much. We choose to add 2 self-attention blocks considering the performance, stability, and computational cost. With the same self-attention layer, a deeper discriminator can help the adversarial training.

To verify the effectiveness of adversarial training, we further explore the learning ability of three typical discriminators: the shallowest one (D_Shallow, i.e., A2L2), the one without attention layer (D_no_attention, i.e., A0L4) and ours (D_Ours, i.e., A2L4). The IoUs for different classes are listed in Table 3.5. It is clear that the self-attention layer can help the discriminator better capture the structure, therefore the accuracy of the students with the structure objects is improved.

In the adversarial training, the student, a.k.a. the generator, is trying to learn the distribution of the real samples (output of the teacher). Because we apply the Wasserstein distance to transfer the difference of two distributions into a more intuitive

TABLE 3.3. The effectiveness of the conditional discriminator in HO distillation. We choose ResNet18 (1.0) as the example student net. PI and PA are employed as the baseline **a**. **BN** represents a batch normalization layer inserted before the discriminator. **Conditional** represents that the RGB image is concatenated as the conditional input of the discriminator.

Method	BN	Conditional	HO	mIoU (%)
a				71.78
b			✓	72.31
c		✓	✓	71.03
d	✓		✓	73.02
e	✓	✓	✓	**74.08**

TABLE 3.4. We choose ResNet18 (1.0) as the example student net. An *AnLm* index represents n attention blocks with m residual blocks in the discriminator. The ability of the discriminator will affect the adversarial training.

Architecture Index	Validation mIoU (%)
Changing self-attention layers	
A4L4	73.46 ± 0.02
A3L4	73.70 ± 0.02
A2L4 (ours)	$\mathbf{74.08 \pm 0.13}$
A1L4	74.05 ± 0.55
A0L4	72.85 ± 0.01
Removing convolution blocks	
A2L4 (ours)	$\mathbf{74.08 \pm 0.13}$
A2L3	73.35 ± 0.10
A2L2	72.24 ± 0.42

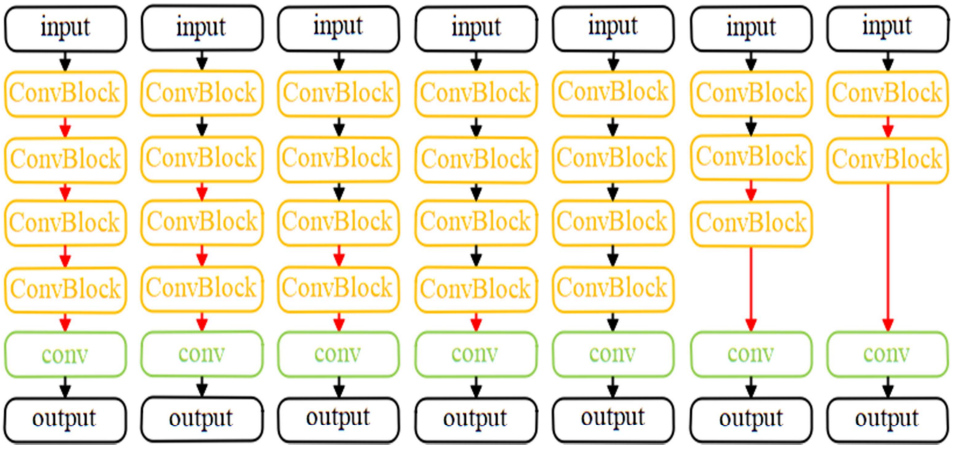


FIGURE 3.4. We show 7 different architectures of the discriminator. The red arrow represents a self-attention layer. The orange block denotes a residual block with stride 2. We add an average pooling layer to the output block to obtain the final score.

score, we can see the score are highly relevant to the quality of the segmentation maps. We use a well-trained discriminator D (A2L4) to evaluate the score of a segmentation map. For each image, we feed five segmentation maps, output by the teacher net, the student net w/o holistic distillation, and the student nets w/ holistic distillation under three different discriminator architectures (listed in Table 3.5) into the discriminator D, and compare the distribution of embedding scores.

TABLE 3.5. We choose ResNet18 (1.0) as the example student net. Class IoU with three different discriminator architectures is reported. The self-attention layer can significantly improve the accuracy of structured objects, such as "truck", "bus", "train", and "motorcycle".

Class	mIoU	road	sidewalk	building	wall	fence	pole	Tra. lig.	Tra. sign	veget.
D_Shallow	72.28	97.31	80.07	91.08	36.86	50.93	62.13	66.4	76.77	91.73
D_no_attention	72.69	**97.36**	**80.22**	**91.62**	**45.16**	**56.97**	62.23	**68.09**	**76.38**	**91.94**
D_Ours	74.10	97.15	79.17	91.60	44.84	56.61	**62.37**	67.37	76.34	91.91
class	terrain	sky	person	rider	car	truck	bus	train	motor.	bicycle
D_Shallow	60.14	93.76	79.89	55.32	93.45	69.44	73.83	69.54	48.98	75.78
D_no_attention	**62.98**	**93.84**	**80.1**	**57.35**	93.45	68.71	75.26	56.28	47.66	75.59
D_Ours	58.67	93.79	79.9	56.61	**94.3**	**75.83**	**82.87**	**72.03**	**50.89**	**75.72**

We evaluate the validation set and calculate the average score difference between different student nets and the teacher net, the results are shown in Table 3.6. With holistic distillation, the segmentation maps produced from the student net can achieve a similar score to the teacher, indicating that GAN helps distill the holistic structure knowledge.

We also draw a histogram to show the score distribution of the segmentation map across the validation set in Figure 3.5. The well-trained D can assign a higher score to a high-quality segmentation map, and the three student nets with the holistic

TABLE 3.6. We choose ResNet18 (1.0) as the example student net. The embedding score difference and mIoU on the validation set of Cityscapes.

Method	Score difference	mIoU
Teacher	0	78.56
Student w/o D	2.28	69.21
w/ D_no_attention	0.23	72.69
w/ D_shallow	0.46	72.28
w/ D_ours	0.14	74.10

TABLE 3.7. Comparison of feature transfer MIMIC [115, 67], attention transfer [166], and local pair-wise distillation [147] against our pair-wise distillation. The segmentation is evaluated by mIoU (%). PI: pixel-wise distillation. MIMIC: using a 1×1 convolution for feature distillation. AT: attention transfer for feature distillation. LOCAL: The local similarity distillation method. PA: our pair-wise distillation. ImN: initializing the network from the weights pre-trained on the ImageNet dataset.

Method	ResNet18 (0.5)	ResNet18 (1.0) + ImN
w/o distillation	55.37	69.10
+ PI	57.07	70.51
+ PI + MIMIC	58.44	71.03
+ PI + AT	57.93	70.70
+ PI + LOCAL	58.62	70.86
+ PI + PA	**61.52**	**71.78**

distillation can generate segmentation maps with higher scores and better quality. Adding self-attention layers and more convolution blocks help the student net to imitate the distribution of the teacher net, and attain better performance.

Feature and local pair-wise distillation. We compare a few variants of the pair-wise distillation:

- Feature distillation by MIMIC [115, 67]: We follow [67] to align the features of each pixel between T and S through a 1×1 convolution layer to match the dimension of the feature
- Feature distillation by attention transfer [166]: We aggregate the response maps into a so-called attention map (single channel), and then transfer the attention map from the teacher to the student.
- Local pair-wise distillation [147]: This method can be seen as a special case of our pair-wise distillation, which only covers a small sub-graph (8-neighborhood pixels for each node).

We replace our pair-wise distillation by the above three distillation schemes to verify the effectiveness of our pair-wise distillation. From Table 3.7, we can see that

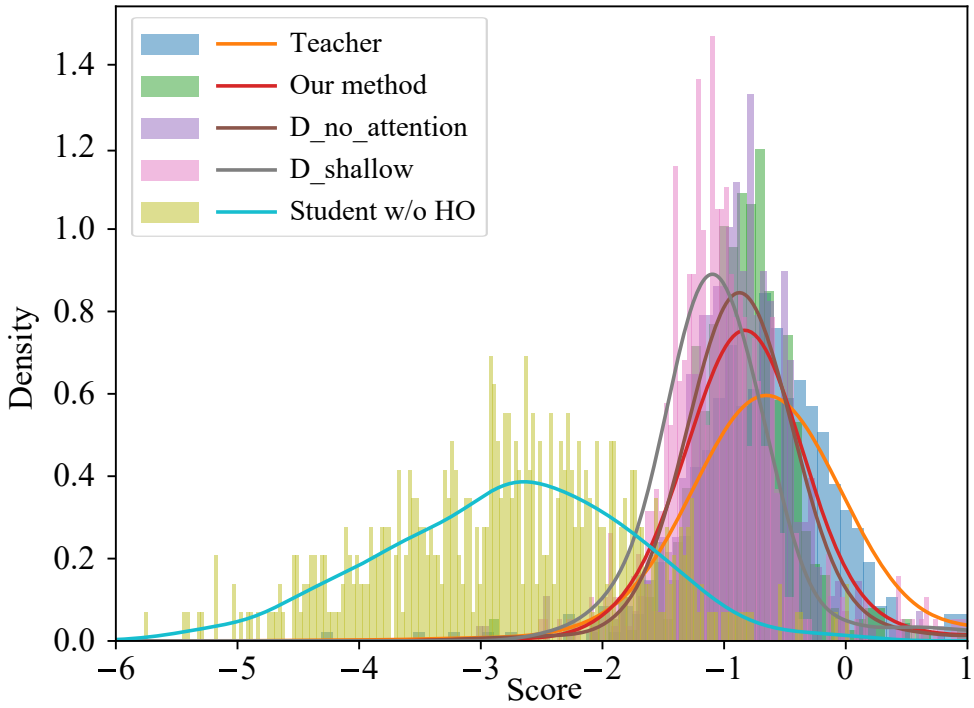


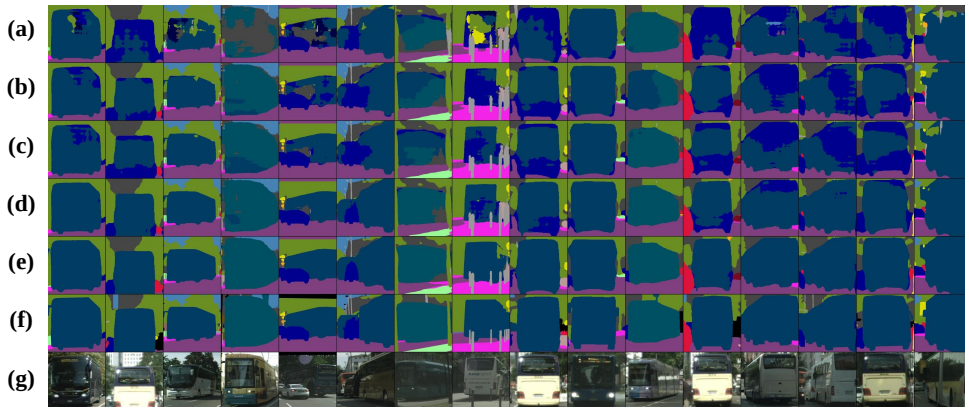
FIGURE 3.5. The score distribution of segmentation maps generated by different student nets evaluated by a well-trained discriminator. With adversarial training, the distributions of the segmentation map become closer to the teacher; and our method (the red one) is the closest one to the teacher (the orange one).

our pair-wise distillation method outperforms all the other distillation methods. The superiority over feature distillation schemes: MIMIC [67] and attention transfer [166], which transfers the knowledge for each pixel separately, comes from that we transfer the structured knowledge other than aligning the feature for each individual pixel. The superiority to the local pair-wise distillation shows the effectiveness of our fully-connected pare-wise distillation which is able to transfer the overall structure information other than a local boundary information [147].

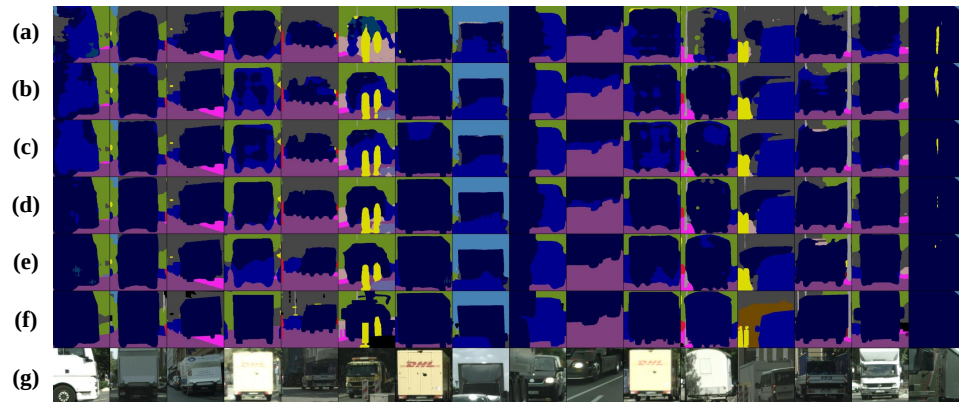
Segmentation Results

Cityscapes. We apply our structure distillation method to several compact networks: MobileNetV2Plus [81] which is based on a MobileNetV2 model, ESPNet-C [92] and ESPNet [92] which are carefully designed for mobile applications. Table 3.8 presents the segmentation accuracy, the model complexity, and the model size. FLOPs[2] is calculated on the resolution 512×1024 to evaluate the complexity. # parameters is the number of network parameters. We can see that our distillation approach can improve the results over 5 compact networks: ESPNet-C and ESPNet [92], ResNet18 (0.5), ResNet18 (1.0), and MobileNetV2Plus [81]. For the networks without pre-training, such as ResNet18 (0.5) and ESPNet-C, the improvements are very significant with 7.3% and 6.6%, respectively. Compared with MD (Enhanced) [147] that uses the

[2]FLOPs is calculated with the PyTorch version implementation [1].



(A) Bus



(B) Truck

FIGURE 3.6. Segmentation results for structured objects with ResNet18 (1.0) trained with different discriminators. (a) W/o holistic distillation, (b) W/ D_shallow, (c) W/ D_no_attention, (d) Our method, (e) Teacher net, (f) Ground truth, (g) Image. One can see that a strong discriminator can help the student learn structure objects better. With the attention layers, labels of the objects are more consistent.

pixel-wise and local pair-wise distillation schemes over MobileNet, our approach with the similar network MobileNetV2Plus achieves higher segmentation quality (74.5 vs 71.9 on the validation set) with a little higher computation complexity and a much smaller model size.

Figure 3.7 shows the IoU scores for each class over MobileNetV2Plus. Both the pixel-wise and structured distillation schemes improve the performance, especially for the categories with low IoU scores. In particular, the structured distillation (pair-wise and holistic) has significant improvement for structured objects, *e.g.*, 17.23% improvement for Bus and 10.03% for Truck. The qualitative segmentation results in Figure 3.8 visually demonstrate the effectiveness of our structured distillation for structured objects, such as trucks, buses, persons, and traffic signs.

CamVid. Table 3.9 shows the performance of the student networks w/o and w/ our distillation schemes and state-of-the-art results. We train and evaluate the student networks w/ and w/o distillation at the resolution 480 × 360 following the setting

TABLE 3.8. The segmentation results on the testing, validation (Val.), and training (Tra.) set of Cityscapes.

Method	#Params (M)	FLOPs (B)	Test §	Val.
Current state-of-the-art results				
ENet [106] †	0.3580	3.612	58.3	n/a
ERFNet [156] ‡	2.067	25.60	68.0	n/a
FCN [122] ‡	134.5	333.9	65.3	n/a
RefineNet [74] ‡	118.1	525.7	73.6	n/a
OCNet [162]‡	62.58	548.5	80.1	n/a
PSPNet [176] ‡	70.43	574.9	78.4	n/a
Results w/ and w/o distillation schemes				
MD [147] ‡	14.35	64.48	n/a	67.3
MD (Enhanced) [147] ‡	14.35	64.48	n/a	71.9
ESPNet-C [92] †	0.3492	3.468	51.1	53.3
ESPNet-C (ours) †	0.3492	3.468	57.6	59.9
ESPNet [92] †	0.3635	4.422	60.3	61.4
ESPNet (ours) †	0.3635	4.422	62.0	63.8
ResNet18 (0.5) †	3.835	33.35	54.1	55.4
ResNet18 (0.5) (ours) †	3.835	33.35	61.4	62.7
ResNet18 (1.0) ‡	15.24	128.2	67.6	69.1
ResNet18 (1.0) (ours) ‡	15.24	128.2	73.1	75.3
MobileNetV2Plus [81] ‡	8.301	86.14	68.9	70.1
MobileNetV2Plus (ours) ‡	8.301	86.14	74.0	74.5

† Train from scratch
‡ Initialized from the weights pre-trained on ImageNet
§ We select the best model along with training on the validation set to submit to the leader board. All our models are tested on a single scale. Some large networks are tested on multiple scales, such as OCNet and PSPNet.

of ENet. Again we can see that the distillation scheme improves the performance. Figure 3.9 shows some samples on the CamVid test set w/o and w/ the distillation produced from ESPNet.

We also conduct an experiment by using an extra unlabeled dataset, which contains 2000 unlabeled street scene images collected from the Cityscapes dataset, to show that the distillation schemes can transfer the knowledge of the unlabeled images. The experiments are done with ESPNet and ESPNet-C. The loss function is almost the same except that there is no cross-entropy loss over the unlabeled dataset. The results are shown in Figure 3.10. We can see that our distillation method with the extra unlabeled data can significantly improve mIoU of ESPNet-c and ESPNet for 13.5% and 12.6%.

ADE20K. The ADE20K dataset is very challenging and contains 150 categories of objects. The frequency of objects appearing in scenes and the pixel ratios of different objects follow a long-tail distribution. For example, the stuff classes like "wall", "building", "floor", and "sky" occupy more than 40% of all the annotated pixels, and

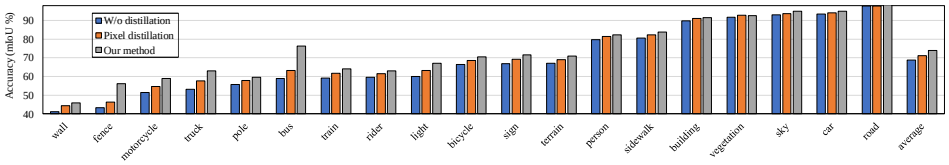


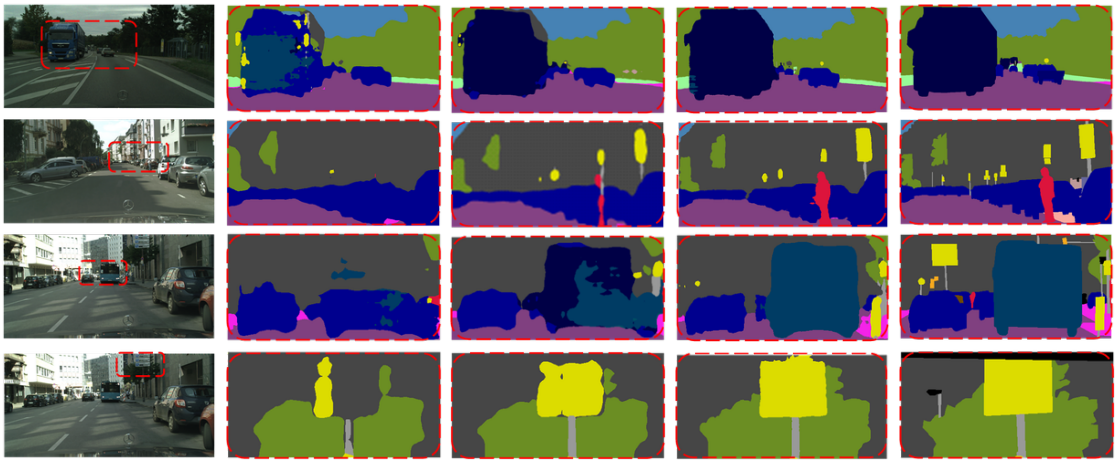
FIGURE 3.7. Illustrations of the effectiveness of pixel-wise and structured distillation schemes in terms of class IoU scores on the network MobileNetV2Plus [81] over the Cityscapes test set. Both pixel-level and structured distillation help improve the performance especially for the hard classes with low IoU scores. The improvement from structured distillation is more significant for structured objects, such as the "bus" and "truck".

TABLE 3.9. The segmentation performance on the test set of CamVid. ImN = ImageNet dataset, and unl = unlabeled street scene dataset sampled from Cityscapes.

Method	Extra data	mIoU (%)	#Params (M)
ENet[106]	no	51.3	0.3580
FC-DenseNet56[28]	no	58.9	1.550
SegNet[8]	ImN	55.6	29.46
DeepLab-LFOV[20]	ImN	61.6	37.32
FCN-8s[122]	ImN	57.0	134.5
ESPNet-C[92]	no	56.7	
ESPNet-C (ours)	no	60.3	0.3492
ESPNet-C (ours)	unl	64.1	
ESPNet[92]	no	57.8	
ESPNet (ours)	no	61.4	0.3635
ESPNet (ours)	unl	65.1	
ResNet18	ImN	70.3	
ResNet18 (ours)	ImN	71.0	15.24
ResNet18 (ours)	ImN+unl	72.3	

the discrete objects, such as vase and microwave at the tail of the distribution, occupy only 0.03% of the annotated pixels.

We report the results for ResNet18 and the MobileNetV2 which are trained with the initial weights pre-trained on the ImageNet dataset, and ESPNet which is trained from scratch in Table 3.10. We follow the same training scheme in [145]. All the results are tested on a single scale. For ESPNet, with our distillation, we can see that the mIoU score is improved by 3.78%, and it achieves higher accuracy with fewer parameters compared to SegNet. For ResNet18 and MobileNetV2, after the distillation, we achieve 2.73% improvement over the one without distillation reported in [145].



(A) (B) (C) (D) (E)

FIGURE 3.8. Qualitative results on the Cityscapes testing set produced from MobileNetV2Plus: (A) initial images, (B) w/o distillation, (C) only w/ pixel-wise distillation, (D) Our distillation schemes: both pixel-wise and structured distillation schemes. (E) Ground truth labels. The segmentation map in the red box about four structured objects: "trunk", "person", "bus" and "traffic sign" are zoomed in. One can see that the structured distillation method (ours) produces more consistent labels.

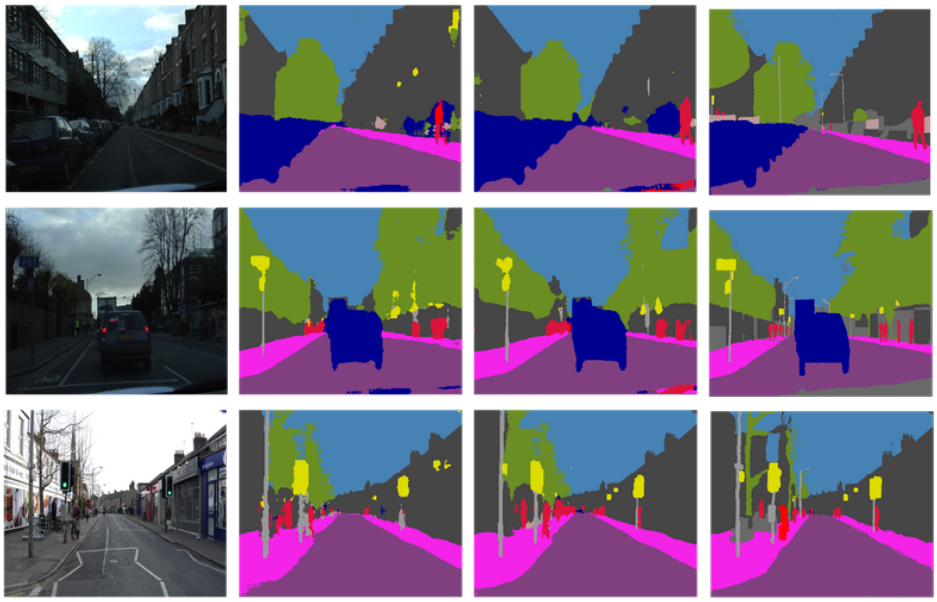
TABLE 3.10. The mIoU and pixel accuracy on the validation set of ADE20K.

Method	mIoU(%)	Pixel Acc. (%)	#Params (M)
SegNet [8]	21.64	71.00	29.46
DilatedNet50 [145]	34.28	76.35	62.74
PSPNet (teacher) [176]	42.19	80.59	70.43
FCN [122]	29.39	71.32	134.5
ESPNet [92]	20.13	70.54	0.3635
ESPNet (ours)	24.29	72.86	0.3635
MobileNetV2 [145]	34.84	75.75	2.17
MobileNetV2 (ours)	38.58	79.78	2.17
ResNet18 [145]	33.82	76.05	12.25
ResNet18 (ours)	36.60	77.97	12.25

3.4.2 Depth Estimation

Implementation Details

Network structures We use the same model described in [143] with the ResNext101 backbone as our teacher model, and replace the backbone with MobileNetV2 as the compact model. **Training details** We train the student net using the crop size 385×385 by mini-batch stochastic gradient descent (SGD) with batchsize of 12. The initialized learning rate is 0.001 and is multiplied by $(1 - \frac{iter}{max-iter})^{0.9}$. For both w/



(A) (B) (C) (D)

FIGURE 3.9. Qualitative results on the CamVid test set produced from ESPNet. (A) Image. (B) Baseline student network trained without distillation. (C) Our method. (D) Ground truth.

and w/o distillation methods, the training epoch is 200.

Dataset

NYUD-V2. The **NYUD-V2** dataset contains 1449 annotated indoor images, in which 795 images are for training and others are for testing. The image size is 640 × 480. Some methods have sampled more images from the video sequence of NYUD-v2 to form a **Large-NYUD-v2** to further improve the performance. Following [143], we do ablation studies on the small dataset and also apply the distillation method on current state-of-the-art real-time depth models trained with Large-NYUD-v2 to verify the effectiveness of the structured knowledge distillation.

Method	#Params (M)	**rel**	**log10**	**rms**	$\boldsymbol{\delta_1}$	$\boldsymbol{\delta_2}$	$\boldsymbol{\delta_3}$
		Lower is better			Higher is better		
Laina et al. [64]	60.62	0.127	0.055	0.573	0.811	0.953	0.988
DORN [36]	105.17	0.115	0.051	0.509	0.828	0.965	0.992
AOB [53]	149.82	0.115	0.050	0.530	0.866	0.975	0.993
VNL (teacher) [143]	86.24	**0.108**	**0.048**	**0.416**	**0.875**	**0.976**	**0.994**
CReaM [125]	**1.5**	0.190	-	0.687	0.704	0.917	0.977
RF-LW [100]	3.0	0.149	-	0.565	0.790	0.955	0.990
VNL	2.7	0.135	0.060	0.576	0.813	0.958	0.991
VNL w/ dis	2.7	**0.130**	**0.055**	**0.544**	**0.838**	**0.971**	**0.994**

TABLE 3.11. Depth estimation results and model parameters on NYUD-v2 test dataset. With the structured knowledge distillation, the performance is improved over all evaluation metrics.

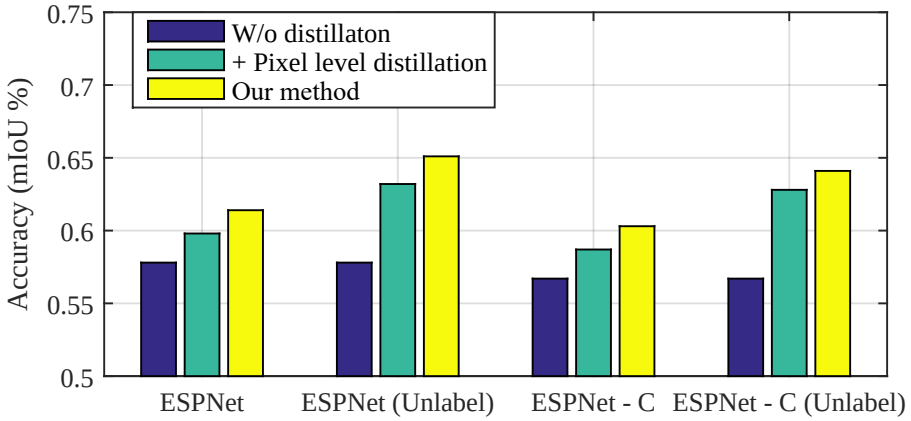


FIGURE 3.10. The effect of structured distillation on CamVid. This figure shows that distillation can improve the results in two cases: trained over only the labeled data and over both the labeled and extra unlabeled data.

Evaluation Metrics

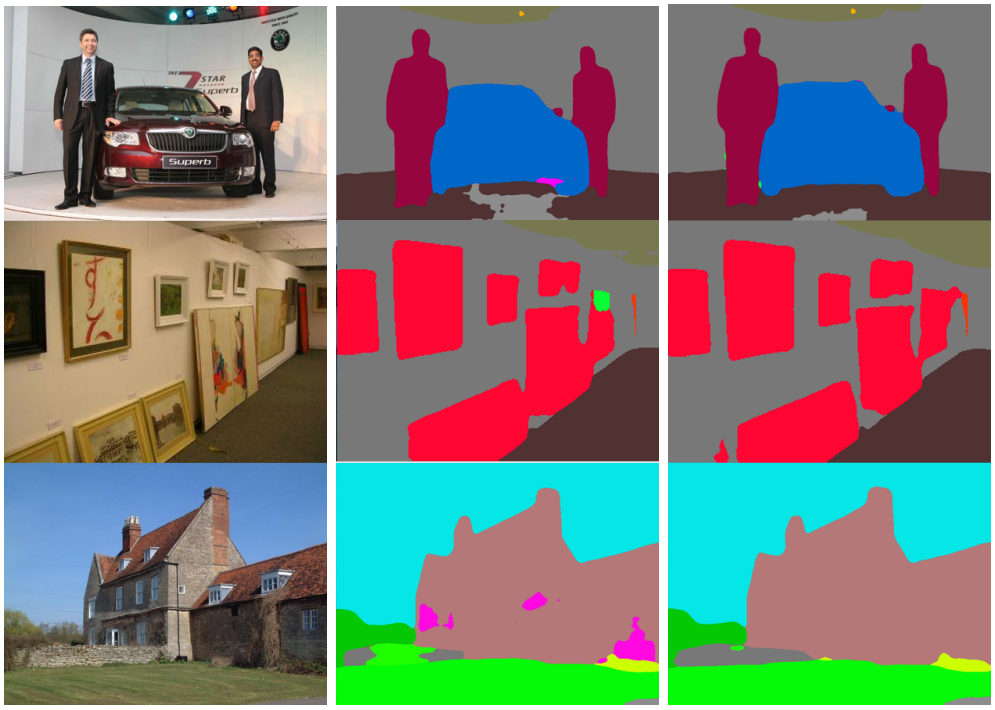
We follow previous methods [143] to evaluate the performance of monocular depth estimation quantitatively based on following metrics: mean absolute relative error (rel), mean log_{10} error (log_{10}), root mean squared error (rms) , and the accuracy under threshold ($\delta_i < 1.25^i, i = 1, 2, 3$).

Results

Ablation studies We compare the pixel-wise distillation and the structured knowledge distillation in this section. In the dense-classification problem, *e.g.*, semantic segmentation, the output logits of the teacher is a soft distribution of all the classes, which contains the relations among different classes. Therefore, directly transfer the logits from teacher models from the compact ones at the pixel level can help improve the performance. Different from semantic segmentation, as the depth map is real values, the output of the teacher is often not as accurate as of the ground truth. In the experiments, we find that adding pixel-level distillation hardly improves the accuracy in the depth estimation task. Thus, we only use structured distillation in the depth estimation task.

To verify that the distillation method can further improve the accuracy with unlabeled data, we use $30K$ images sampled from the video sequence of NYUD-v2 without the depth map. The results are shown in Table 3.12. We can see that structured knowledge distillation performs better than pixel-wise distillation, and adding extra unlabelled data can further improve the accuracy.

Comparison with state-of-the-art methods We apply the distillation method to a few state-of-the-art lightweight models for depth estimation. Following [143], we train the student net on Large-NYUD-v2 with the same constraints in [143] as our



(A) (B) (C)

FIGURE 3.11. Qualitative results on ADE20K produced from MobileNetV2. (A) Image. (B) Baseline student network trained without distillation. (C) Our method.

Method	Baseline	PI	+PA	+PA +HO	+PA +HO +Unl
rel	0.181	0.183	0.175	0.173	**0.160**

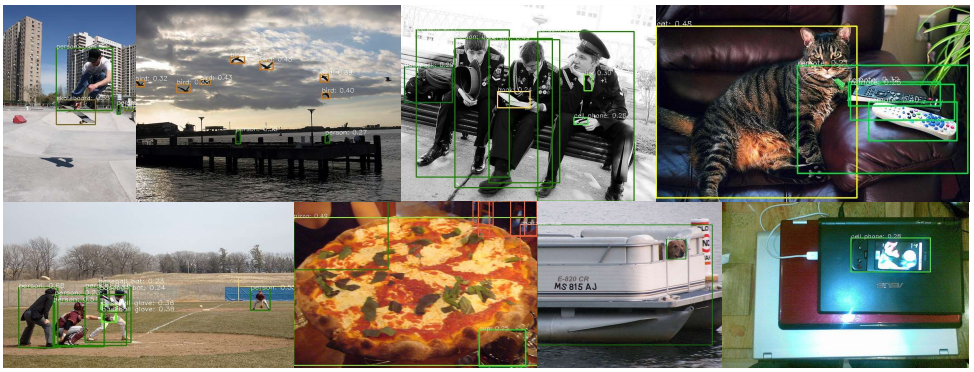
TABLE 3.12. Relative error on the NYUD-V2 test dataset. 'Unl' means Unleblled data sampled from the large video sequence. The pixel-level distillation alone can not improve the accuracy. Therefore we only use structured-knowledge distillation in the depth estimation task.

baseline and achieve 13.5 in the metric 'rel'. Following the same training setups, with the structured knowledge distillation terms, we further improve the strong baseline and achieve a relative error (rel) of 13.0. In Table 3.11, we list the model parameters and accuracy of a few state-of-the-art large models along with some real-time models, indicating that the structured knowledge distillation works on a strong baseline.

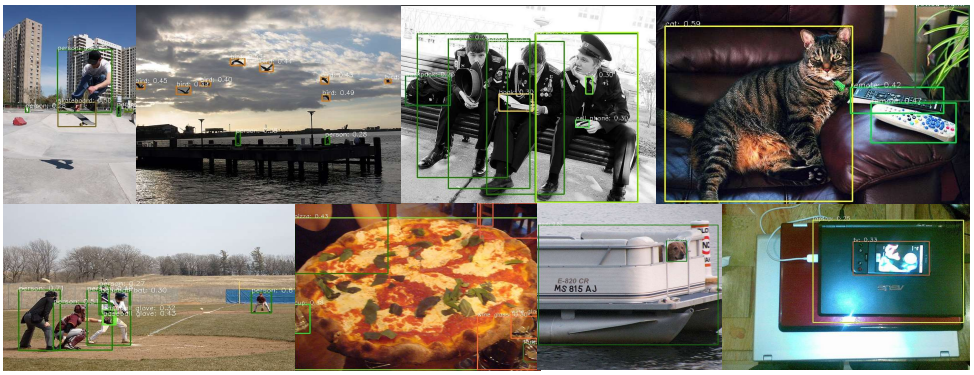
3.4.3 Object Detection

Implementation Details

Network structures We experiment with the recent one-stage architecture FCOS [133], using the backbone ResNeXt-32x8d-101-FPN as the teacher network. The channel in the detector towers is set to 256. It is a simple anchor-free model but can achieve comparable performance with state-of-the-art two-stage detection methods.



(A) Detection results w/o distillation



(B) Detection results w/ distillation

FIGURE 3.12. Detection results on the COCO dataset. With the structured knowledge distillation, the detector can improve the results with occluded, highly overlapped and extremely small objects. It can also produce a higher classification score compared to the baseline.

We choose two different models based on the MobileNetV2 backbone: c128-MNV2 and c256-MNV2 released by FCOS [133] as our student nets, where c represents the channel in the detector towers. We apply the distillation loss on all the output levels of the feature pyramid network.

Training setup We follow the training schedule in FCOS [133]. For ablation studies, all the teacher, the student w/ and w/o distillation are trained with stochastic gradient descent (SGD) for 90K iterations with the initial learning rate being 0.01 and a mini-batch of 16 images. The learning rate is reduced by a factor of 10 at iteration 60K and 80K, respectively. Weight decay and momentum are set to be 0.0001 and 0.9, respectively. To compare with other state-of-the-art real-time detectors, we double the training iterations and the batch size, and the distillation method can further improve the results on the strong baselines.

Dataset

COCO Microsoft Common Objects in Context (COCO) [77] is a large-scale detection benchmark in object detection. There are $115K$ images for training and $5K$ images for validation. We evaluate the ablation results on the validation set, and we also submit the final results to the test-dev of COCO.

Method	mAP	AP50	AP75	APs	APm	APl
Teacher	42.5	61.7	45.9	26.0	46.2	54.3
student	31.0	48.5	32.7	17.1	34.2	39.7
+MIMIC [67]	31.4	48.1	33.4	16.5	34.3	41.0
+PA	31.9	49.2	33.7	17.7	34.7	**41.3**
Ours	**32.1**	**49.5**	**34.2**	**18.5**	**35.3**	41.2

TABLE 3.13. PA vs. MIMIC on the minival split with MobileNetV2-c256 as the student net. Both distillation methods can improve the accuracy of the detector, and the structured knowledge distillation performs better than the pixel-wise MIMIC. By applying all the distillation terms, the results can be further improved.

Method	mAP	AP50	AP75	APs	APm	APl
Teacher	42.5	61.7	45.9	26.0	46.2	54.3
C128-MV2	30.9	48.5	32.7	17.1	34.2	39.7
w/ distillation	31.8	49.2	33.8	17.8	35.0	40.4
C256-MV2	33.1	51.1	35.0	18.5	36.1	43.4
w/ distillation	33.9	51.8	35.7	19.7	37.3	43.4

TABLE 3.14. Detection accuracy with and without distillation on COCO-minival.

Evaluation Metrics

Average precision (AP) computes the average precision value for recall value over 0 to 1. The mAP We also report AP50 and AP75 represent the AP with a single IoU of 0.5 and 0.75, respectively. APs, APm, and APl are AP across different scales for small, medium, and large objects.

Results

Comparison with different distillation methods To demonstrate the effectiveness of the structured knowledge distillation, we compare the pair-wise distillation method with the previous MIMIC [67] method, which aligns the feature map on pixel-level. We use the c256-MNV2 as the student net and the results are shown in Table 3.13. By adding the pixel-wise MIMIC distillation method, the detector can be improved by 0.4% in mAP. Our structured knowledge distillation method can improve by 0.9% in mAP. Under all evaluation metrics, the structured knowledge distillation method performs better than MIMIC. By combing the structured knowledge distillation with the pixel-wise distillation, the results can be further improved to the mAP of 32.1%. Comparing to the baseline method without distillation, the improvement of AP75, APs, and APl is more sound, indicating the effectiveness of the distillation method.

We show some detection results in Figure 3.12. One can see that the detector trained with the distillation method can detect more small objects such as “person” and “bird”.

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	ms/img
RetinaNet [76]	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2	198
RetinaNet [76]	ResNeXt-101-FPN	40.8	61.1	44.1	24.1	44.2	51.2	-
FCOS [133] (teacher)	ResNeXt-101-FPN	**42.7**	**62.2**	**46.1**	**26.0**	**45.6**	**52.6**	**130**
YOLOv2 [109]	DarkNet-19	21.6	44.0	19.2	5.0	22.4	35.5	**25**
SSD513 [83]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8	125
DSSD513 [35]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1	156
YOLOv3 [110]	Darknet-53	33.0	**57.9**	34.4	18.3	35.4	41.9	51
FCOS (student) [133]	MobileNetV2-FPN	31.4	49.2	33.3	17.1	33.5	38.8	45
FCOS (student) w/ dis	MobileNetV2-FPN	**34.1**	52.2	**36.4**	**19.0**	**36.2**	**42.0**	45

TABLE 3.15. Detection results and inference time on the COCO test-dev. The inference time was reported in the original papers [133, 76]. Our distillation method can improve the accuracy of a strong baseline with no extra inference time.

Results of different student nets We follow the same training steps (90K) and batch size (32) as in FCOS [133] and apply the distillation method on two different structures: C256-MV2 and C128-MV2. The results of w/ and w/o distillation are shown in Table 3.14. By applying the structured knowledge distillation combine with pixel-wise distillation, the mAP of C128-MV2 and C256-MV2 are improved by 0.9 and 0.8, respectively.

Results on the test-dev The original mAP on the validation set of C128-MV2 reported by FCOS is 30.9% with 90K iterations. We double the training iterations and train with the distillation method. The final mAP on minival is 33.9%. We submit these results to the COCO test-dev to show the position of our method, comparing against state-of-the-art methods. To make a fair comparison, we also double the training iterations without any distillation methods and obtain mAP of 32.7% on minival. The test results are in Table 3.15, and we also list the AP and inference time for some state-of-the-art one-stage detectors to show the position of the baseline and our detectors trained with the structured knowledge distillation method.

3.5 Conclusion

In this chapter, we have studied knowledge distillation for training compact dense prediction networks with the help of cumbersome/teacher networks. By considering the structure information in dense prediction problem, we have presented two structural distillation schemes: pair-wise distillation and holistic distillation. We demonstrate the effectiveness of our proposed distillation schemes on several recently-developed compact networks on three dense prediction tasks: semantic segmentation, depth estimation, and object detection. Our structured knowledge distillation methods are complementary to traditional pixel-wise distillation methods.

Chapter 4

channel-wise distillation

4.1 Introduction

Knowledge distillation (KD) has been proven to be a simple and effective tool for training compact dense prediction models in Chapter 3. The lightweight student networks are trained by extra knowledge transferred from large teacher networks. However, in Chapter 3, the proposed KD methods require a large training memory as the pair-wise distillation will rely on spatial size. Similar methods align the activation maps from the student and teacher network in the spatial domain, typically by normalizing the activation values on each spatial location and minimizing point-wise and/or pair-wise discrepancy. To be more efficient during the training process and combine more information from different sources, we propose to normalize the activation map in each channel to obtain a channel distribution. By simply minimizing the Kullback–Leibler (KL) divergence between the channel distribution of the two networks, the distillation process pays more attention to the most salient regions in each channel, which are useful for dense prediction tasks.

We conduct experiments on fundamental dense prediction tasks, including semantic segmentation and object detection. Experiments demonstrate that our simple and effective channel-wise distillation outperforms state-of-the-art distillation methods considerably, and requires less computational cost during training. In particular, we improve the RetinaNet based on ResNet50 backbone by 3.4% in terms of mAP on COCO dataset, and PSPNet based on ResNet18 backbone by 6.07% in terms of mIoU on Cityscapes dataset. Code will be available upon acceptance.

4.2 Background

As described in Chapter 3, dense prediction tasks are a group of fundamental tasks in computer vision, including semantic segmentation [177, 19], depth estimation [143] and object detection [72, 131]. These tasks require learning strong feature representations for complex scene understanding goals. Thus, the state-of-the-art models usually need high computational costs, making them hard to apply to mobile devices. Recently, compact networks designed for dense prediction tasks have drawn much attention. Moreover, effectively training lightweight networks has been studied

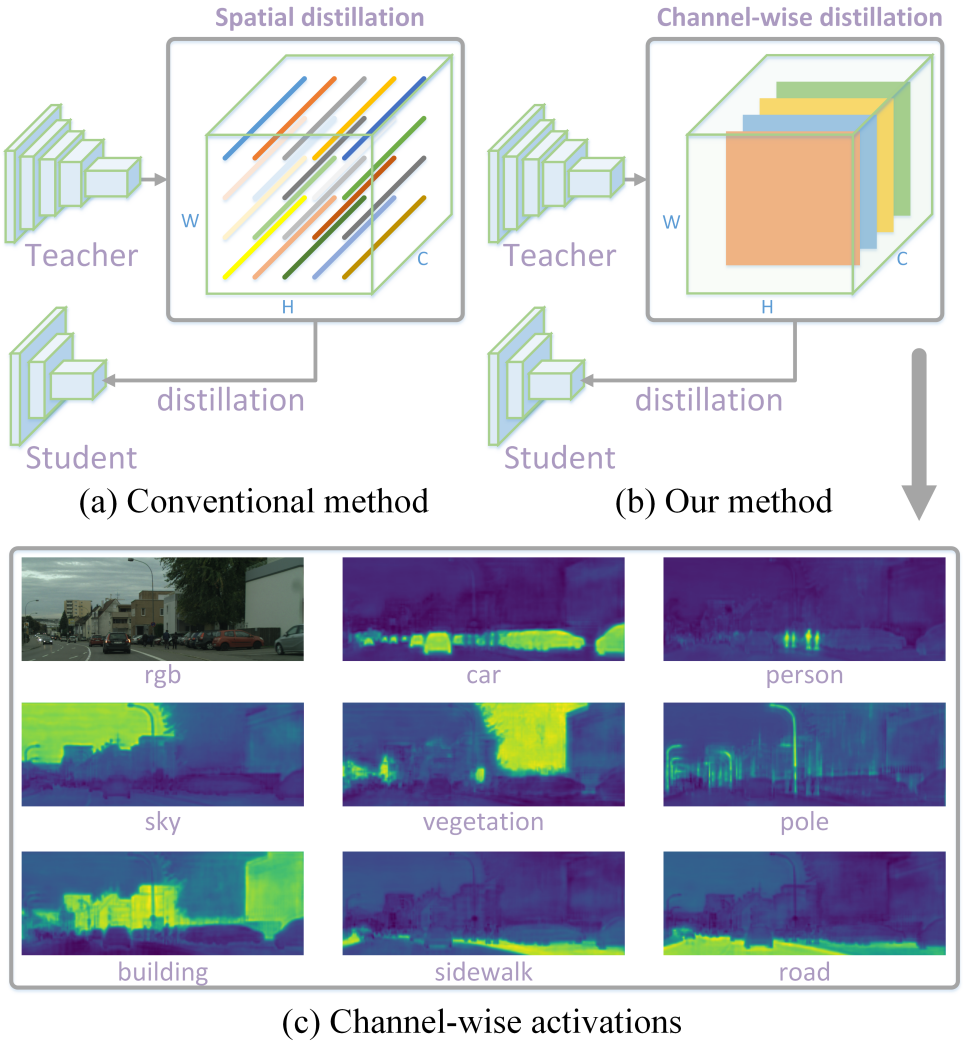


FIGURE 4.1. Spatial knowledge distillation (top-left) works by aligning feature maps in the spatial domain. Our channel-wise distillation (top-right) instead aligns each channel of the student's feature maps to that of the teacher network by minimizing the KL divergence. The bottom plot shows that the activation values of each channel tend to encode saliency of scene categories.

in previous works through knowledge distillation (KD). Dense prediction tasks are per-pixel prediction problems, which are more complex and challenging than image-level classification. Previous research [85, 68] found that directly transferring the KD methods [49, 3] in classification to semantic segmentation may not lead to satisfactory results. Strictly aligning the point-wise classification scores or the feature maps between the teacher and student network may enforce overly strict constraints and lead to sub-optimal solutions.

Our proposed structural knowledge distillation [85, 87] and the following work [51] pay attention to enforce the correlations among different spatial locations. As shown in Figure 4.1(a), the activation values [1] on each spatial location are normalized. Then,

[1] The activation values in this chapter include the final logits and the inner feature maps.

some tasks specific relationships are conducted by aggregating a sub-set of different spatial locations, such as pair-wise relations [85, 146] and inner-class relations [51]. Such methods may work better than the point-wise alignment in capturing spatial structure information and improve the performance of the student network. However, every spatial location in the activation map contributes equally to the knowledge transferring, which may bring redundant information from the teacher network.

In this chapter, we propose a novel channel distribution (CD) distillation by normalizing the activation maps in each channel for dense prediction tasks as shown in Figure 4.1(b). Then, a simple asymmetry Kullback–Leibler (KL) divergence is employed to minimize the difference between the channel distributions from the teacher and the student networks. We show an example of the channel distribution in semantic segmentation in Figure 4.1(c). The feature activations of each channel tend to encode saliency parts of scene categories. In each channel, the student will pay more attention to mimic the regions with larger activation values, which will lead to a more accurate localization in dense prediction tasks. For example, in object detection, the student network will pay more attention to learn the activation of the foreground objects and loosen the constraints in the background.

Some recent works pay attention to the knowledge contained in channels. Channel distillation [181] proposes to transfer the activation in each channel into one attention value, which is useful for image-level classification tasks but may lose too much spatial information for dense prediction tasks. Other works, like MGD [163], Channel exchanging [140] and CSC [105] verify the importance of the channel information, but they ignore the effect of normalizing the value in each channel to form the channel distribution.

Experiments are conducted on semantic segmentation and object detection. Ablation studies show that the simple normalizing operations in each channel can improve the baseline spatial distillation by a large margin. Besides, the asymmetry KL divergence also contributes to the final results. The proposed channel-wise distillation is simple and easy to transfer to different tasks and network structures. We obtain state-of-the-art performance compared to recent knowledge distillation methods on four challenging benchmarks and various network structures. We summarize our main contributions as follows.

- Unlike those existing spatial distillation approaches, we propose a novel channel-wise distillation paradigm for knowledge distillation for dense prediction tasks.
- The proposed channel-wise distillation significantly outperforms state-of-the-art KD methods for semantic segmentation and object detection, and requires less computational cost during training.
- We show consistent improvements on four benchmark datasets with various network structures on semantic segmentation and object detection tasks, demonstrating that our method is general. Given its simplicity and effectiveness, we believe that our method can serve as a strong baseline KD method for dense prediction tasks.

TABLE 4.1. Current spatial distillation methods. i and j indicate the pixel index. $D(\cdot)$ is a discriminator, and $N(i)$ indicates 8-neighborhood of pixel i. S_i is the pixel set having the same label as pixel i and $|S_i|$ stands for the size of the set S_i.

Loss	$\varphi(u, v)$	$\phi(x)$	
		Formulation	Dimensionality
Point-wise alignment			
Attention transfer [165]	L_1 or L_2	$\sum_{c=1}^{C} \|x_{ic}\|^p$	$1 \times W \times H$
Pixelwise [85, 21, 87, 142]	KL	$\text{softmax}(x_i/\tau)$	$C \times W \times H$
Pairwise or higher order alignment			
Local similarity [146]	L_1 or L_2	$\sum_{j \in N(i)} \|x_j - x_i\|$	$1 \times W \times H$
Pairwise affinity [85, 46, 87]	L_2	$\frac{x_i^T x_j}{\|x_i\|_2 \cdot \|x_j\|_2}$	$1 \times W \times H$
IFVD [142]	L_2	$cos(x_i, \sum_{j \in S_i} x_j/\|S_i\|)$	$1 \times W \times H$
Holistic [85, 87, 142]	EM [40]	$D(x_i)$	1

4.3 Method

In this section, we first present a brief introduction to spatial knowledge distillation, and then we describe our proposed channel-wise distillation.

4.3.1 Spatial Distillation

Existing KD methods often employ a point-wise alignment or align structured information among spatial locations, which can be formulated as:

$$\ell = \ell_t(y, y^S) + \alpha \cdot \varphi(\phi(y_i^T), \phi(y_i^S)). \tag{4.1}$$

Here the task loss ℓ_t is still applied with y being the ground-truth labels. The cross-entropy loss is usually employed in semantic segmentation. In object detection, the task loss usually follows the original paper, and the distillation loss is added as the regularization. α is a hyper-parameter to control the loss weight.

For better illustrating the knowledge distillation functions $\varphi(\cdot)$ and the transformation $\phi(\cdot)$ used in the literature, representative spatial distillation methods are listed in Table 4.1. Attention Transfer (AT) [165] uses an attention mask to squeeze the feature maps into a single channel for distillation. The pixel-wise loss [50] directly aligns the point-wise class probabilities. The local affinity [146] is computed by the distance between the center pixel and its 8 neighborhood pixels. The pairwise affinity [85, 46, 87] is employed to transfer the similarity between pixel pairs. The similarity between each pixel's feature and its corresponding class-wise prototype is computed to transfer the structural knowledge [142]. The holistic loss [85, 87] use the adversarial scheme to align the high-order relations between feature maps from the two networks. Note that, the last four terms consider the correlation among pixels. Existing KD methods as shown in Table 4.1 are all spatial distillation methods. All these methods consider

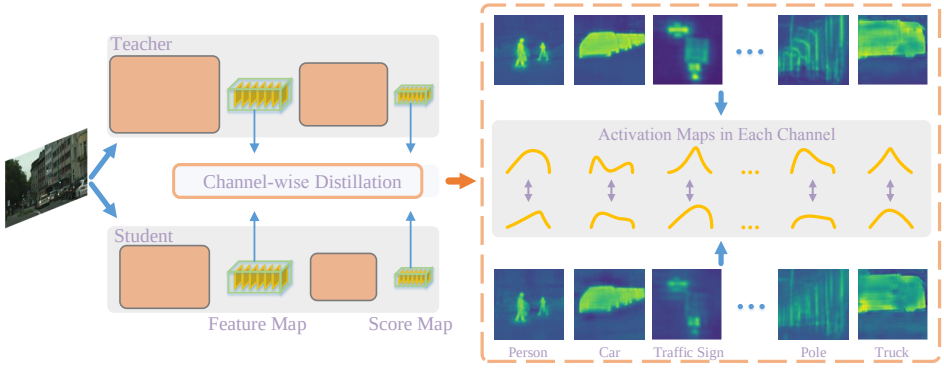


FIGURE 4.2. The overall architecture of our proposed method. The plot on the left is the paradigm of our teacher-student strategy, where the feature map and the score map can be used for channel-wise distillation. The plot on the right is the detailed description of channel-wise distillation. Activated regions correspond to scene categories.

the N channel activation values of a spatial location as the feature vectors to operate on.

4.3.2 channel-wise Distillation

To better employ the knowledge in each channel, we propose to build a channel distribution by normalizing the values in each channel. Inspired by Figure 4.1(c), the activation of different channels encodes the saliency of scene categories of an input image. Besides, a well-trained teacher network tends to produce activation maps of clearer category-specific masks for each channel—which is expected—as illustrated on the right part of Figure 4.2.

We propose a novel channel-wise distillation paradigm to guide the student to directly learn the information in channel from the well-trained teacher.

As illustrated in Figure 4.2, our method consists of the teacher network, student network, and channel-wise distillation module. The teacher and student networks are denoted as T and S, and the activation maps from T and S are $\mathbf{y}^T$ and $\mathbf{y}^S$, respectively. The channel-wise distillation loss can be formulated as:

$$\varphi(\phi(\mathbf{y}^T), \phi(\mathbf{y}^S)) = \varphi(\phi(\mathbf{y}_c^T), \phi(\mathbf{y}_c^S)). \tag{4.2}$$

ϕ is used to transfer the activation values into the channel distribution. Here we employ a softmax normalization:

$$\phi_{cd}(\mathbf{y}_c) = \frac{\exp(\frac{y_{c,i}}{\mathcal{T}})}{\sum_{i=1}^{W \cdot H} \exp(\frac{y_{c,i}}{\mathcal{T}})}, \tag{4.3}$$

where $c = 1, ..., C$. C is the number of channels and $\mathcal{T}$ is the temperature. The distribution will be softer if we use a larger $\mathcal{T}$, which means we focus on more regions in each channel. By applying the softmax normalization, we remove the influences of different magnitude scales between the large networks and the compact networks,

which will benefit the knowledge distillation as shown in the previous work [137]. A 1×1 convolution layer will be employed to upsampling the number of channels for the student network if the number of channels is different between the teacher and the student. φ evaluates the discrepancy between the channel from the teacher network and the student network. We use the KL divergence as φ which can be formulated as:

$$\varphi_{cd}(y^T, y^S) = \frac{\mathcal{T}^2}{C} \sum_{c=1}^{C} \sum_{i=1}^{W \cdot H} \cdot \phi(y_{c,i}^T) \cdot \log\Big[\frac{\phi(y_{c,i}^T)}{\phi(y_{c,i}^S)}\Big]. \tag{4.4}$$

Following [49], $\mathcal{T}^2$ is used to balance the magnitude between the soft and hard targets. The KL divergence is an asymmetric metric. From Equation (4.4), we can see if $\phi(y_{c,i}^T)$ is large, the $\phi(y_{c,i}^S)$ should be as large as $\phi(y_{c,i}^T)$ to minimize the KL divergence; otherwise, if $\phi(y_{c,i}^T)$ is very small, the KL divergence will pay less attention to minimize the $\phi(y_{c,i}^S)$. Thus, the student network tend to produce similar activation distribution in the foreground sciency parts and ignore the redundancy background.

4.4 Experiments

In this section, we first describe the implementation details and the experiment settings. Then, we compare our channel-wise distillation method with other state-of-the-art distillation methods and conduct ablation studies on semantic segmentation. Finally, we show consistent improvements in semantic segmentation and object detection with various benchmarks and student network structures.

TABLE 4.2. Comparison between computation complexity and performance on the validation set among various distillation methods. The mIoU is calculated on the Cityscapes validation set with PSPNet-R101 as the teacher network and PSPNet-R18 as the student network. The complexity depends on the shape ($h_x \times w_x \times c_x$) of the input. $\mathcal{O}(D)$ denotes the discriminator complexity. The superscript $\circledast$ means that additional channel alignment convolution is needed. All the results are the mean of three runs.

Network		Structural	Complexity	Val mIoU(%)	
				Featuremap	Scoremap
Teacher		–	–	78.56	78.56
Student		–	–	69.10	69.10
Spatial	AT [165]	×	$h_x \cdot w_x \cdot (c_x)^p$	72.37(+3.27)$^\circledast$	72.32(+3.22)
	PI [21, 142, 85, 87]	×	$h_x \cdot w_x \cdot c_x$	70.02(+0.92)$^\circledast$	71.74(+2.64)
	LOCAL [146]	√	$8h_x \cdot w_x \cdot c_x$	69.81(+0.71)	69.75(+0.65)
	PA [85, 46, 87]	√	$(h_x \cdot w_x)^2 \cdot c_x$	71.23(+2.13)	71.41(+2.31)
	IFVD [142]	√	$h_x \cdot w_x \cdot c_x \cdot n$	71.35(+2.25)	70.66(+1.56)
	HO [85, 87, 142]	√	$\mathcal{O}(D)$	–$^\circledast$	72.13(+3.03)
Channel	CD (Ours)	√	$h_x \cdot w_x \cdot c_x$	**74.87**(+5.77)$^\circledast$	**72.82**(+3.72)

4.4.1 Experimental Settings

Datasets. Three representative semantic segmentation benchmarks, *i.e.*, Cityscapes [26], ADE20K [180] and Pascal VOC [32] are considered. We also apply the proposed distillation method to object detection on MSCOCO 2017 [77], which is a large-scale dataset that contains over 120k images with 80 categories.

The Cityscapes dataset is used for semantic urban scene understanding. It contains 5,000 finely annotated images with 2,975/500/1,525 images for training/validation/testing respectively, where 30 common classes are provided and 19 classes are used for evaluation and testing. The size of each image is 2048×1024 pixels. And there are all gathered from 50 different cities. The coarsely labeled data is not used in our experiments.

The Pascal VOC dataset contains 1,464/1,449/1,456 images for training/validation/testing. It contains 20 foreground object classes and an extra background class. In addition, the dataset is augmented by extra coarse labeling, which resulting in 10,582 images for training. The training split is used for training, and the final performance is measured on the validation set across 21 classes.

The ADE20K dataset covers 150 classes of diverse scenes, where the annotation is detailed for semantic parsing. It contains 20K/2K/3K images for training, validation, and testing. In our experiments, we report the segmentation accuracy on the validation set.

Evaluation Metrics. To evaluate the performance and efficiency of our proposed channel-wise distillation method on semantic segmentation, following the previous work [51, 87], we test each strategy via the mean Intersection-over-Union (mIoU) to indicate the segmentation accuracy in all experiments under a single-scale setting. The parameter number is computed by summing the parameters in the model and the floating-point operations per second (FLOPs) are calculated with a fixed input size (512×1024). Besides, the mean class Accuracy (mAcc) is listed for Pascal VOC and ADE20K. To evaluate the performance on object detection, we report the mean Average Precision (mAP), the inference speed (FPS), and the model size (parameters) following the previous work [168].

Implementation Details. For semantic segmentation, the teacher network is PSPNet with ResNet101 (PSPNet-R101) as the backbone for all experiments. We employ several different architectures, including PSPNet [177], DeepLab [171] with the backbones of ResNet18, and MobileNetV2 as student networks to verify the effectiveness of the channel-wise distillation. In the ablation study, we analyze the effectiveness of our method based on PSPNet with ResNet18 (PSPNet-R18). Unless otherwise indicated, the training image for the student is randomly cropped into 512×512, the batch size is set to 8, and the number of the training step is 40k. We set $\mathcal{T} = 3$ and $\mathcal{W} = 3$ in all experiments. For object detection, we employ the same teacher and student networks and the training settings as in [168].

4.4.2 Comparing with Current Knowledge Distillation Methods

To verify the effectiveness of our proposed channel-wise distillation, we compare our method with current distillation methods listed below:

- Attention Transfer (AT) [165]: Sergey et al. calculate the summation of all channels on each spatial location to obtain a single channel attention map. L_2 is employed to minimize the difference between the attention map.
- Local affinity(LOCAL) [146]: For each pixel, a local similarity map is constructed, which considers the correlations between itself and its 8 neighborhood pixels. L_2 is employed to minimize the difference between the local affinity map.
- Pixel-wise distillation (PI) [85, 87, 142, 21]: KL divergence is used to align the distribution of each spatial location from two networks.
- Pair-wise distillation (PA) [85, 46, 87]. : The correlations between all pixel pairs are considered.
- Intra-class feature variation distillation (IFVD) [142]: The set of similarity between the feature of each pixel and its corresponding class-wise prototype is regarded as the intra-class feature variation to transfer the structural knowledge.
- Holistic distillation (HO)[85, 87, 142]: The holistic embeddings of feature maps are computed by a discriminator, which is used to minimize the discrepancy between high-order relations.

TABLE 4.3. The class IoU of our proposed channel-wise distillation method compared with the other two typical structural knowledge transfer methods on the validation set of Cityscape, where PSPNet-R18 (1.0) was selected as the student network. The results are from one run.

Method	mIoU	road	sidewalk	building	wall	fence	pole	traffic light	traffic sign	vegetation
PA	71.41	97.30	80.48	90.76	37.89	52.78	60.33	63.48	74.06	91.69
IFVD	71.66	97.56	81.44	**91.49**	44.45	**55.95**	62.40	66.38	76.44	91.85
CD	**75.13**	**97.64**	**81.97**	91.89	**49.44**	56.84	**62.53**	**68.73**	**77.60**	**92.20**
Class	terrain	sky	person	rider	car	truck	bus	train	motorcycle	bicycle
PA	58.60	93.48	78.96	55.45	93.42	63.79	78.48	60.12	**51.62**	74.01
IFVD	61.29	93.97	78.64	52.33	93.50	60.25	74.70	58.81	44.85	75.41
CD	**63.37**	**94.32**	**80.06**	**58.49**	**94.18**	**70.31**	**85.61**	**72.85**	52.92	**76.58**

We apply all these popular distillation methods to both the feature map and the score map. The conventional cross-entropy loss is applied in all experiments. The computational complexity and performance of spatial distillation methods are reported in Table 4.2. Given the input feature map (score map) with the size of $h_f \times w_f \times c$ ($h_s \times w_s \times n$), where $h_f(h_s) \times w_f(w_s)$ is the shape of the feature map (score map). c is the number of channels and n is the number of classes. As shown in Table 4.2, all distillation methods can improve the performance of the student network. Our

channel-wise distillation method outperforms all spatial distillation methods, it outperforms the best spatial distillation method (AT) by 2.5%. Moreover, channel-wise distillation is more efficient as it requires less computational cost than other methods during the training phase. The channel-wise distillation on the feature map works better than on the score map, which may due to that the channel number is larger on the feature map and may contain more detailed salient objects. We only employ the channel-wise distillation on the feature map in the following experiments for simplicity and efficiency.

Furthermore, we list the detailed class IoU of our method and two recent state-of-the-art methods, PA [85] and IFVD [51] in Table 4.3. These methods propose to transfer structure information in semantic segmentation. Our methods significantly improve the class accuracy of several objects, such as traffic light, terrain, wall, truck, bus, and train, indicating that the channel-wise can also transfer the structural knowledge.

4.4.3 Ablation Study

We show the effectiveness of the channel-wise distillation and discuss the choice of the hyper-parameters in semantic segmentation in this section. The baseline student model is PSPNet-R18, and the teacher model is the PSPNet-R101. All the results are evaluated on the validation set of Cityscapes.

Effectiveness of channel-wise distillation. The normalized channel distribution and asymmetric KL divergence play an important role in our distillation method. We conduct experiments with four different variants to show the effectiveness of proposed methods in Table 4.4. All the distillation methods are applied to the same feature maps as input and adopted the same training scheme as described in Section 4.4.1. 'PI' represents the pixel-level knowledge distillation, which normalizes the activation of each spatial location. 'L_2 w/o NORM' represents that we directly minimize the difference between the feature maps from two networks, which considers the difference at all locations in all channels equally. 'Bhat' is short for Bhattacharyya distance [10], which is a symmetrical distribution measurement, which aligns the discrepancy in each channel. From Table 4.4, we can see that the asymmetric KL divergence considering the normalized channel distribution discrepancy achieves the best performance. Note that as the KL divergence is asymmetric, the input of the student and teacher can not be swapped. We experiment by changing the order of the input in the KL divergence, and the training does not converge.

Visualization Results. We list the visualization results in Figure 4.3 and Figure 4.8 to intuitively demonstrate that, the channel-wise distillation method (CD) outperforms the spatial distillation strategy (attention transfer). Besides, to evaluate the effectiveness of the proposed channel-wise distillation, we visualize the activation in each channel of the student network under three paradigms, *i.e.*, original network, distilled by the attention transfer (AT) and channel-wise distillation respectively, in Figure 4.4 and Figure 4.5. We also present the visualization results in Figure 4.6 to

intuitively demonstrate that, the channel distillation method (CD) outperforms the spatial distillation strategy.

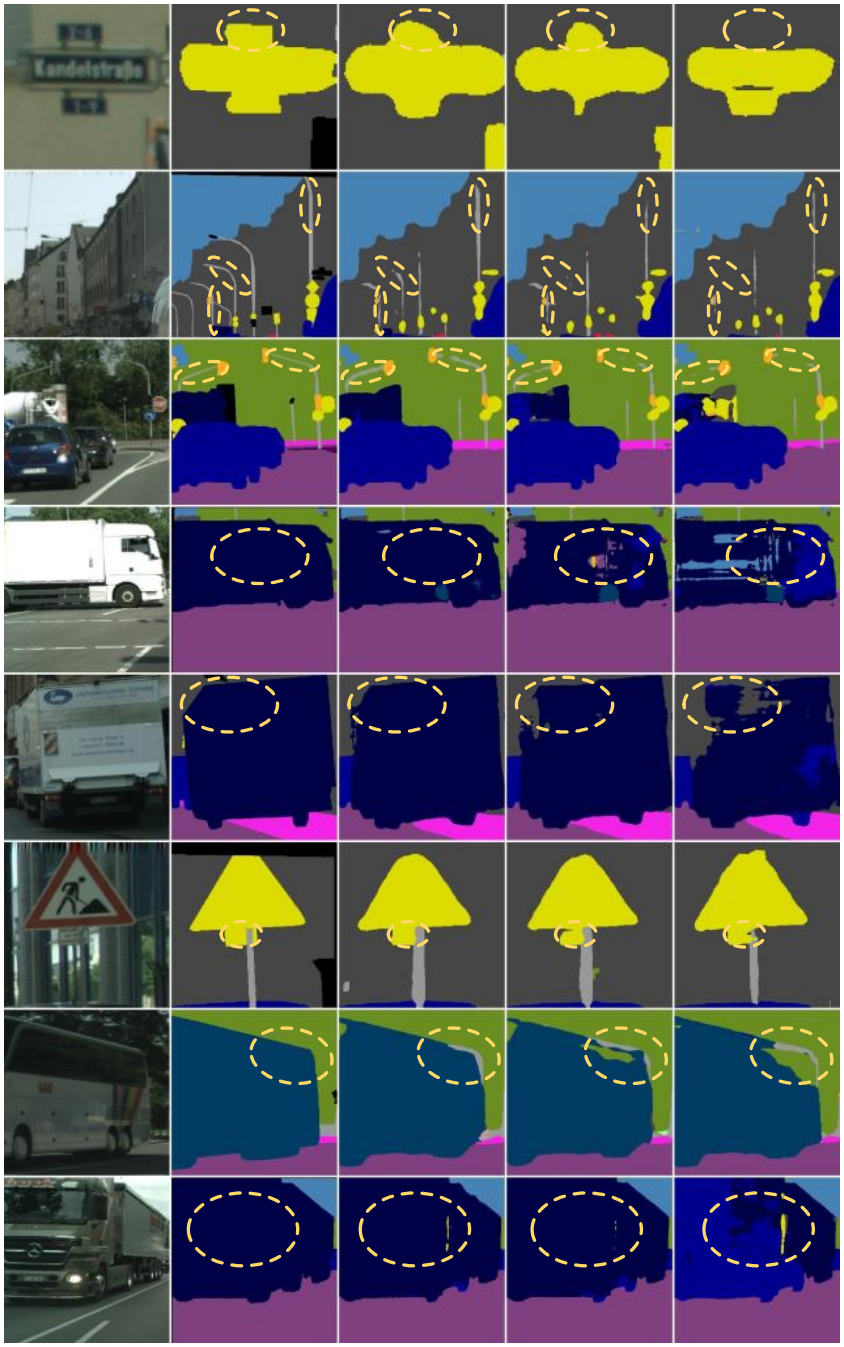


FIGURE 4.3. Qualitative segmentation results on Cityscapes produced from PSPNet-R18: (a) raw images, (b) ground truth (GT), (c) channel-wise distillation (CW), (d) the spatial distillation schemes: attention transfer (AT), and (e) output of the original student model.

Impact of the temperature parameter and loss weights. We conduct experiments to change the channel distribution by adjusting the temperature parameter

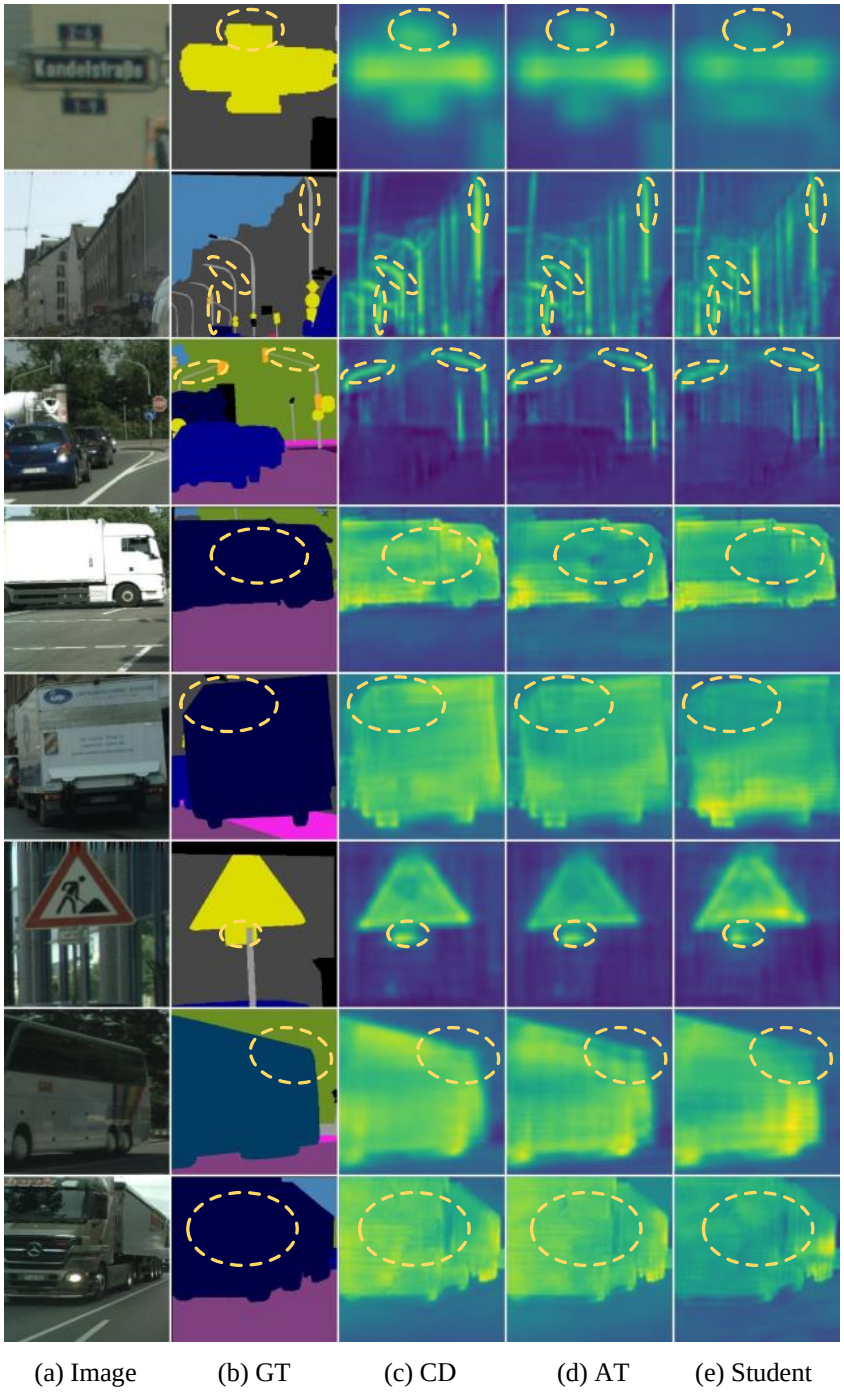


FIGURE 4.4. The channel distribution of the student under three paradigms. (a) raw images, (b) ground truth (GT), (c) channel distillation, (d) the spatial distillation schemes: attention transfer (AT), and (e) output of the original student model.

$\mathcal{T}$ under different loss weights α. All the results are the mean of three runs, which are illustrated in Figure 4.7. The loss weight is set to $1, 2, 3$, and $\mathcal{T} \in [1, 5]$. The distribution tends to be softer if we increase $\mathcal{T}$. From the figure, we can see that a soften distribution may help the knowledge distillation. Besides, in a certain range, the performance is stable. The performance will drop a lot if $\mathcal{T}$ is extremely small,

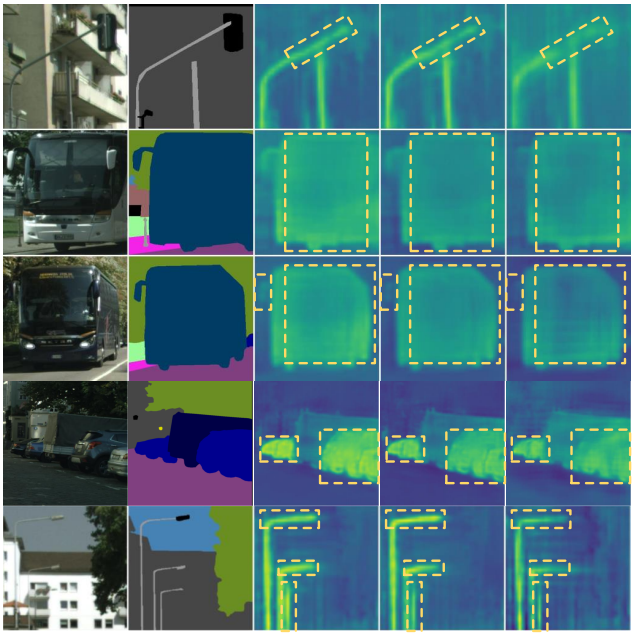


FIGURE 4.5. The channel distribution of the student under three paradigms. The yellow dotted lines show that the activation maps of CD are better than that of AT and the student network.

which means we only focus on limited salient pixels. We get the best performance when $\mathcal{T} = 4$ and $\alpha = 3$ with the PSPNet18 on the Cityscapes validation set. To avoid over-fitting and for simplicity, we choose $\mathcal{T} = \alpha = 3$ in all other experiments, which also achieves a promising result.

4.4.4 Semantic Segmentation Results

We demonstrate that our proposed method can bring consistent improvement compared with state-of-the-art semantic segmentation distillation methods, *i.e.*, structural knowledge distillation for segmentation/dense prediction (SKDS [87] /SKDD [85]) and

TABLE 4.4. Effectiveness of channel-wise distillation on semantic segmentation. We can see that with the channel normalization and the asymmetry KL divergence, the proposed channel-wise distillation achieves the best performance among other variants. All the results are the mean of three runs.

Method	Normalize	Divergence	mIoU
Teacher	-	-	78.56
Student	-	-	69.10
PI	Spatial	KL	70.02
L_2 w/o NORM	None	MSE	70.83
L_2	Channel	MSE	71.60
Bhat	Channel	Bhat	72.21
Ours	Channel	KL	**74.87**

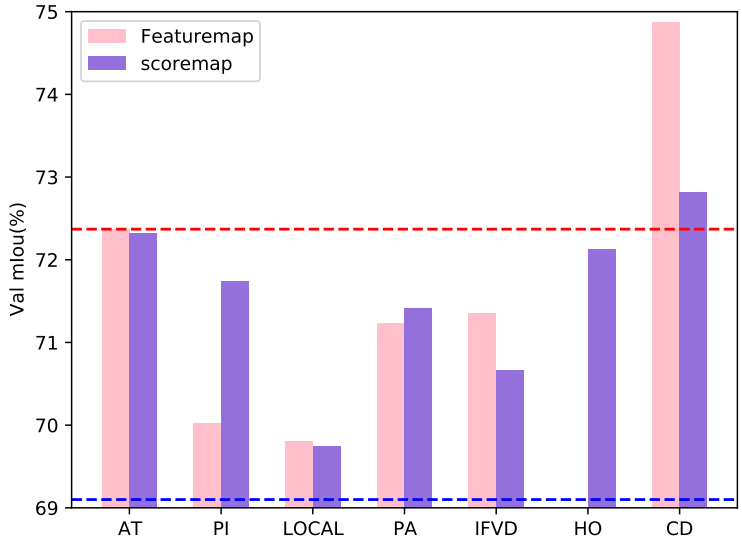


FIGURE 4.6. Illustration of the performance under different individual distillation methods. The red (blue) dotted line is the performance of AT (student). The proposed channel-wise distillation method achieves better results than any other spatial distillation method.

intra-class feature variation distillation (IFVD [142]), under various student networks. To make a fair comparison, we use the proposed channel-wise distillation for feature maps to replace the PA/IFVD on the feature maps proposed in [85, 51]. The pixel-wise distillation (PI) and the holistic distillation (HO) on the score map are also included following previous methods [85, 51].

Cityscape. We first evaluate the performance of our method on the Cityscapes dataset. Various student networks with different encoders and decoders are used to verify the effectiveness of our method. Encoders include ResNet18 (initialized with or without the weights pre-trained on ImageNet, a channel-halved variant of ResNet18 [43]) and MobileNetV2 [118], and decoders include PSPhead [177] and ASP-Phead [19]. Table 4.6 shows the results on Cityscapes.

Our method outperforms SKD and IFVD on seven student networks and three benchmarks, which further indicates that the channel-wise distillation is effective for semantic segmentation.

For the student with the same architectural type as the teacher, *i.e.*, PSPNet-R18$^\diamond$(0.5), PSPNet-R18$^\diamond$ and PSPNet-R18*, the improvements are more significant. As for the student with different architectural types with the teacher, *i.e.*, PSPNet-MBV2*, DeepLab-R18$^\diamond$(0.5), DeepLab-R18* and DeepLab-MBV2*, our method achieves consistent improvement compared with SKDS and IFVD, which proves that our channel-wise distillation is effective and can generalize well between different teacher and student networks.

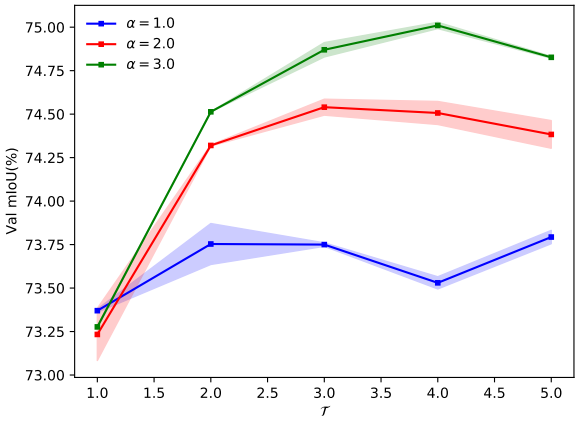


FIGURE 4.7. Impact of the temperature parameter $\mathcal{T}$ and the loss weight α.

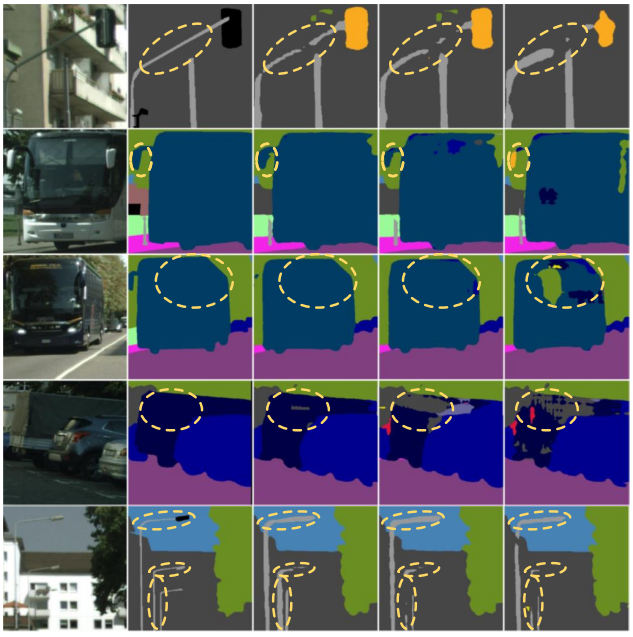


FIGURE 4.8. Qualitative segmentation results on Cityscapes produced from PSPNet-R18: (a) raw images, (b) ground truth (GT), (c) channel-wise distillation (CD), (d) the best spatial distillation schemes: attention transfer (AT), and (e) output of the original student model.

The student network of a compact model capacity (PSPNet-R18$^{\diamond}$(0.5)) shows inferior distillation performance (67.26%) compared to the student with large parameters (PSPNet-R18*) (74.87%). This may be attributed to the fact that the capability of small networks is limited compared with the teacher network and can not sufficiently absorb the knowledge of the current task. For PSPNet-R18, the student initialized

TABLE 4.5. Comparison between our methods and other distillation methods on object detection.

Model		AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	FPS	Params
Two-stage	Faster RCNN	38.4	59.0	42.0	21.5	42.1	50.3	18.1	43.57
	+Chen et al. Method [16]	38.7	59.0	42.1	22.0	41.9	51.0	18.1	43.57
	+Wang et al. Method [139]	39.1	59.8	42.8	22.2	42.9	51.1	18.1	43.57
	+Heo et al. Method [48]	38.9	60.1	42.6	21.8	42.7	50.7	18.1	43.57
	+Zhang et al. [168]	41.5	62.2	45.1	23.5	45.0	55.3	18.1	43.57
	+Our Method	41.7	62.0	45.5	23.3	45.5	55.5	18.1	43.57
One-stage	RetinaNet	37.4	56.7	39.6	20.0	40.7	49.7	20.0	36.19
	+Heo et al. [48]	37.8	58.3	41.1	21.6	41.2	48.3	20.0	36.19
	+Zhang et al. [168]	39.6	58.8	42.1	22.7	43.3	52.5	20.0	36.19
	+Our Method	40.8	60.4	43.4	22.7	44.5	55.3	20.0	36.19
Anchor free	RepPoints	38.6	59.6	41.6	22.5	42.2	50.4	18.2	36.62
	+Zhang et al. [168]	40.6	61.7	43.8	23.4	44.6	53.0	18.2	36.62
	+Our Method	42.0	63.0	45.3	24.1	46.1	55.0	18.2	36.62

by the weight pre-trained on ImageNet obtains the best distillation performance (improved from 70.04% to 74.87%), further demonstrating that the well-initialized parameters help the distillation. Thus, the better student net leads to better distillation performance, but the improvement is smaller as the gap between the teacher and student network is smaller.

To further demonstrate the effectiveness of the proposed channel-wise distillation, we only employ the proposed CD on the feature maps as our final results on Pascal VOC and ADE20K. The experiment results are reported in Table 4.7 and Table 4.8. Multi student-network variants with different encoders and decoders are used to validate the efficiency of our method in this chapter. Encoders include ResNet18 and MobileNetV2, and decoders include PSP-head and ASPP-head.

Pascal VOC. We evaluate the performance of our method on the Pascal VOC dataset. The distillation results are listed in Table 4.7. Our proposed CD improves PSPNet-R18 without distillation by 3.83%, outperforms the SKDS and IFVD by 1.51% and 1.21%. Consistent improvements on other student networks with different encoders and decoders are achieved. The gains on PSPNet-MBV2 with our method is 3.55%, surpassing the SKDS and IFVD by 1.98% and 1.20%. As for DeepLab-R18, our CD improves the student from 66.81% to 69.97%, outperforming the SKDS and IFVD by 1.84% and 1.55% respectively. Besides, the performance of DeepLab-MBV2 with our distillation is increased from 50.80% to 54.62%, outperforming the SKDS and IFVD by 2.51% and 1.23% respectively.

ADE20K. We also evaluate our method on the ADE20K dataset to further demonstrate that CD works better than other structural knowledge distillation methods. The results are shown in Table 4.8. Our proposed CD improves PSPNet-R18 without distillation by 3.83%, and outperforms the SKDS and IFVD by 1.51% and 1.21% in several. Notable performance gains on other students with different encoders and decoders are

also consistently achieved, As for PSPNet-MBV2, our method achieves a superior performance of 27.97%, surpassing the student, SKDS, and IFVD by 4.82%, 3.18%, and 2.64%. The gain on DeepLab-R18 with our CD is 2.48%, outperforming the SKDS and IFVD by 1.85% and 0.84%. Finally, the performance of DeepLab-MBV2 with our channel-wise distillation is increased from 24.98% to 29.18%, outperforming the SKDS and IFVD by 3.08% and 1.93% respectively.

4.4.5 Object Detection Results

We also apply our channel-wise distillation method on the object detection task. The experiments are conducted on MS COCO2017 [77]. Various student networks under different paradigm, *i.e.*, a two-stage anchor-based method (Faster RCNN [113]), a one-stage anchor-based method (RetinaNet [72]) and anchor-free method (RepPoints [151]), are used to validate the effectiveness of our method. To make a fair comparison, we experiment on the same teacher with the same hyper-parameters as in [168]. The only modification is that the feature alignment is changed to our channel-wise distillation. The results are shown in Table 4.5. From the table, we can see that our methods achieve consistent improvements (about 3.4% mAP) on strong baseline student networks. Compared with previous state-of-the-art distillation methods [168], our simple channel-wise distillation performs better, especially with anchor-free methods. We improve the RepPoint by 3.4% while Zhang et al. improve the RepPoint by 2%. Besides, we can see that the proposed channel-wise distillation can improve AP_{75} more significantly, demonstrating that the channel-wise distillation can improve the ability of localization better.

4.5 Conclusion

In this chapter, we have summarized previous segmentation distillation methods as the spatial distillation paradigm, and a novel structural knowledge transfer strategy, *i.e.*, channel-wise distillation, is proposed. Experimental results show that the proposed channel-wise distillation method consistently outperforms almost all existing KD methods on three public benchmark datasets with various network backbones. Additionally, our experiments demonstrate the efficiency and effectiveness of our channel-wise distillation, and it can further complement the spatial distillation methods.

We hope that the proposed simple and effective channel-wise distillation can serve as a strong baseline for effectively training compact networks for many other dense prediction tasks, including object detection, instance segmentation and panoptic segmentation.

TABLE 4.6. Comparison of student variants with the state-of-the-art distillation methods on Cityscapes, where $^{\diamond}$ denotes to be trained from scratch and * indicates to be initialized by the weights pre-trained on ImageNet, and R18 (MBV2) is the abbreviation for Resnet18 (MobileNetV2).

Method	Params (M)	FLOPs (G)	mIoU (%)	
			Val	Test
ENet [2]	0.358	3.612	–	58.3
ESPNet [116]	0.363	4.422	–	60.3
ERFNet [29]	2.067	25.60	–	68.0
ICNet [173]	26.50	28.30	–	69.5
FCN [59]	134.5	333.9	–	62.7
RefineNet [75]	118.1	525.7	–	73.6
OCNet [161]	62.58	548.5	–	80.1
Results w/ and w/o distillation schemes				
T:PSPNet [177]	70.43	574.9	78.5	78.4
S:PSPNet-R18$^{\diamond}$(0.5)	3.835	31.53	55.40	54.10
+SKDS [87]	3.835	31.53	61.60	60.50
+SKDD [85]	3.835	31.53	62.35	–
+IFVD [142]	3.835	31.53	63.35	63.68
+Ours	3.835	31.53	67.26	67.33
S:PSPNet-R18$^{\diamond}$	13.07	125.8	57.50	56.00
+SKDS [87]	13.07	125.8	63.20	62.10
+SKDD [85]	13.07	125.8	64.68	–
+IFVD [142]	13.07	125.8	66.63	65.72
Ours	13.07	125.8	70.04	70.11
S:PSPNet-R18*	13.07	125.8	69.72	67.60
+SKDS [87]	13.07	125.8	72.70	71.40
+SKDD [85]	13.07	125.8	74.08	–
+IFVD [142]	13.07	125.8	74.54	72.74
+Ours	13.07	125.8	75.79	74.37
S:PSPNet-MBV2*	1.98	16.40	58.64	57.43
+SKDS [87]	1.98	16.40	61.12	60.36
+IFVD [142]	1.98	16.40	62.74	61.92
+Ours	1.98	16.40	64.37	63.12
S:DeepLab-R18$^{\diamond}$(0.5)	3.15	31.06	61.83	60.51
+SKDS [87]	3.15	31.06	62.71	61.69
+IFVD [142]	3.15	31.06	63.12	62.37
+Ours	3.15	31.06	65.60	64.33
S:DeepLab-R18*	12.62	123.9	73.37	72.39
+SKDS [87]	12.62	123.9	73.87	72.63
+IFVD [142]	12.62	123.9	74.09	72.97
+Ours	12.62	123.9	75.25	74.12
S:DeepLab-MBV2*	2.45	20.39	65.94	65.07
+SKDS [87]	2.45	20.39	66.73	65.81
+IFVD [142]	2.45	20.39	67.04	66.12
+Ours	2.45	20.39	67.92	66.87

TABLE 4.7. The mIoU and mAcc on the validation set of Pascal VOC 2012, R18 (MBV2) is the abbreviation for Resnet18 (MobileNetV2).

Method	Params	mIoU(%)	mAcc(%)
FCN [59]	134.5	69.9	78.1
DeepLabV3 [19]	87.1	77.9	85.7
PSANet [175]	78.13	77.9	86.6
GCNet [14]	68.82	77.8	85.9
ANN [185]	65.2	76.7	84.5
OCRNet [160]	70.37	80.3	87.1
Results w/ and w/o our distillation schemes			
T:PSPNet [177]	70.43	78.52	79.57
S:PSPNet-R18	13.07	65.42	80.43
+SKDS [87]	13.07	67.73	81.73
+IFDV [142]	13.07	68.04	82.25
+Ours	13.07	69.25	83.14
S:PSPNet-MBV2	1.98	62.38	77.82
+SKDS [87]	1.98	63.95	78.93
+IFDV [142]	1.98	64.73	79.81
+Ours	1.98	65.93	81.45
S:DeepLab-R18	12.62	66.81	81.14
+SKDS [87]	12.62	68.13	82.26
+IFDV [142]	12.62	68.42	82.70
+Ours	12.62	69.97	83.47
S:DeepLab-MBV2	2.45	50.80	74.24
+SKDS [87]	2.45	52.11	75.17
+IFDV [142]	2.45	53.39	76.02
+Ours	2.45	54.62	77.13

TABLE 4.8. The mIoU and mAcc on the validation set of ADE20K, R18 (MBV2) is the abbreviation for Resnet18 (MobileNetV2).

Method	Params	mIoU(%)	mAcc(%)
FCN [59]	134.5	39.91	49.62
DeepLabV3 [19]	87.1	44.99	55.81
PSANet [175]	78.13	43.74	54.09
GCNet [14]	68.82	43.68	54.28
ANN [185]	65.2	42.93	53.25
OCRNet [160]	70.37	43.70	53.74
Results w/ and w/o our distillation schemes			
T:PSPNet [177]	70.43	44.39	45.35
S:PSPNet-R18	13.07	24.65	33.66
+SKDS [87]	13.07	25.11	33.72
+IFDV [142]	13.07	25.72	33.83
+Ours	13.07	26.80	34.02
S:PSPNet-MBV2	1.98	23.15	32.93
+SKDS [87]	1.98	24.79	34.04
+IFDV [142]	1.98	25.33	35.57
+Ours	1.98	27.97	37.16
S:DeepLab-R18	12.62	24.89	33.60
+SKDS [87]	12.62	25.52	34.10
+IFDV [142]	12.62	26.53	34.79
+Ours	12.62	27.37	35.34
S:DeepLab-MBV2	2.45	24.98	35.34
+SKDS [87]	2.45	26.10	36.51
+IFDV [142]	2.45	27.25	37.23
+Ours	2.45	29.18	38.08

Chapter 5

Efficient Semantic Video Segmentation

5.1 Introduction

Although we can achieve a promising accuracy with efficiency on the benchmark dataset of images through knowledge distillation methods, the efficient networks may produce inconsistent results when tested on a video sequence. A few methods take the correlations in the video sequence into account, *e.g.*, by propagating the results to the neighboring frames using optical flow, or extracting frame representations using multi-frame information, which may lead to inaccurate results or unbalanced latency. In contrast, in this chapter, we explicitly consider the temporal consistency among frames as extra constraints during training and process each frame independently in the inference phase. Thus no computation overhead is introduced for inference. To further improve the performance of the efficient convolutional networks, new temporal knowledge distillation methods are designed. Weighing among accuracy, temporal smoothness, and efficiency, our proposed method outperforms previous keyframe based methods and corresponding baselines which are trained with each frame independently on benchmark datasets including Cityscapes and Camvid.

5.2 Background

In recent years, the development of deep learning has brought significant success to the task of image semantic segmentation [176, 132, 17] on benchmark datasets, but often with a high computational cost. This task becomes computationally more expensive when extending to video. For a few real-world applications, *e.g.*, autonomous driving and robotics, it is challenging but crucial to build a fast and accurate video semantic segmentation system.

In the previous chapters, we have developed several knowledge distillation methods for training compact networks in dense prediction tasks, such as semantic segmentation. However, directly applying the real-time models to each frame in a video sequence will produce inconsistent results. In this chapter, we discuss how to build a semantic video segmentation system by training a compact model across frames.

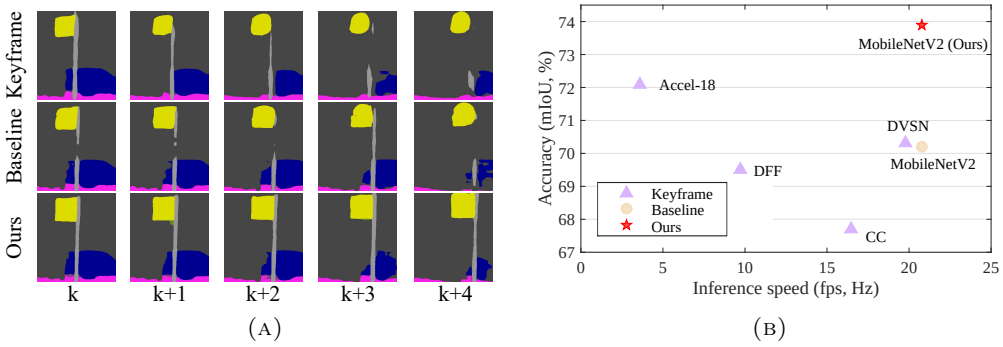


FIGURE 5.1. **(a)** Visualization results on consecutive frames: *Keyframe*: Accel18 [57] propagates and fuses the results from the keyframe (k) to non-key frames ($k+1, \dots$), which may lead to poor results on non-key frames. *Baseline*: PSPNet18 [176] trains the model on single frames. Inference on single frames separately can produce temporally inconsistent results. *Ours*: training the model with the correlations among frames and inferring on single frames separately lead to high quality and smooth results. **(b)** Comparing our enhanced MobileNetV2 model with previous keyframe based methods: Accel [57], DVSN [149], DFF [182] and CC [123]. The inference speed is evaluated on a single GTX 1080Ti.

Previous works for semantic video segmentation can be categorized into two groups. The first group focuses on improving the performance for video segmentation by performing post-processing among frames [82], or employing extra modules to use multi-frames information during inference [37]. The high computational cost makes it difficult for mobile applications. The second group uses keyframes to avoid processing of each frame, and then propagate [182, 183, 149] the outputs or the feature maps to other frames (non-key frames) using optical flows. Keyframe based methods indeed accelerate inference. However, it requires different inference time for keyframes and non-key frames, leading to an unbalanced latency, thus being not friendly for real-time processing. Moreover, accuracy cannot be guaranteed for each frame due to the cumulative warping error, for example, the first row in Figure 5.1a.

Efficient semantic segmentation methods on 2D images [92, 154, 104] have draw much attention recently. Clearly, applying compact networks to each frame of a video sequence independently may alleviate the latency and enable real-time execution. However, directly training the model on each frame independently often produces temporally inconsistent results on the video as shown in the second row of Figure 5.1a. To address the above problems, we explicitly consider the temporal consistency among frames as extra constraints during the training process and employ compact networks with per-frame inference to ease the problem of latency and achieve real-time inference.

A motion guided *temporal loss* is employed with the motivation of assigning a consistent label to the same pixel along with the time axis. A motion estimation network is introduced to predict the motion (*e.g.*, optical flow) of each pixel from the current frame to the next frame based on the input frame pair. Predicted semantic

labels are propagated to the next frame to supervise predictions of the next frame. Thus, the temporal consistency is encoded into the segmentation network through this constraint.

We design a new *temporal consistency knowledge distillation* strategy to help the training of efficient networks. Distillation methods are widely used in image recognition tasks [86, 47, 67], and achieve great success. Different from previous distillation methods, which only consider the spatial correlations, we embed the temporal consistency into distillation items. We extract the pair-wise frames dependency by calculating the pair-wise similarities for different locations between two frames, and further encode the multi-frames dependency into a latent embedding by using a recurrent unit, ConvLSTM [124]. The new distillation methods not only improve temporal consistency but also boost segmentation accuracy. We also include the spatial knowledge distillation methods (i.e. pixel-wise distillation and pair-wise distillation) described in Chapter 3 of single frames in training to further improve the accuracy.

We evaluate the proposed methods on semantic video segmentation benchmarks: Cityscapes [25], Camvid [11] and 300VW-Mask [141]. A few compact backbone networks, i.e., PSPNet18 [176], MobileNetV2 [117] and a lightweight HRNet [127], are included to verify that the proposed methods can empirically improve the segmentation accuracy and the temporal consistency, without any extra computation and post-processing during inference. The proposed methods also show superiority in the trade-off of accuracy and inference speed. For example, with the per-frame inference fashion, our enhanced MobileNetV2 [117] can achieve higher accuracy with a faster inference speed compared with state-of-the-art keyframe-based methods as shown in Figure 5.1b. We summarize our main contributions as follows.

- We process semantic video segmentation with compact models by per-frame inference, without introducing post-processing and computation overhead, enabling real-time inference without latency.
- We explicitly consider the temporal consistency in the training process by using a temporal loss and newly designed temporal consistency knowledge distillation methods.
- Empirical experiment results on Cityscapes and Camvid show that with the help of proposed training methods, the compact models outperform previous state-of-the-art semantic video segmentation methods weighing among accuracy, temporal consistency, and inference speed.

5.3 Method

In this section, we show how we exploit the temporal information during training. As shown in Figure 5.2(a), we introduce two terms: a simple temporal loss (Figure 5.2(b)) and newly designed temporal consistency knowledge distillation strategies (Figure 5.2(c) and Figure 5.2(d)). The temporal consistency of the single-frame models can be significantly improved by employing temporal loss. However, if compact

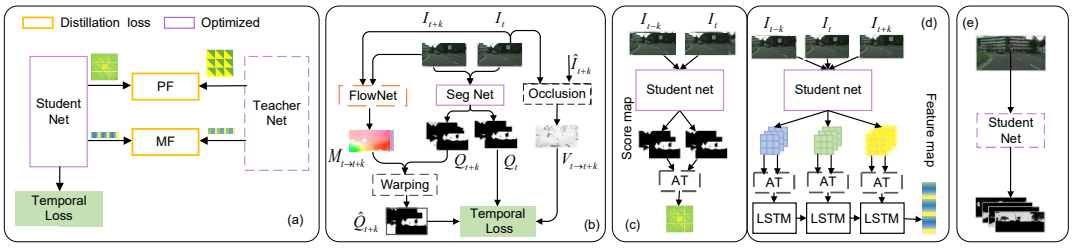


FIGURE 5.2. (a) **Overall of proposed training scheme**: We consider the temporal information by the temporal consistency knowledge distillation (c and d) and the temporal loss (b) during training. (b) **Temporal loss (TL)** encode the temporal consistency through motion constraints. Both the teacher net and the student net are enhanced by the temporal loss. (c) **Pair-wise frame dependency (PF)**: encode the motion relations between two frames. (d) **multi-frame dependency (MF)**: extract the correlations of the intermediate feature maps among multi-frames. We only show the forward pass of the student net here and apply the same operations on the teacher net to get the dependency cross frames as soft targets. (e) **The inference process**. All the proposed methods are only applied during training. We can improve the temporal consistency as well as the segmentation accuracy without any extra parameters or post-processing during inference.

models are employed for real-time execution, there is still a performance gap between large models and small ones. We design new temporal consistency knowledge distillation to transfer the temporal consistency from large models to small ones. With the help of temporal information, the segmentation accuracy can also be boosted.

5.3.1 Motion Guided Temporal Consistency

Training semantic segmentation networks independently on each frame of a video sequence often leads to undesired inconsistency. Conventional methods include previous predictions as an extra input, which introduces extra computational cost during inference. We employ previous predictions as supervised signals to assign consistent labels to each corresponding pixel along the time axis.

As shown in Figure 5.2(b), for two input frames $\mathbf{I}_t$, $\mathbf{I}_{t+k}$ from time t and $t+k$, we have:

$$\ell_{tl}(\mathbf{I}_t, \mathbf{I}_{t+k}) = \frac{1}{N} \sum_{i=1}^{N} V^{(i)}_{t \Rightarrow t+k} \left\| \mathbf{q}^i_t - \hat{\mathbf{q}}^i_{t+k \Rightarrow t} \right\|_2^2 \tag{5.1}$$

where $\mathbf{q}^i_t$ represents the predicted class probability at the position i of the segmentation map $\mathbf{Q_t}$, and $\hat{\mathbf{q}}^i_{t+k \Rightarrow t}$ is the warped class probability from frame $t+k$ to frame t, by using a motion estimation network(*e.g.*, FlowNet) $f(\cdot)$. Such an $f(\cdot)$ can predict the amount of motion changes in the x and y directions for each pixel: $f(\mathbf{I}_{t+k}, \mathbf{I}_t) = \mathbf{M}_{t \to t+k}$, where $\delta i = \mathbf{M}_{t \to t+k}(i)$, indicating the pixel on the position i of the frame t moves to the position $i + \delta i$ in the frame $t+k$. Therefore, the segmentation maps between two input frames are aligned by the motion guidance. An occlusion mask

$\mathbf{V}_{t\Rightarrow t+k}$ is designed to remove the noise caused by the warping error: $\mathbf{V}_{t\Rightarrow t+k} = \exp(-\left|\mathbf{I}_t - \hat{\mathbf{I}}_{t+k}\right|)$, where $\hat{\mathbf{I}}_{t+k}$ is the warped input frame. We employ a pre-trained optical flow prediction network as the motion estimation net in implementation. We directly consider the temporal consistency during the training process through the motion-guided temporal loss by constraining a moving pixel along the time steps to have a consistent semantic label. Similar constraints are proposed in image processing tasks [63, 152], but rarely discussed in semantic segmentation. We find that the straightforward temporal loss can improve the temporal consistency of single-frame models significantly.

5.3.2 Temporal Consistency Knowledge Distillation

Inspired by [86], we build a distillation mechanism to effectively train the compact student net S by making use of the cumbersome teacher net T. The teacher net T is already well trained with the cross-entropy loss and the temporal loss to achieve a high temporal consistency as well as the segmentation accuracy. Different from previous single frame distillation methods, two new distillation strategies are designed to transfer the temporal consistency from T to S: pair-wise-frames dependency (PF) and multi-frame dependency (MF).

Pair-wise-Frames Dependency. Following [86], we denote an attention (AT) operator to calculate the pair-wise similarity map $\mathbf{A}_{\mathbf{X}_1,\mathbf{X}_2}$ of two input tensors $\mathbf{X}_1, \mathbf{X}_2$, where $\mathbf{A}_{\mathbf{X}_1,\mathbf{X}_2} \in \mathbb{R}^{N\times N\times 1}$ and $\mathbf{X}_1, \mathbf{X}_2 \in \mathbb{R}^{N\times C}$. For the pixel a_{ij} in $\mathbf{A}$, we calculate the cosine similarity between $\mathbf{x}_1^i$ and $\mathbf{x}_2^j$ from $\mathbf{X}_1$ and $\mathbf{X}_2$, respectively: $a_{ij} = \mathbf{x}_1^{i\top}\mathbf{x}_2^j/(\|\mathbf{x}_1^i\|_2\|\mathbf{x}_2^j\|_2)$. It is an efficient and easy way to encode the correlations between two input tensors.

As shown in Figure 5.2(c), we encode the pair-wise dependency between the prediction of every two neighboring frame pairs by using the AT operator and get the similarity map $\mathbf{A}_{\mathbf{Q}_t,\mathbf{Q}_{t+k}}$, where $\mathbf{Q}_t$ is the segmentation map of frame t and a_{ij} of $\mathbf{A}_{\mathbf{Q}_t,\mathbf{Q}_{t+k}}$ denotes the similarity between the class probabilities on the location i of the frame t and the location j of the frame $t+k$. If a pixel on the location i of frame t moves to location j of frame $t+k$, the similarity a_{ij} may be higher. Therefore, the pair-wise dependency can reflect the motion correlation between two frames.

We align the pair-wise-frame (PF) dependency between the teacher net T and the student net S,

$$\ell_{PF}(\mathbf{Q_t}, \mathbf{Q_{t+k}}) = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}(a_{ij}^{\mathsf{S}} - a_{ij}^{\mathsf{T}})^2, \tag{5.2}$$

where $\forall a_{ij}^{\mathsf{S}} \in \mathbf{A}_{\mathbf{Q}_t,\mathbf{Q}_{t+k}}^{\mathsf{S}}$ and $\forall a_{ij}^{\mathsf{T}} \in \mathbf{A}_{\mathbf{Q}_t,\mathbf{Q}_{t+k}}^{\mathsf{T}}$.

Multi-Frame Dependency. As shown in Figure 5.2(d), for a video sequence $\mathcal{I} = \{\dots\mathbf{I}_{t-1}, \mathbf{I}_t, \mathbf{I}_{t+1}\dots\}$, the corresponding feature maps $\mathcal{F} = \{\dots\mathbf{F}_{t-1}, \mathbf{F}_t, \mathbf{F}_{t+1}\dots\}$ are extracted from the output of the last convolutional block before the classification layer. Then, the self-similarity map, $\mathbf{A}_{\mathbf{F}_t,\mathbf{F}_t}$, for each frame are calculated by using

AT operator in order to: 1) capture the structure information among pixels, and 2) align the different feature channels between the teacher net and student net.

We employ a ConvLSTM unit to encode the sequence of self-similarity maps into an embedding $\mathbf{E} \in \mathbb{R}^{1\times D_e}$, where D_e is the length of the embedding space. For each time step, the ConvLSTM unit takes $\mathbf{A}_{\mathbf{F}_t,\mathbf{F}_t}$ and the hidden state which contains the information of previous $t-1$ frames as input and gives an output embedding $\mathbf{E}_t$ along with the hidden state of the current time step. We align the final output embedding at the last time step, $\mathbf{E}^{\mathsf{T}}$ and $\mathbf{E}^{\mathsf{S}}$ from T and S, respectively. The output embedding encodes the relations of the whole input sequence, named multi-frame dependency (MF). The distillation loss based on multi-frame dependency is termed as: $\ell_{MF}(\mathcal{F}) = \left\|\mathbf{E}^{\mathsf{T}} - \mathbf{E}^{\mathsf{S}}\right\|_2^2$.

The parameters in the ConvLSTM unit are optimized together with the student net. To extract the multi-frame dependency, both the teacher net and the student net share the weight of the ConvLSTM unit. Note that there exists a model collapse point when the weights and bias in the ConvLSTM are all equal to zero. We clip the weights of ConvLSTM between a certain range and enlarges the $\mathbf{E}^{\mathsf{T}}$ as a regularization to prevent the model collapse. As the $\mathbf{E}^{\mathsf{T}}$ is maximized during the training as the regularization, the weight of the ConvLSTM unit could not be zero. Under this constraints, we minimize the $\left\|\mathbf{E}^{\mathsf{T}} - \mathbf{E}^{\mathsf{S}}\right\|_2^2$ to enable that the multi-frame dependency among the student network is same as that among the teacher network.

5.3.3 Optimization

We pre-train the teacher net with the segmentation loss and the temporal loss to attain a segmentation network with high semantic accuracy and temporal consistency. When optimizing the student net, we fix the weight of the motion estimation net (FlowNet) and the teacher net. These two parts are only used to calculate the temporal loss and the distillation terms, which can be seen as extra regularization terms during the training of the student net. During training, we also employ conventional cross-entropy loss, and the single frame distillation method (SF) proposed in Chapter 3 on every single frame to improve the segmentation accuracy. The whole objective function for a sampled video sequence consists of the conventional cross-entropy loss ℓ_{ce}, the single-frame distillation loss ℓ_{SF}, temporal loss, and the temporal consistency distillation terms:

$$\ell = \sum_{t=1}^{T'} \ell_{ce}^{(t)} + \lambda\Big(\sum_{t=1}^{T} \ell_{SF}^{(t)} + \sum_{i=1}^{T-1} \ell_{tl}(\mathbf{Q}_t, \mathbf{Q}_{t+1}) + \sum_{i=1}^{T-1} \ell_{PF}(\mathbf{Q}_t, \mathbf{Q}_{t+1}) + \ell_{MF}\Big), \tag{5.3}$$

where T is the number of all the frames in one training sequence, and T' is the number of labeled frames. Due to the high labeling cost in semantic video segmentation tasks [25, 11], most of the datasets are only annotated with sparse frames. Our methods can be easily applied to the sparse-labeled dataset, because 1) we can make use of large teacher models to generate soft targets; and 2) we care about the temporal

consistency between two frames, which can be self-supervised through motion. The loss weight for all regularization terms λ is set to 0.1.

After training the compact network, all the motion-estimation net, teacher net, and distillation modules can be removed. We only keep the student net as the semantic video segmentation network. Thus, both the segmentation accuracy and the temporal consistency can be improved with no extra computational cost in the per-frame inference process.

5.3.4 Implementation Details

Dataset. We evaluate our proposed method on Camvid [11] and Cityscapes [25], which are standard benchmarks for semantic video segmentation [57, 123, 103].
Network structures. Different from the keyframe based method, which takes several frames as input during inferring, we apply our training methods to a compact segmentation model with per-frame inference. There are three main parts while training the system:

- A light-weight segmentation network. We conduct most of the experiments on ResNet18 with the architecture of PSPnet [176], namely PSPNet18. We also employ MobileNetV2 [117] and a light-weight HRNet-w18 [127] to verify the effectiveness and generalization ability.
- A motion estimation network. We use a pre-trained FlowNetV2 [108] to predict the motion between two frames. Because this module can be removed during inferring, we do not need to consider employing a lightweight flownet for acceleration, like in DFF [182] and GRFP [103].
- A teacher network. We adopt widely-used segmentation architecture PSPNet [176] with a ResNet101 [42] as the teacher network, namely PSPNet101, which is used to calculate the soft targets in distillation items. We train the teacher net with the temporal loss to enhance the temporal consistency of the teacher.

Random sampled policy. In order to reduce the computational cost while training video data, and make use of more unlabeled frames, we randomly sample frames in front of the labelled frame, named 'frame_f' and behind of the labelled frame, named 'frame_b' to form a training triplet (frame_f, labelled frame, frame_b), instead of only using the frames right next to the labelled ones. The random sampled policy can take both long term and short term correlations into consideration and achieve better performance. Training on a longer sequence may show better performance with more expensive computation.
Evaluation metrics. We evaluate our method on three aspects: accuracy, temporal consistency, and efficiency. The accuracy is evaluated by widely-used mean Intersection over Union (mIoU) and pixel accuracy for semantic segmentation [86]. We report the model parameters (#Param) and frames per second (fps) to show the efficiency of employed networks. We follow [63] to measure the temporal stability of a video based on the mean flow warping error between every two neighbouring frames. Different

TABLE 5.1. Accuracy and temporal consistency on Cityscapes validation set. SF: single-frame distillation methods, PF: our proposed pair-wise-frame dependency distillation method. MF: our proposed multi-frame dependency distillation method, TL: the temporal loss. The proposed distillation methods and temporal loss can improve both the temporal consistency and accuracy, and they are complementary to each other.

Scheme index	SF	PF	MF	TL	mIoU	Pixel accuracy	Temporal consistency
a					69.79	77.18	68.50
b	✓				70.85	78.41	69.20
c		✓			70.32	77.96	70.10
d			✓		70.38	77.99	69.78
e				✓	70.67	78.46	70.46
f		✓	✓		71.16	78.69	70.21
g	✓			✓	71.36	78.64	70.13
h		✓	✓	✓	71.57	78.94	70.61
i	✓	✓	✓		72.01	79.21	69.99
j	✓	✓	✓	✓	**73.06**	**80.75**	**70.56**

from [63], we use the mIoU score instead of the mean square error to evaluate the semantic segmentation results.

5.4 Experiments

5.4.1 Ablations

All the ablation experiments are conducted on the Cityscapes dataset with the PSPNet18.

TABLE 5.2. Impact of the random sample policy. RS: random sample policy, TC: temporal consistency, TL: temporal loss, Dis: distillation terms, ALL: combine TL with Dis. The proposed random sample policy can improve the accuracy and temporal consistency.

Method	RS	mIoU	TC
PSPNet18 + TL		70.04	70.21
PSPNet18 + TL	✓	70.67	70.46
PSPNet18 + Dis		71.24	69.48
PSPNet18 + Dis	✓	72.01	69.99
PSPNet18 + ALL		72.87	70.05
PSPNet18 + ALL	✓	73.06	70.56

Effectiveness of proposed methods. In this section, we verify the effectiveness of the proposed training scheme. Both the accuracy and temporal consistency are shown in Table 5.1. We build the baseline scheme *a*, which is trained on every single labeled frame. Then, we apply three distillation terms: the single-frame dependency (SF), the

pair-wise-frame dependency (PF), and multi-frame dependency (MF), separately, to get the scheme b, c and d. The temporal loss is employed in the scheme e. Compared with the baseline scheme, all the schemes can improve accuracy as well as temporal consistency. To compare scheme b with c and d, one can see that the newly designed distillation scheme across frames can improve the temporal consistency to a greater extent. From the scheme e, we can see the temporal loss is most effective for the improvement of temporal consistency. To compare scheme f with i, we can see that single frame distillation methods [86] can improve the segmentation accuracy but may harm the temporal consistency.

To further improve the performance, we combine the distillation terms with the temporal loss and achieve the mIoU of 73.06% and temporal consistency of 70.56%. We do not increase any parameters or extra computational cost with per-frame inference. Both the distillation terms and the temporal loss can be seen as regularization terms, which can help the training process. Such regularization terms introduce extra knowledge from the pre-trained teacher net and the motion estimation network. Besides, performance improvement also benefits from temporal information and unlabelled data from the video.

Impact of the random sample policy. We apply the random sample (RS) policy when training with video sequence to make use of more unlabelled images, and capture the long-term dependency. Experiment results are shown in Table 5.2. By employing the random sampled policy, both the temporal loss and distillation terms can benefit from more sufficient training data in the video sequences, and obtain an improvement on mIoU from 0.24% to 0.69% as well as the temporal consistency from 0.19% to 0.63%. We employ such a random sampled policy considering the memory cost during training.

Impact of the teacher net. The temporal loss can improve the temporal consistency of both cumbersome models and compact models. We compare the performance of the student net training with different teacher nets (i.e., with and without the proposed temporal loss) to verify that the temporal consistency can be transferred with our designed distillation term. The results are shown in Table 5.3. The temporal consistency of the teacher net (PSPNet101) can be enhanced by training with temporal loss by 1.97%. Meanwhile, the mIoU can also be improved by 0.69%. By using the

TABLE 5.3. Influence of the teacher net. TL: temporal loss. TC: temporal consistency. We use the pair-wise-frame distillation to show our design can transfer the temporal consistency from the teacher net.

Method	Teacher Model	mIoU	TC
PSPNet101	None	78.84	69.71
PSPNet101 + TL	None	79.53	71.68
PSPNet18	None	69.79	68.50
PSPNet18	PSPNet101	70.26	69.27
PSPNet18	PSPNet101 + TL	**70.32**	**70.10**

TABLE 5.4. We compare our methods with recent efficient image/video semantic segmentation networks on three aspects: accuracy (mIoU,%), smoothness (TC, %), and inference speed (fps, Hz). TL: temporal loss, ALL: all proposed terms, TC: temporal consistency, #Param: parameters of the networks.

Method	Backbone	#Params	Cityscapes			Camvid		
			mIoU	TC	fps	mIoU	TC	fps
Video-based methods: Train and infer on multi frames								
CC [123]	VGG16	-	67.7	71.2	16.5	-	-	-
DFF [182]	ResNet101	-	68.7	71.4	9.7	66.0	78.0	16.1
GRFP [103]	ResNet101	-	69.4	-	3.2	66.1	-	6.4
DVSN [149]	ResNet101	-	70.3	-	19.8	-	-	-
Accel [57]	ResNet101/18	-	72.1	70.3	3.6	66.7	76.2	7.1
Single frame methods: Train and infer on each frame independently								
PSPNet [176]	ResNet101	68.1	78.8	69.7	1.7	77.6	77.1	4.1
SKD-MV2 [86]	MobileNetV2	8.3	74.5	68.2	14.4	-	-	-
SKD-R18 [86]	ResNet18	15.2	72.7	67.6	8.0	72.3	75.4	13.3
PSPNet18 [176]	ResNet18	13.2	69.8	68.5	9.5	-	-	-
HRNet-w18 [126, 127]	HRNet	3.9	75.6	69.1	18.9	-	-	-
MobileNetV2 [117]	MobileNetV2	3.2	70.2	68.4	20.8	74.4	76.8	27.8
Ours: Train on multi frames and infer on each frame independently								
Teacher Net	ResNet101	68.1	**79.5**	**71.7**	1.7	**79.4**	**78.6**	4.1
PSPNet18+TL	ResNet18	13.2	71.1	70.0	9.5	-	-	-
PSPNet18+ALL	ResNet18	13.2	73.1	70.6	9.5	-	-	-
HRNet-w18+TL	HRNet	3.9	76.4	69.6	18.9	-	-	-
HRNet-w18+ALL	HRNet	3.9	76.6	70.1	18.9	-	-	-
MobileNetV2+TL	MobileNetV2	3.2	70.7	70.4	20.8	76.3	77.6	27.8
MobileNetV2+ALL	MobileNetV2	**3.2**	73.9	69.9	**20.8**	78.2	77.9	**27.8**

enhanced teacher net in the distillation framework, the segmentation accuracy is comparable (70.26 **vs.** 70.32), but the temporal consistency has a significant improvement (69.27 **vs.** 70.10), indicating that the proposed distillation methods can transfer the temporal consistency from the teacher net.

Discussions. We focus on improving the accuracy and temporal consistency for real-time models by making use of temporal correlations. Thus, we do not introduce extra parameters during inference. A series of work [174, 154, 104] focus on designing network structures for fast segmentation on single images and achieve promising results. They do not contradict our work. We will verify that our methods can generalize to different network structures, *e.g.* ResNet18, MobileNetV2, and HRNet in the next session. Besides, large models [176, 184] can achieve high segmentation accuracy but have low inference speed. The temporal loss is also effective when applying to large models, *e.g.*, our teacher net.

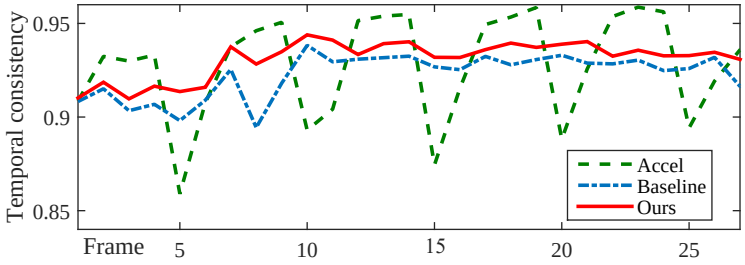


FIGURE 5.3. The temporal consistency between neighboring frames in one sampled sequence on Cityscapes. The keyframe based method Accel shows severe jitters between keyframes and others.

5.4.2 Results on Cityscapes

Comparison with single-frame-based methods. Single-frame methods are trained and inferred on each frame independently. Directly apply such methods to video sequences will produce inconsistent results. We apply our training schemes to several efficient single-frame semantic segmentation networks: PSPNet18 [176], MobileNetV2 [117] and HRNet-w18 [127, 126]. Metrics of mIoU, temporal consistency, inference speed, and model parameters are shown in Table 5.4. As Table 5.4 shows, the proposed training scheme works well with a few compact backbone networks (*e.g.*, PSPNet18, HRNet-w18, and MobileNetV2). Both temporal consistency and segmentation accuracy can be improved using the temporal information among frames.

We also compare our training methods with the single-frame distillation method [86]. According to our observation, GAN based distillation methods proposed in [86]can produce inconsistent results. For example, with the same backbone ResNet18, training with the GAN based distillation methods (SKD-R18) achieves higher mIoU: 72.7 vs. 69.8, and a lower temporal consistency: 67.6 vs. 68.5 compared with the baseline PSPNet18, which is trained with cross-entropy loss on every single frame. We replace the GAN based distillation term with our temporal consistency distillation terms and the temporal loss, denoted as "PSPNet18+ALL". Both accuracy and smoothness are improved. Note that we also employ a smaller structure of the PSPNet with half channels than in [86].

Comparison with video-based methods. Video-based methods are trained and inferred on multi frames, we list current methods including keyframe based methods: CC [123], DFF [182], DVSN [149], Accel [57] and multi-frame input method: GRFP [103] in Table 5.4. The compact networks with per-frame inference can be more efficient than video-based methods. Besides, with per-frame inference, semantic segmentation networks have no unbalanced latency and can handle every frame independently. Table 5.4 shows the proposed training schemes can achieve a better trade-off between the accuracy and the inference speed compared with other state-of-the-art semantic video segmentation methods, especially the MobileNetV2 with the fps of 20.8 and mIoU of 73.9. Although keyframe methods can achieve a high average temporal consistency score, the predictions beyond the keyframe are of low quality.

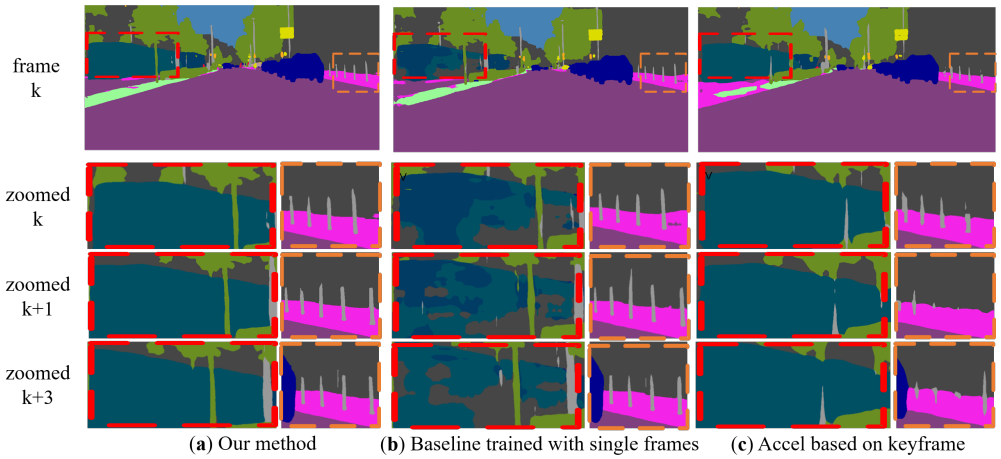


FIGURE 5.4. **Qualitative outputs.** (**a**): PSPNet18, training on multi frames and inferring on each frame. (**b**): PSPNet18, training and inferring on each frame. (**c**): Accel-18 [57], training and inferring on multiple frames. The keyframe is selected in every five frames. For better visualization, we zoom the region in the red and orange box. The proposed method can give more consistent labels to the moving train and the trees in the red box. In the orange boxes, we can see our methods have similar quantity results in each frame while the keyframe-based methods may generate worse results in the frame (*e.g.*, $k+3$) which is far from the keyframe (i.e., k).

Thus, the temporal consistency will be very low between keyframe and non-key frames, as shown in Figure 5.3. The high average temporal consistency score is mainly from the low-quality predictions on non-key frames. In contrast, our method can produce stable segmentation results on each frame.

Qualitative visualization. Qualitative visualization results are shown in Figure 5.4, in which, we can see, the keyframe-based method Accel-18 will produce unbalanced quality segmentation results between the keyframe (*e.g.*, the orange box of k) and non-key frames (*e.g.*, the orange box of $k+1$ and $k+3$), due to the different forward-networks it chooses. By contrast, ours can produce stable results on the video sequence because we use the same enhanced network on all frames. Compared with the baseline method trained on single frames, we can see our proposed method can produce more smooth results, *e.g.*, the region in red boxes. The improvement of temporal consistency is more clearly shown in the video comparison results. Moreover, we show a case of the temporal consistency between neighboring frames in a sampled frame sequence in Figure 5.3. Temporal consistency between two frames is evaluated by the warping pixel accuracy. The higher, the better. The keyframe-based method will produce jitters between keyframe and non-key frames, while our training methods can improve the temporal consistency for every frame. The temporal consistency between non-key frames is higher than our methods, but the segmentation performance is lower than ours.

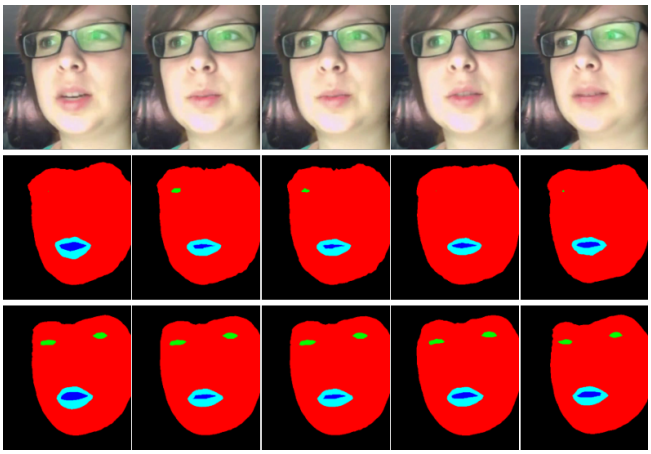


FIGURE 5.5. Visualization results on 300VW-Mask. **First row**: Input frames; **Second row**: Segmentation results from the baseline under semi-supervised settings. **Third row**: Segmentation results from ours under semi-supervised settings.

5.4.3 CamVid

We provide additional experiments on CamVid. We use MobileNetV2 as the backbone of the PSPNet. In Table 5.4, the segmentation accuracy, and the temporal consistency are improved compared with the baseline method. We also outperform current state-of-the-art semantic video segmentation methods with a better trade-off between the accuracy and the inference speed. We use the pre-trained weight from cityscapes following VideoGCRF [15], and achieve better segmentation results of 78.2 vs. 75.2. VideoGCRF [15] can achieve 22 fps with 321×321 resolution on a GTX 1080 card. We can achieve 78 fps with the same resolution. The consistent improvements on both datasets verify the value of our training schemes for real-time semantic video segmentation.

5.4.4 300VW-Mask

Implementation details. Both Cityscape and CamVid are outdoor datasets focus on the road scene. To test the generalization ability of our method, we conduct experiments with a new application, face mask segmentation, on the 300VW-Mask dataset [141]. The 300VW-Mask dataset is sampled from 300 videos in the wild, which contains 114 videos taken in unconstrained environments. The per-pixel annotations include one background class and four foreground classes: facial skin (FC), eyes, outer mouth (OMT), and inner mouth (IMT). 619/58/80 of 1-s sequences (30fps) selected from 93/9/12 videos are used for training/validation/testing, which contain 18570/1740/2400 face images in total. We train the model using the crop size of 385×385 for 80 epochs with 16 images per batch. PSPNet18 is employed as our per-frame model. The teacher net with ResNet50 is trained with temporal loss. Following the previous work [141], all the metrics are calculated on the original image resolution, and the mIoU is calculated without the background class.

Experiment settings. Different from the previous two datasets, 300VW-Mask has annotations on each frame in the 1-s sequence. Thus, we conduct experiments under two different settings. For the supervised setting, we train the per-frame model on the whole labeled dataset with and without learning from the frames. For the semi-supervised setting, we sampled 10% of the labeled data, and train the per-frame models (both the teacher and student network) on the labeled data.

Results. Experiment results are shown in Table 5.5. It is clearly shown that learning from the video sequence during training can significantly improve the accuracy and the smoothness of the per-frame model. We achieve 67.41% in terms of mIoU with only 10% of the labeled data, outperforming state-of-the-art methods with the same training splits. Our proposed method can have a larger improvement under the semi-supervised setting, demonstrating that part of the gain comes from making use of unlabeled data as analyzed before. The student network can achieve better performance as the pseudo labels are generated by test-time augmentation, which is stronger than the single-scale teacher net.

TABLE 5.5. Experiments results on 300VW-Mask [141]. The temporal stability (Tsb), temporal consistency (TC), and Insertion-over-Union (IoU) for each class are reported. mIoU is calculated without the background class. The compared methods can be referred to in [141].

Method	mIoU	FC	Eyes	OMT	IMT	BG	TC↑	Tsb↓	Backbone
Train on the Whole Labeled Dataset									
FME	63.76	90.58	57.89	62.78	43.79	94.36	-	-	ResNet50
Face tracker	60.09	88.77	50.01	61.04	40.56	97.71	-	-	-
DeeplabV2	58.66	90.55	50.19	58.58	35.31	94.38	-	-	VGG16
FCN-VGG16	55.71	91.12	44.18	58.60	28.95	94.87	-	-	VGG16
Teacher	67.61	91.04	61.44	63.5	54.45	92.29	71.31	0.008013	ResNet50
Baseline	59.04	89.71	50.82	53.23	42.38	90.06	69.72	0.008122	ResNet18
Baseline_video	67.62	91.33	64.00	62.63	52.53	92.56	71.33	0.008025	ResNet18
Train on the 10% Labeled Dataset									
Teacher	65.74	91.20	59.91	62.00	49.88	92.51	70.77	0.00827	ResNet50
Baseline	50.81	88.97	50.03	59.59	46.51	90.26	69 09	0.01098	ResNet18
Baseline_video	67.41	90.97	61.62	63.05	53.98	92.17	71.27	0.00839	ResNet18

5.5 Conclusion

In this chapter, we have developed real-time video segmentation methods that consider not only accuracy but also temporal consistency. To this end, we have proposed to use compact networks with per-frame inference. We explicitly consider the temporal correlation during training by using: the temporal loss and the new temporal consistency knowledge distillation. For inference, the model processes each frame separately, which does not introduce latency and avoids post-processing. The compact networks achieve considerably better temporal consistency and semantic accuracy, without introducing extra computational cost during inference. Our experiments have verified

the effectiveness of each component that we have designed. They can improve the performance individually and are complementary to each other.

Chapter 6

Auxiliary Overparameterization

6.1 Introduction

The large teacher models and temporal information may be hard to obtain under some cases. In this. chapter, we propose to use the idea of overparameterization for training efficient convolutional networks. It is observed that overparameterization (*i.e.*, designing neural networks whose number of parameters is larger than statistically needed to fit the training data) can improve both optimization and generalization while compact networks are more difficult to be optimized. However, overparameterization leads to slower test-time inference speed and more power consumption. To tackle this problem, we propose a novel auxiliary module to simulate the effect of overparameterization. During training, we expand the compact network with the auxiliary module to formulate a wider network to assist optimization while during inference only the original compact network is kept. Moreover, we propose to automatically search the hierarchical auxiliary structure to avoid adding supervisions heuristically. In experiments, we explore several challenging resource constraint tasks including light-weight classification, semantic segmentation, and multi-task learning with hard parameter sharing. We empirically find that the proposed auxiliary module can maintain the complexity of the compact network while significantly improving the performance.

6.2 Background

High performance of CNNs usually requires extensive computing and memory resources which makes it hard to deploy big models on resource-constrained devices. To tackle this issue, designing energy-efficient compact models for mobile devices is attracting more and more attention. Due to the limited number of parameters or complexity, directly learning simpler networks may run into worse results due to the optimization difficulty. Previous works [65, 176, 98] show that adding auxiliary losses in mid-level layers can accelerate the training process. And some advanced training strategies, such as knowledge distillation [50, 115, 164, 186, 86], propose to train efficient models with larger models. We have discussed the effectiveness of knowledge distillation in the previous chapters. However, under some cases, the complex teacher network is hard and computationally expensive to obtain. The quality of the teacher

model will greatly affect the performance of the student model, which may be the upper bound of the performance.

Interestingly, as a counterpart of the compact model, one can use overparameterization tricks to overcome the optimization difficulty. It has been observed that overparameterization for neural networks actually improves both training speed and generalization. For example, it is theoretically proved by [6] that increasing depth can improve the convergence of linear neural networks. Zhu et al.[4, 5] prove that when the network is sufficiently overparameterized, simple optimization algorithms (*e.g.*, SGD) can learn a network with a certain risk and a small generalization error in polynomial time using polynomially many samples. However, directly optimizing and deploying such a network with more parameters may not be hardware-friendly. Inspired by these observations, we, therefore, propose to utilize overparameterization to assist the optimization of compact networks during the training stage, while we discard the overparameterized portion during inference. In this way, we can learn a more accurate compact model without increasing any complexity during inference. Specifically, we propose to expand the compact network with a novel auxiliary module to formulate a wider (i.e., overparameterized) network. It is worth noting that the wider network shares the parameters of the original compact network, where the gradients of shared parameters are weighted averaged from the two networks during the training stage. In this way, we still target optimizing the compact model while improving the convergence from the additional parameters.

It is also worth noting that the proposed method is also complementary to the existing model compression approaches such as network pruning [66, 158, 89, 187]. In specific, the proposed approach is a general training strategy that works on the off-the-shelf compact models. Our method can improve the convergence of a pruned compact network during training.

Moreover, to avoid hand-crafted heuristics to explore the design space for the auxiliary module, we utilize an efficient neural architecture search (NAS) method [98] to automatically search the auxiliary module. We also suppose the employment of NAS may offer interpretability of the auxiliary network, by analyzing the tend of each operator.

In this chapter, we work on two tasks with efficient networks. On one hand, we assist the training of small models with limited capacities for a single task, including image classification and semantic segmentation. On the other hand, we propose to improve the hard parameters sharing Multi-task learning (MTL) system [9, 96, 62], which aims to solve different tasks simultaneously within only one forward pass for mobile applications. The shared backbone is difficult to optimize due to the conflict objectives of different tasks, which we treat as a case of the efficient network.

6.3 Method

In this section, we describe the auxiliary overparameterization method to train a compact model for a single task. We first give an overview of the proposed approach, then introduce the basic module design and optimization. We further introduce how to automatically search the auxiliary architecture. Note that the single task scenario can be easily generalized to the multi-task case, and the details are explored in Sec. A.

6.3.1 Overview

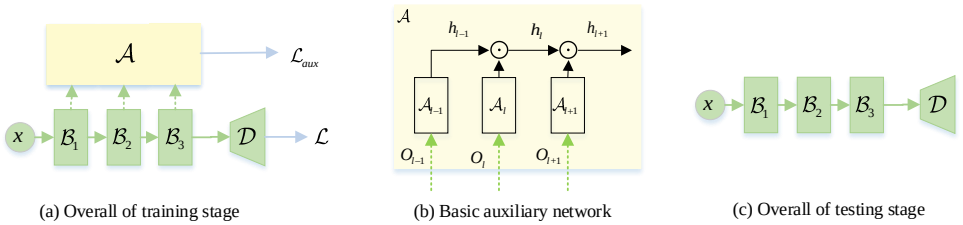


FIGURE 6.1. An overview of the proposed framework.

The overview of the framework is shown in Fig. 6.1. Let $\{\mathcal{X}, \mathcal{Y}\}$ be the training samples. The compact main network $\mathcal{A}thcalF$ consists of a backbone $\mathcal{A}thcalB$ and a decoder $\mathcal{A}thcalD$, which is shown in Fig. 6.1 (c). However, directly optimizing $\mathcal{A}thcalF$ may be difficult due to its limited number of parameters. To ease the optimization, we propose to explicitly add additional parameters during training by introducing an auxiliary module $\mathcal{A}$. In particular, the auxiliary module $\mathcal{A}$ takes the intermediate outputs of the shared parameters from $\mathcal{A}thcalB$, and provides extra gradients over different scales of the $\mathcal{A}thcalB$. In contrast, $\mathcal{A}thcalF$ does not depend on the extra parameters in $\mathcal{A}$, and benefits from the gradients and context information provided by $\mathcal{A}$. Therefore in the testing phase, only the main network $\mathcal{A}thcalF$ is used for testing while the auxiliary module $\mathcal{A}$ is removed.

6.3.2 Basic auxiliary module

In this section, we will describe a basic structure of the auxiliary module $\mathcal{A}$, which is made up of a sequential of adaptors and aggregators as shown in Fig. 6.1(b). A typical backbone framework usually consists of several blocks $\{\mathcal{A}thcalB_l\}_{l=1}^{L}$, which generate the intermediate output feature $\{\boldsymbol{O}_l\}_{l=1}^{L}$ on different scales. For the output of *l-th* block, we apply a trainable adaptor $\mathcal{A}^l(\cdot)$ to transfer feature map into a task specific space and get the adapted feature $\mathcal{A}^l(\boldsymbol{O}_l)$. We then use an operator $\odot$ to aggregate the adapted feature with the output of the previous hidden representation $\boldsymbol{h}_{l-1}$ to get $\boldsymbol{h}_l$,

$$\boldsymbol{h}_l = \boldsymbol{h}_{l-1} \odot \mathcal{A}^l(\boldsymbol{O}_l). \tag{6.1}$$

For the basic adaptor $\mathcal{A}^l(\cdot)$, we choose a simple 1×1 convolutional layer followed by a batch normalization layer. The aggregate operator we employ is the concatenation

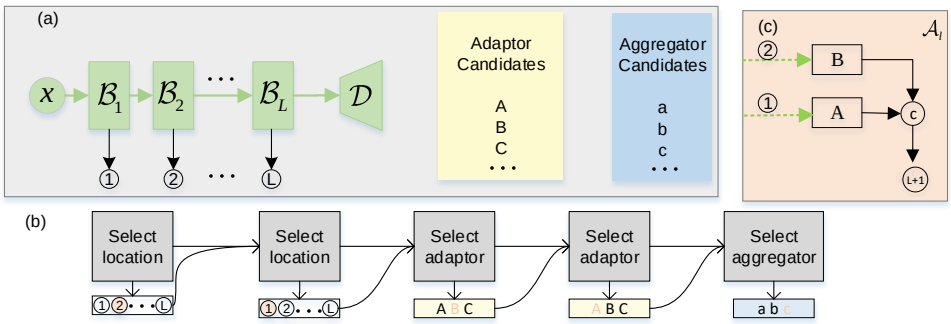


FIGURE 6.2. (a) Auxiliary module search space. (b) The controller output for generating the *l-th* auxiliary cell, $\mathcal{A}_l$. (c) An example of the sampled structure with the output of (b).

operation. It is worth noting that the hierarchical structure of $\mathcal{A}$ works like skip connections in ResNet [44] and enables the auxiliary loss to directly propagates back. Moreover, this structure of $\mathcal{A}$ and the policy on how it interacts with the main network $\mathcal{A}thcalF$ are apparently not optimal. To avoid human heuristics, we further explain the automatic design strategy.

6.3.3 Optimization

To jointly optimize the main network $\mathcal{A}thcalF$ and the auxiliary module $\mathcal{A}$, we formulate the objective function as follow,

$$\begin{aligned}\min_{\{\boldsymbol{\theta}^{\mathcal{B}},\boldsymbol{\theta}^{\mathcal{D}},\boldsymbol{\theta}a^{\mathcal{A}}\}} & \mathcal{L}(\mathcal{F}(\mathcal{X};\boldsymbol{\theta}^{\mathcal{B}},\boldsymbol{\theta}^{\mathcal{D}}),\{\mathcal{Y}\}) \\ & + \mathcal{L}_{aux}(\mathcal{A}(\boldsymbol{O}_1,\cdots,\boldsymbol{O}_L;\boldsymbol{\theta}^{\mathcal{A}},\boldsymbol{\theta}^{\mathcal{B}}),\{\mathcal{Y}\}),\end{aligned} \tag{6.2}$$

where $\boldsymbol{\theta}^{\mathcal{A}thcalB}$, $\boldsymbol{\theta}^{\mathcal{A}thcalD}$ and $\boldsymbol{\theta}^{\mathcal{A}}$ represent the parameters for the backbone $\mathcal{A}thcalB$, decoder $\mathcal{A}thcalD$ and auxiliary module $\mathcal{A}$, respectively. $\mathcal{L}$ is the task objective and $\mathcal{L}_{aux}$ is the auxiliary loss. For clarity of notation, we ignore the last output layer for the auxiliary module $\mathcal{A}$. From the equation, we can note that $\boldsymbol{\theta}^{B}$ is shared among $\mathcal{A}thcalB$ and $\mathcal{A}$, which means we explicitly utilize the overparameterization to assist the optimization over $\mathcal{A}thcalB$. Following the chain rules, the gradient of the shared parameters will have an additional term that comes from $\mathcal{L}_{aux}$, which contains extra supervisions from different scales.

6.3.4 Searching the Auxiliary Module

The basic structure of the auxiliary block is a feasible solution to achieve our goal of accelerating the training of the shared parameters and help the convergence of the network. However, the specific choices of the adaptor, aggregate operation, and the connection among different blocks still need manual design. We further utilize the neural architecture search (NAS) to explore a high-performing architecture of the auxiliary module in an automated way. The pipeline is shown in Fig. 6.2.

Fig. 6.2 (a) represents for the whole search space, including the input locations (loc), adaptor operations (op_{ad}) and the aggregate operations (op_{ag}). Two adaptors and an aggregate operation form an auxiliary cell. An LSTM-based controller will predict a sequence of operations and their locations. Each auxiliary cell is encoded by a sub-sequence of length 5. As shown in Fig. 6.2 (b), the controller firstly generates two indexes to choose nodes in loc as the input nodes, then generates indexes from op_{ad} for the adaptor operations. Finally an index in op_{ag} is sampled to aggregate together two adapted features to get an output node. The output of the auxiliary cell will be appended to loc to serve as a new choice of input locations. An example of sampled auxiliary cell $\mathcal{A}_l$ is shown in Fig. 6.2 (c). The output of the controller is a sequence of length $5 * L$, where L is the number of auxiliary cells in the auxiliary module $\mathcal{A}$. As described in the previous section, the number of auxiliary cells is equal to the number of blocks in the backbone. Since the searching space for the auxiliary module, $\mathcal{A}$ is not as large as the backbone network $\mathcal{A}thcalB$, there is no need to make a trade-off by searching a sub-network then duplicate similar to [178, 80, 12, 107]. To improve the searching accuracy, we jointly search for the whole auxiliary module.

Following the previous work of searching the structure for dense prediction problem [98], we include the following 6 operators for adaptor candidates op_{ad}: separable $conv3 \times 3$, $conv1 \times 1$, separable $conv3 \times 3$ with dilation rate 3, separable $conv3 \times 3$ with dilation rate 6, skip connections and deformable [27] $conv3 \times 3$. The aggregator candidates op_{ag} include two operations: per-pixel summation and channel-wise concatenation of two inputs. The basic structure described in the previous section is included in the designed search space.

After the controller samples a structure of auxiliary module $\mathcal{A}$, we train the sampled structure on the meta-train set and evaluate it on the meta-val set. The geometric mean of the evaluation metrics is employed as the reward. Gradient for the controller to maximize the expected reward is estimated with PPO[120]. For MTL learning, we assume there is a total of T tasks. The searching strategy for T tasks is the same as that in the single task, but the output length of the controller is expanded to $5 * L * T$. We generate auxiliary cells for each task following a heretical order, and more details can be referred to in Sec A.2 in the appendix. We choose the structure with the highest reward during the search process in the experiments.

6.4 Experiments

In this section, we first empirically evaluate the performance of the proposed auxiliary training method on light-weight image classification and semantic segmentation. Then we extend it to multi-task learning, where we investigate joint semantic segmentation and depth estimation, and then evaluate three tasks by adding a head to predict surface normal. Follow the previous works [172, 86, 99], we employ top-1 accuracy (Acc.) to evaluate the image classification, the pixel accuracy (Pixel Acc.) and mean intersection over union (mIoU) to evaluate the segmentation task, mean absolute

relative error (Rel Error) and root mean squared error (RMS Error) to evaluate the depth estimation and the mean error of angle (Mean Error) to evaluate the surface normal.

6.4.1 Experiments on Light-weight Single Tasks

TABLE 6.1. Training with/w.o auxiliary network on ImageNet classification with ResNet-18. We employ the basic auxiliary network and introduce 1×1 and 3×3 convolutions as the adaptor, respectively.

Method	Baseline	basic-conv1x1	basic-conv3x3
Accuracy, %	69.7	70.1	**70.3**

We start from training a single task on the fundamental classification task on ImageNet to verify the effectiveness of overparameterization training. We employ the ResNet-18 as our baseline network and add the basic auxiliary network with different adaptors (basic-conv1x1, basic-conv3x3), respectively. The results are shown in Table 6.1. From Table 6.1, we can see that adding an auxiliary network during training can improve the accuracy over baseline. And when increasing the complexity of the auxiliary network, the performance can be further boosted.

We also conduct an experiment on semantic segmentation with the proposed auxiliary overparamterization method on the ADE20K dataset. It contains 150 classes under diverse scenes. The dataset is divided into $20K/2K/3K$ images for training, validation, and testing respectively. To compare with previous works [86, 145], we employ a Resnet-18 as our baseline network, and follow the training settings in [86]. We compare the proposed method with some other training strategies with auxiliary supervisions. **KD**: conventional knowledge distillation method with pixel-wise supervision on the logits output. **Addition loss**: adding additional loss in the middle layers. The performance of SegNet [8], DilatedNet50 [145], PSPNet [176] and FCN [122] on ADE20K are listed as reference. We show the training accuracy curves in Figure 6.3 and the evaluation results in Table 6.2 to demonstrate the effectiveness of our method.

TABLE 6.2. Semantic segmentation results on the test set of ADE20K.

Method	mIoU (%)	Pixel Acc. (%)
SegNet	21.64	71.00
FCN	29.39	71.32
DilatedNet50	34.28	76.35
PSPNet	42.19	80.59
ResNet-18	32.17	75.32
ResNet-18 + KD	35.48	76.78
ResNet-18 + Addition loss	33.82	76.05
ResNet-18+ Base Auxi	35.78	76.99
ResNet18 + NAS Auxi	**36.13**	**77.24**

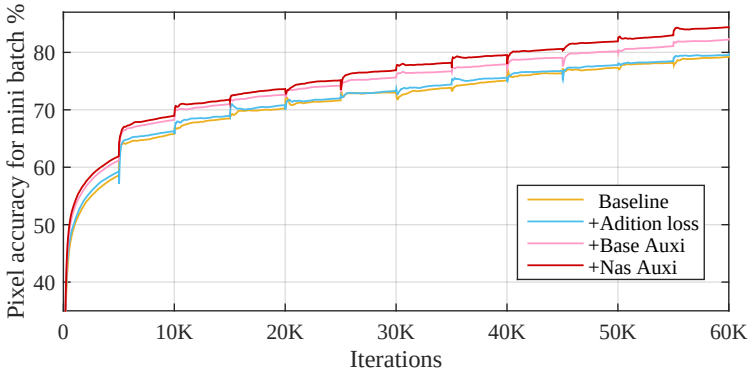


FIGURE 6.3. **Training Accuracy.** During the training stage, all the auxiliary training strategies can boost the the pixel accuracy on the training mini-batch, which indicates that the auxiliary module can improve the optimization.

From the Figure 6.3, we can see that during the training stage, adding an auxiliary module can significantly boost the training accuracy, which indicates that the auxiliary module can assist the optimization of the main network and accelerate the training. From the results in Table 6.2, we can observe that our method performs much better than adding additional losses. Employing the NAS auxiliary module can avoid the human heuristic and further improve the results. To compare with the knowledge distillation, our method achieves comparable results. However, our method is operationally easier than KD, because we do not need to pre-train a teacher network which is usually much deeper than the student network. In particular, the flexible auxiliary module is lightweight and shares the parameters of the main network which saves considerable training burden.

6.4.2 Experiments on Multi-task Learning

Network structure. We employ the general hard parameter sharing structure by sharing the hidden layers (i.e., encoder) between all tasks while keeping several task-specific output layers (i.e., decoders). We use the MobileNetV2 [117] as the shared encoder. Three variants of the main networks are as follows: i) *Baseline*: the baseline decoder is a two-layer task-specific classification module followed by an $8\times$ bilinear upsampling layer. ii) *Context*: We further add an ASPP module [18] in the shared parameters to learn a stronger shared representation. iii) *U-shape*:A U-net structure decoder are added following the design of [143]. A ResNet-50 backbone is also employed to compare with other state-of-the-art methods.

Dataset. *NYUD-v2* dataset consists of 464 different indoor scenes and has more than $100k$ raw data with depth maps, which are commonly used in the depth estimation task [143]. $1,449$ images are officially selected to further annotate with segmentation labels, in which 795 images are split for training and others are for testing. Following previous works [99, 172], we also generate coarse semantic labels using a pre-trained segmentation network [97] for $4k$ randomly sampled raw data in official training scenes

of *NYUD-v2*, which only has the depth maps. We name this dataset as *NYUD-v2-expansion*. We further conduct the experiments on *SUNRGBD* dataset, which contains 10,355 RGB-D images with semantic labels, of which 5,285 for training and 5,050 for testing. We perform the segmentation task with 13 semantic classes.

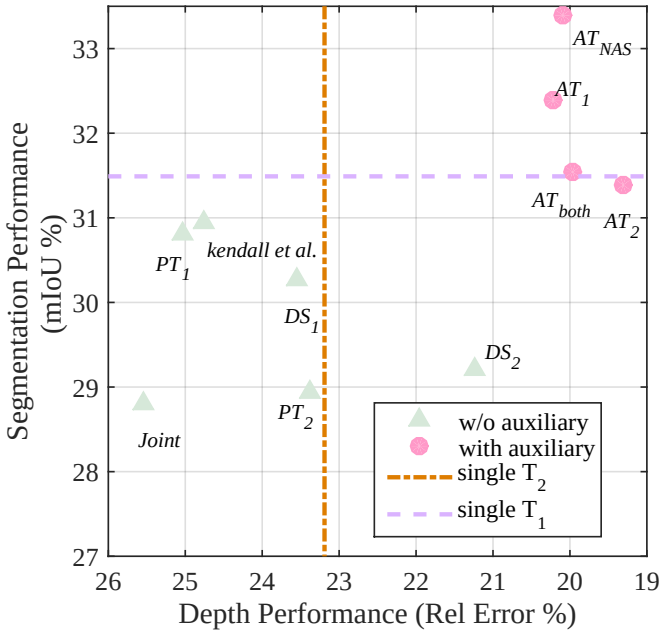


FIGURE 6.4. **Performance of different training strategies.** We report the depth prediction and semantic segmentation results on the *NYUD-v2*. Top-right is better. We can see that adding an auxiliary network can significantly boost the performance, even better than that of a single task.

Different Training Strategies

To investigate the effectiveness of the proposed method, we include the following training strategies for study to train a multi-task system on *NYUD-v2* with two tasks: T_1 (semantic segmentation) and T_2 (depth estimation). i) **Single**: Solve the task independently (shown in Fig. 6.4 with the dotted line parallel to the coordinate axis). 2) **Joint**: Jointly train two tasks with the sum of two losses. 3) **Prior**: Initialize from a well-trained task (T_1 or T_2), and then jointly train two tasks (shown in Fig. 6.4 with PT_1 and PT_2, separately). 4) **Kendall et al.** [61]: Use the uncertainty weighting proposed by Kendall et al. [61]. 5) **Deep Supervision**: Train with additional losses supervised by T_1 or T_2 (shown in Fig. 6.4 with DS_1 and DS_2, separately). We add the additional losses at the end of each block, and sum them all with scale of 0.1. 6) **Auxi**: Train with the auxiliary modules. AT_1, AT_2 and AT_{both} represent for adding basic auxiliary module (described in Sec. 6.3.2) to supervise the T_1, T_2 or T_1 and T_2, respectively. AT_{NAS} represents for adding the *NAS* structure [1] as auxiliary modules with two task supervisions. All reported results are based on the *Baseline* network. The detailed training setup can be found in Sec. A.2 in the appendix.

[1] The detailed structure can be found in Sec. A.1 in the appendix.

TABLE 6.3. The performance by applying the auxiliary module on top of different network structures.

Method	baseline		context		U-shape	
	Rel Error	mIoU	Rel Error	mIoU	Rel Error	mIoU
#Params (M)	3.31		3.98		4.19	
Depth-only	18.7	-	17.13	-	14.91	-
Segmentation-only	-	34.6	-	36.21	-	38.05
Joint	19.4	33.1	17.64	33.06	15.12	37.38
Ours	**18.6**	**34.8**	**16.03**	**37.13**	**14.64**	**38.85**

From Fig. 6.4, we can see that directly *joint* training the two tasks will decrease the performance for both, which indicates the competing task objectives make the shared weights hard to optimize. The strategies of *prior* and *Kendall et al.* can have a better performance than *joint* but are still worse than the single task baseline. Adding *deep supervision* is sensitive to the location and the scale, therefore it has limited contribution to the performance. In contrast, the proposed auxiliary overparameterization training method can significantly boost the performance by providing extra gradients for the shared weights during training. The overparameterized auxiliary module increases the capacity of the whole network, which assists the shared backbone to learn a better representation and generalize better. By adding a single auxiliary module (i.e., AT_1 or AT_2) or adding both modules AT_{both}, we find solutions that are better than two single tasks. It indicates that the inductive bias between different tasks can help the training of the single task. To replace the basic auxiliary modules with *NAS* auxiliary modules, we get a 1.9% improvement on mIoU for the semantic segmentation task, while the performance of the depth prediction task is comparable.

To further analyze the benefit of training with the auxiliary module, we show the gradients and the loss curves of the training process in Fig. 6.5. *Base* means we jointly train the two tasks. *Auxi* indicates that we add the auxiliary module to supervise the depth estimation. All the training settings are kept the same. We randomly sample some parameters from the shared convolutional layers to calculate the average gradient during the training process. The training loss curve of depth estimation is also included. From Fig. 6.5, we can observe adding an auxiliary module provide extra gradients for the shared parameters, which may accelerate the optimization of the network [101]. Meanwhile, the training loss for the depth estimation task is lower, which shows that the auxiliary overparameterization method can help the network convergence better.

Different Main Architectures

In this section, we further explore the effectiveness of the proposed overparameterization training strategy with different main network structures. We employ the structure, which has the highest reward on the validation set during the search processing as

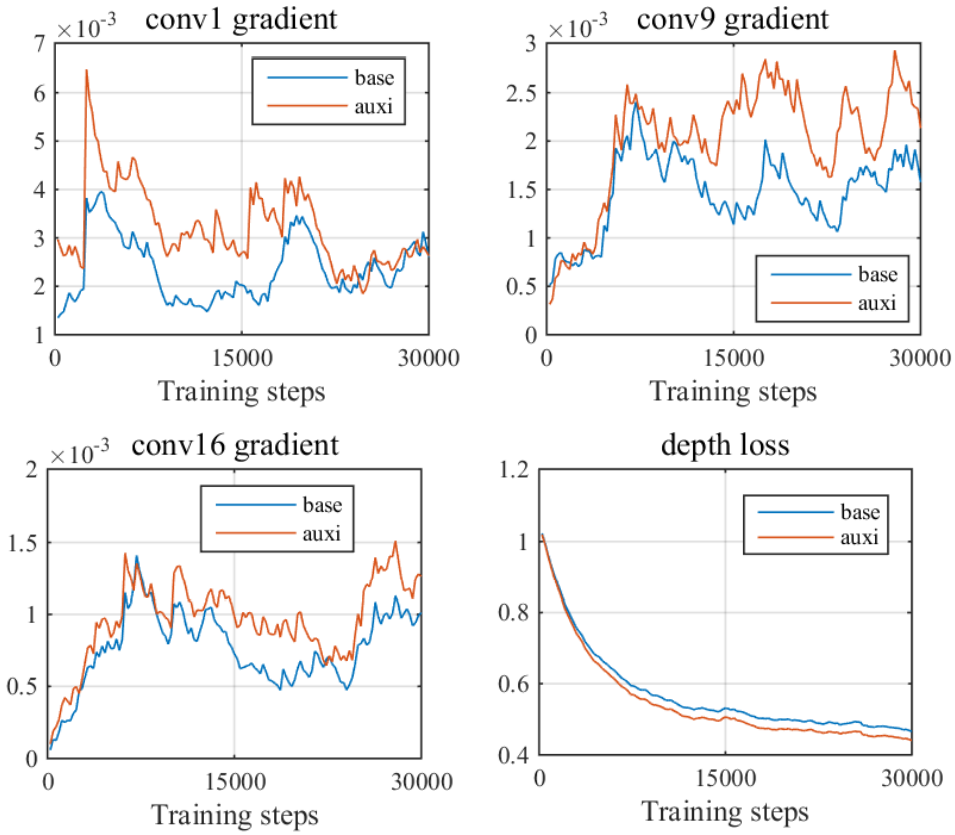


FIGURE 6.5. **Training curves**. Base: jointly train two tasks. Auxi: adding auxiliary module to supervise the depth estimation. Here we show three samples of the average gradients for different layers (a-c) and the training loss curve for the depth estimation task. We can observe that the gradient w.r.t the shared parameters is enhanced.

our auxiliary module. All the structures are pre-trained on the *NYUD-V2-expansion* and then fine-tuned on the *NYUD-V2*. The results are shown in Table 6.3. From Table 6.3, we can observe that as we increase the network's capacity, the performance of each task baseline boosts. It is a general way for one to get a higher performance while sacrificing speed. Moreover, for different main network architectures, the auxiliary module can improve the performance for both tasks. It shows that the auxiliary network can provide extra gradient for the shared backbone and mimic the effect of overparameterization during training, which can help the convergence of the network. More importantly, even though the auxiliary module introduces extra parameters during training, we can easily remove this module during the inference, just like a pruning process.

Different Auxiliary Architectures.

After testing on different variants of main networks, we then explore the influence of the capacity of the auxiliary module. We conduct experiments with the same architecture *Context* and employ variants for the auxiliary module. The results of

TABLE 6.4. Results on the test set of *NYUD-v2*. We show that the auxiliary module can improve the strong baseline to compare with other state-of-the-art methods designed for multi-task.

Method	Segmentation		Depth Prediction	
	mIoU	Pixel Acc.	Rel Error	RMS Error
Eigen and Fergus [30]	34.1	65.6	0.158	0.641
Sem-CRF+[96]	39.2	68.6	0.200	0.816
Real-Time [99]	42.0	-	0.149	0.565
TRL-ResNet50[172]	46.4	**76.2**	0.144	0.501
ResNet-50-*depth-only*	-	-	0.137	0.509
ResNet-50-*seg-only*	43.0	71.3	-	-
ResNet-50-*joint*	42.9	71.1	0.143	0.563
ResNet-50-*Auxi-Base*	46.1	73.8	**0.129**	**0.466**
ResNet-50-*NAS*	**46.6**	74.2	0.135	0.470

TABLE 6.5. The performance w.r.t. different auxiliary architectures.

Method	basic-conv1x1	basic-conv3x3	NAS
Rel Error %	16.12	**15.94**	16.03
mIoU %	35.02	35.71	**37.13**

various auxiliary modules with different structures are reported in Table 6.5. From the table, we can observe that increasing the complexity of the auxiliary module can further boost the performance. For example, by replacing the convolution 1×1 in the basic structure of the adaptor with a large kernel of 3×3, we get performance gain on both two tasks by 0.18% and 0.69% respectively. For the *NAS* method, the segmentation task has a 1.42% improvement compared to the basic structure with convolution 3×3. The automatic searching method can avoid human heuristics, and get better performance. The employed NAS has several ways [98] to accelerate the training, therefore, we can finish searching for more than 1000 structures in one GPU day. Moreover, we suppose the employment of NAS may offer interpretability for the auxiliary network. We can see the trend of each operator, for example, the number of 'skip connected' operators decreases along with the RL training, which indicates the 'skip connected' is not helpful for the auxiliary network.

Comparison with State-of-the-art Methods

We further test the proposed approach to compare with other state-of-the-art methods designed for multi-task. We replace the backbone to ResNet-50, and employ task-specific decoder follow [161] for each task. The results are shown in Table 6.4. The auxiliary module can also boost the performance with a strong baseline such as ResNet-50. To compare with TRL-ResNet-50 [172], we get better results over the mIoU, Rel Error, and RMS Error by adding the NAS auxiliary network during training.

Network Pruning with Auxiliary Learning

TABLE 6.6. Fine-tuning pruned ResNet-18 with/without the auxiliary module on NYUDv2.

	Baseline(1.0)	pruning(0.5)	pruning(0.5)+Auxi
Pixel Acc. %	**57.1**	54.6	55.2
Rel Err. %	**24.4**	26.1	25.7

We also conduct an experiment to verify that our method is complementary to prune a compact structure. They focus on designing a new compact structure, while we work on the off-the-shelf compact models. Our method can improve the convergence of a pruned compact network during training. Here, we combine our method with the pruned structure and show the result in Table 6.6. We use Network Slimming [88] to prune a ResNet-18 with a pruning ratio of 50%, then we fine-tune on the NYUD-V2 dataset with and without our basic auxiliary module. In particular, the auxiliary training improves the pruned baseline by 0.6% and 0.4% for semantic segmentation and depth estimation, respectively.

Experiments on SUNRGBD

Finally, we conduct experiments on a larger dataset SUNRGBD with our proposed training method to verify its generalization ability. The experiments are based on a *U-shape* MobileNetV2, we further expand the number of the tasks from two to three by adding a head performing the surface normal estimation. The results are reported in Table 6.7. We add the auxiliary module for each task separately or add them all together. We observe the overparameterization training approach boosts the performance for a specific single task. When combining them together, we can get an average improvement compared to the *joint* baseline. Besides, we can see that the depth prediction task and the surface normal estimation task are highly related, especially when we add the auxiliary module supervised by the surface normal loss, and the performance of the depth prediction branch is better than the *joint* baseline, even the *Auxi-depth*. It shows that the context information from the surface normal task plays an important role in the depth prediction task.

6.5 Conclusion

In some cases, the large teacher net and temporal information are hard to obtain. We apply the overparameterization principle to designing an auxiliary overparameterization strategy for training efficient networks. In specific, we have designed an auxiliary module to make a wider network that shares the weights of the compact network to mimic an overparameterization during training. In the inference process, we can discard the auxiliary module. Moreover, we have utilized the neural architecture search method to automatically explore the structure of the auxiliary module

TABLE 6.7. Semantic segmentation, depth prediction, and surface normal estimation results on the test set of SUNRGBD. *Auxi-depth*, *Auxi-seg* and *Auxi-normal* represent for adding a single auxiliary module with supervised loss from depth estimation, semantic segmentation, and surface normal task, respectively. *Auxi-all* represents for adding them all together.

Auxi type	Segmentation		Depth	Surface Normal
	Pixel Acc.	mIoU	Rel Error	Mean Error (°)
joint	80.7	53.7	22.8	28.7
Auxi-depth	80.8	54.0	20.8	27.0
Auxi-seg	**82.9**	**55.3**	21.7	28.3
Auxi-normal	80.9	54.4	**20.3**	**25.7**
Auxi-all	81.3	54.9	20.5	26.1

to avoid human heuristics. The proposed approach can be treated as an alternative strategy to knowledge distillation for training compact models. We have empirically shown that such a training strategy can improve the optimization and generalization of the compact model with two case studies: the lightweight single task and the hard parameter sharing multi-task system.

Chapter 7

Conclusion

Deep learning has developed rapidly in recent years. Related technologies have been applied in various fields such as agriculture, medical care, transportation, and security. Therefore, it is of great significance to build a lightweight model that is efficient, stable, and accurate.

In this thesis, we develop efficient fully-convolutional networks for dense prediction tasks. We propose several training schemes to improve the performance of efficient models with extra training constraints. First, we discuss how to use structural knowledge in building a knowledge distillation framework for dense prediction tasks. We propose pair-wise distillation and holistic distillation based on generative adversarial networks. The pair-wise distillation aligns the correlations among pixels between the teacher and the student network. The holistic distillation encourages the distribution of the output logits map to be consistent between two networks. We first discuss the effectiveness of the semantic segmentation task and then extend the distillation framework to object detection and depth estimation tasks. Superior performance has been achieved on five datasets among three dense prediction tasks, demonstrating the effectiveness and the generalization ability of the proposed method. The performance can be better if trained extra unlabeled data is involved. To the best of our knowledge, we are the first to propose the concept of structural knowledge distillation for dense prediction tasks.

Based on the knowledge distillation framework for dense prediction tasks, we further propose a simple and effective channel-wise distillation. The channel-wise distillation method normalized the activation values in each channel to get the channel distribution, and then align the most significant region in each channel. It significantly reduces the training cost and achieves better performance on semantic segmentation and object detection.

We further extend our core idea, adding extra constraints during training, to video segmentation. We propose to use an efficient convolutional network to process the video sequence frame by frame. In the previous chapters, we benefit from adding extra constraints from a larger teacher net. In the semantic video segmentation, except for learning from a large teacher net, we further explore the constraints between temporal frames. We model the motion between frames by using a pre-trained optical flow prediction network and propose a motion loss. To reduce the gap between the

efficient models and the large models, two new temporal knowledge distillation methods are designed. The temporal consistency and accuracy of the efficient model can be improved.

Finally, as the teacher network is hard to obtain in some cases, i.e. the multi-task learning system, we explore the effectiveness of the auxiliary module. Benefit from the over-parametric theory, we can get better performance with the help of the auxiliary module. We demonstrate the effectiveness of the proposed auxiliary module with semantic segmentation, depth prediction, and surface normal. The advantages of apply training methods to improving the performance of efficient models can be summarized as follows:

- The training methods do not increase any computational costs during the testing process.
- The training methods are general and can be applied to various network structures.
- Some training methods, such as knowledge distillation and motion losses, can be applied to unlabeled data.

Limitations and Future work. In this thesis, we mainly apply our method to dense prediction tasks on 2D images and extend to video sequences under supervised settings. We start a trail on applying our method to unlabeled data and achieve promising performance. With the development and innovation of technology, artificial intelligence systems will face more abundant data input sources. Large, rich unlabeled data will also become very easy to obtain. Being able to make full use of multi-modal input data and unlabeled input data will be very meaningful for the training of compact models. There are still some limitations in our work. The improvement through training methods is limited by the capacity of the efficient model. Thus, developing more efficient network structures is still important. For example, using neural network architecture search will be helpful to find low computational cost models. Our method can serve for newly designed structures. Besides, the training constraints can also come from some prior knowledge, like in three of my collaborative works [155, 153, 45]. I hope this thesis can serve as a new way to build compact networks in the computer vision community.

Appendix A

Appendix for Auxiliary Overparameterization

In this supplementary material, we provide additional details for Chapter 6.

A.1 Extension to Multi-task

To train a multi-task learning system, firstly we need to get a large dataset with the mapping from a single input space $\mathcal{X}$ to multiple labels $\{\mathcal{Y}\}_{t\in\{T\}}^{t}$, i.i.d. $\{x_i, y_i^1, \cdots, y_i^T\}_{i\in\{N\}}$, where T is the number of the tasks and N is the number of the data samples. The MTL network $\mathcal{F}$ consists of a single shared parameter space $\boldsymbol{\theta}^{sh}$ and T sets of task-specific parameters $\boldsymbol{\theta}^t$. Here we denote the loss function for task t as $\mathcal{L}^t(\cdot)$ to describe the difference between the output of $\mathcal{F}$ for task t: $f^t(\mathcal{X}; \boldsymbol{\theta}^{\mathcal{B}}, \boldsymbol{\theta}^t)$, and the ground truth label: $\{\mathcal{Y}\}^t$. Therefore, we formulate the final objective function for a hard parameter sharing network as follows:

$$\min_{\{\boldsymbol{\theta}^{sh}, \boldsymbol{\theta}^1, \cdots, \boldsymbol{\theta}^T\}} \sum_{t=1}^{T} \alpha^t \mathcal{L}^t(f^t(\mathcal{X}; \boldsymbol{\theta}^{sh}, \boldsymbol{\theta}^t), \{\mathcal{Y}\}^t), \tag{A.1}$$

where α^t is a combination coefficient for the t-th task. In this work, we do not focus on adjusting the α^t to boost the performance, so we set the $\alpha^t = 1$ to all cases to simplify the basic objective function to $\min_{\{\boldsymbol{\theta}^{sh}, \boldsymbol{\theta}^1, \cdots, \boldsymbol{\theta}^T\}} \sum_{t=1}^{T} \mathcal{L}^t(f^t(\mathcal{X}; \boldsymbol{\theta}^{sh}, \boldsymbol{\theta}^t), \{\mathcal{Y}\}^t)$.

The basic structure of an auxiliary module for task t is the same as shown in the main paper, and we can extend the objective function (i.e., Eq. (2)) in the main paper to multi-task case as follows:

$$\begin{aligned} \min_{\{\boldsymbol{\theta}^{sh}, \boldsymbol{\theta}^1, \cdots, \boldsymbol{\theta}^T, \boldsymbol{\theta}_a^1, \cdots, \boldsymbol{\theta}_a^T\}} & \Big(\sum_{t=1}^{T} \mathcal{L}^t(f^t(\mathcal{X}; \boldsymbol{\theta}^{sh}, \boldsymbol{\theta}^t), \{\mathcal{Y}\}^t) \\ & + \sum_{t=1}^{T} \mathcal{L}_{aux}^t(\mathcal{A}^t(\boldsymbol{O}_1, \cdots, \boldsymbol{O}_L; \boldsymbol{\theta}^{sh}, \boldsymbol{\theta}_a^t), \{\mathcal{Y}\}^t)\Big), \end{aligned} \tag{A.2}$$

where $\boldsymbol{\theta}_a^t$ and $\mathcal{L}_{aux}^t(\cdot)$ represent the adaptor parameters and the auxiliary loss for the t-th task, respectively.

To extend the search policy to the multi-task case, as described in the main paper, we enlarge the output length of the controller to T times as many as a single task and follow the order in Fig. A.1. We have reserved the possibility of an association between different tasks in the search space, for example, for the *l-th* cell in task t, it can choose the output of task 1 to task $t-1$ in the first $l-1$ cells as input. The auxiliary module $\mathcal{A}$ can learn heretical relationships among all tasks. We show the search results of a sampled structure, which has the highest reward score on the validation set in Fig. A.2.

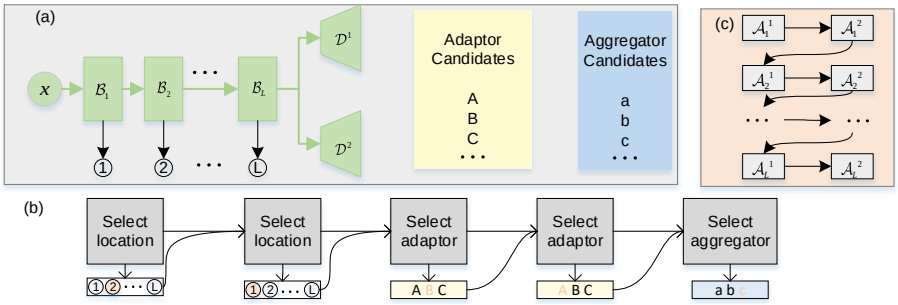


FIGURE A.1. (a) Auxiliary structure search space for multi-task learning. (b) Controller output for generating a single cell for the *l-th* auxiliary cell of task t, $\mathcal{A}_l^t$. (c) The order of generating the whole auxiliary module recursively among different tasks.

A.2 Training Details

In this section, we will introduce the training setups for the experiments. All the experiments are implemented in PyTorch.

A.2.1 Semantic Segmentation

As for the single task of semantic segmentation, we use the open-source implementation [179], and follow their settings. We set the initial learning rate as 0.02 and weight decay as 0.0001 by default, the input image is resized to the length randomly chosen from the set 300, 375, 450, 525, 600 due to that the images are of various sizes on ADE20K. The batch size is 8 and we also synchronize the mean and standard-deviation of BN cross multiple GPUs. We train all the experiments for 20 epochs.

A.2.2 Multi-task Learning

Common data augmentation is employed with the random flip, random reshape (from 0.5 to 2.1) and random crop with the training size 385×385. The ground truth of depth should be normalized with the scale of random reshape. The batch size is 12 for all experiments. We verify our proposed method on the *NYUD-v2* dataset. We train the *single* task baseline and the *joint* baseline for $30k$ iterations with the initial learning rate of 0.01 and weight decay of 0.0001. The learning rate is multiplied by $(1 - \frac{iter}{max-iter})^{0.9}$. For the *prior* training strategy, we initialize the network with the

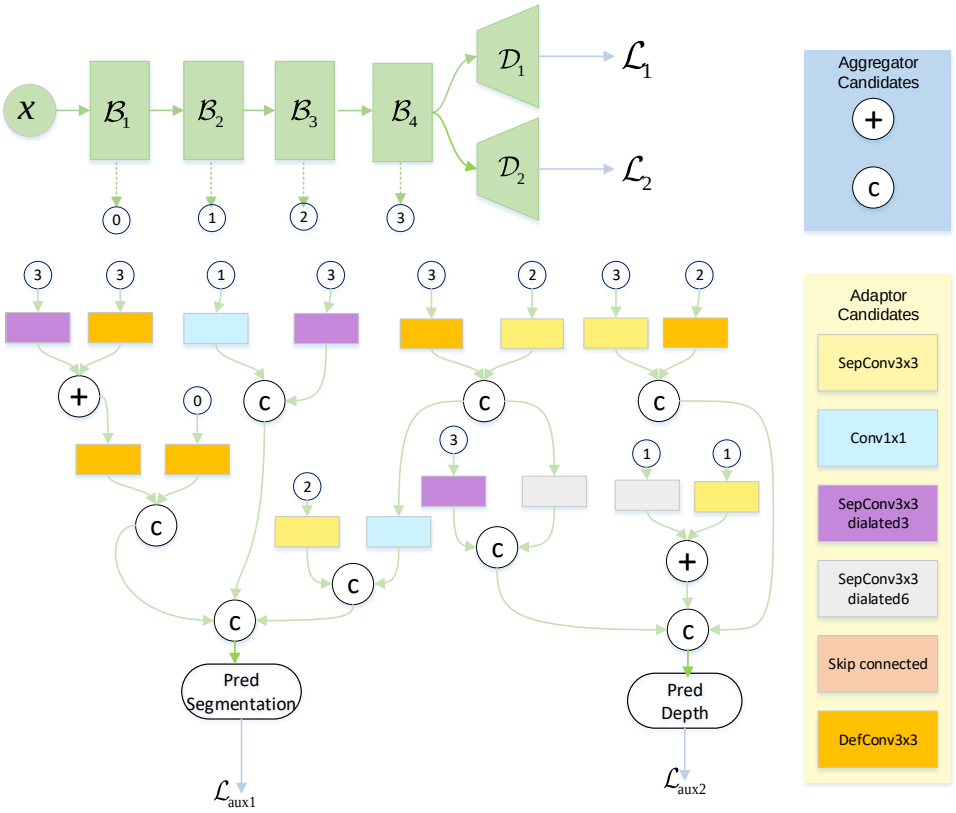


FIGURE A.2. The auxiliary modules are sampled by reinforcement learning. We show the detailed structure used in the multi-task experiments for depth prediction and semantic segmentation.

single task baseline, and then jointly train two tasks with a learning rate of 0.001 for $30k$ iterations with the same learning rate schedule. To make a fair comparison, for adding a single auxiliary module, we follow the same training setting with *prior*. And when adding auxiliary modules supervised by two tasks, we follow the training setups of the *joint* baseline. In other sections, the models are pre-trained on *NYUD-v2-expansion* for $40k$ iterations with an initial learning rate of 0.01, and then fine tune on the *NYUD-v2* with a fixed learning rate of 0.00001 for $10k$ iterations. On the *SUNRGBD* dataset, the models are trained for $80k$ iterations with the initial learning rate of 0.01 both with and without auxiliary modules.

A.3 Visualization Results

In this section, we show some visualization results on *NYUD-v2* and *SUNRGBD*. The multi-task system can generate multiple outputs in one forward pass.

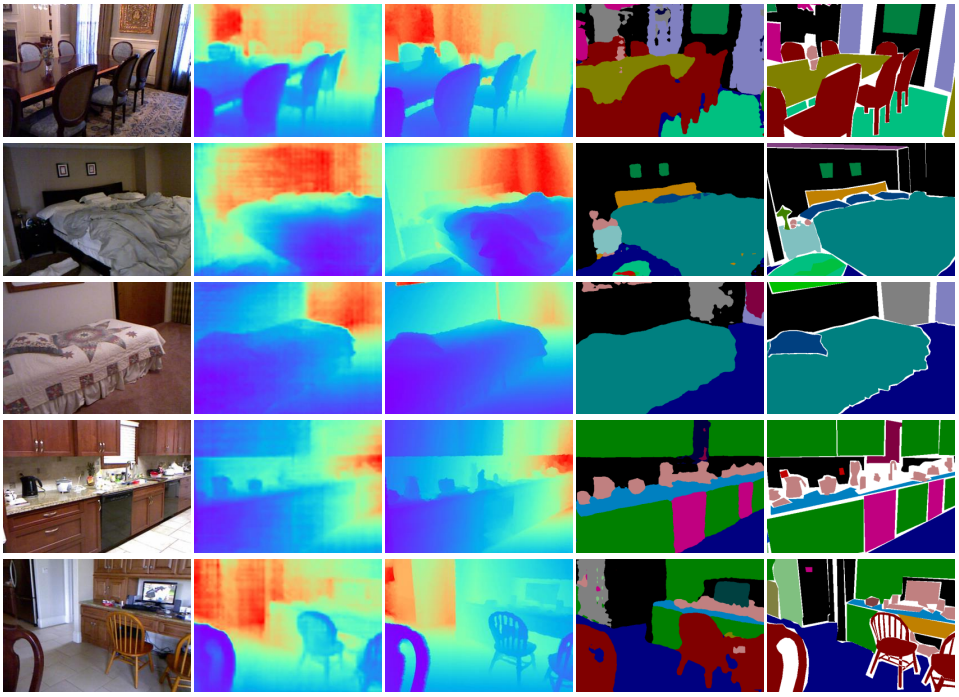


FIGURE A.3. **Visualization Results on NYUD-v2** (a) Input image. (b) Predicted depth results. (c) Ground truth depth results. (d) Predicted semantic segmentation results. (e) Ground truth semantic segmentation results.

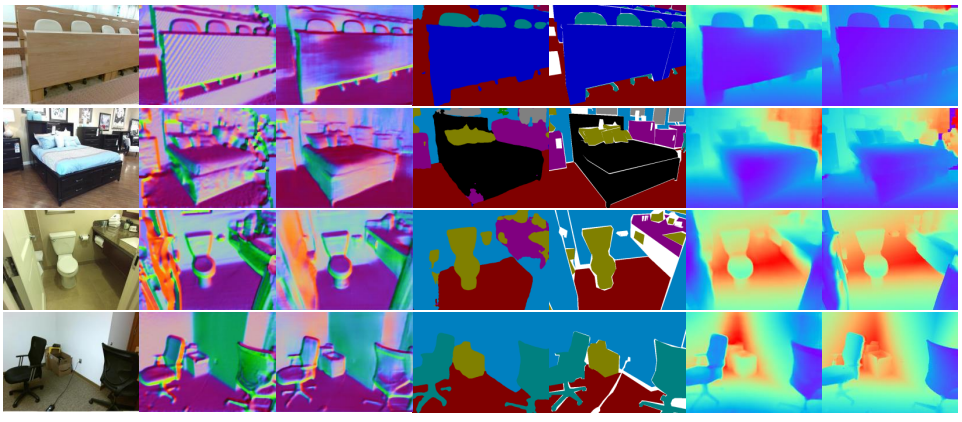


FIGURE A.4. **Visualization Results on SUNRGBD** (a) Input image. (b) Predicted surface normal. (c) Ground truth surface normal. (d) Predicted semantic segmentation results. (e) Ground truth semantic segmentation results. (f) Predicted depth results. (e) Ground truth depth results.

Bibliography

[1] https://github.com/warmspringwinds/pytorch-segmentation-detection/blob/master/pytorch_segmentation_detection/utils/flops_benchmark.py. 2018.

[2] Paszke Adam et al. "ENet: A Deep Neural Network Architecture for Real-Time Semantic Segmentation". In: *IEEE Conf. Comput. Vis. Pattern Recog.* (2016).

[3] Romero Adriana et al. "Fitnets: Hints for thin deep nets". In: *Int. Conf. Learn. Represent.* (2015).

[4] Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. "Learning and generalization in overparameterized neural networks, going beyond two layers". In: *arXiv preprint arXiv:1811.04918* (2018).

[5] Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. "A convergence theory for deep learning via over-parameterization". In: *arXiv preprint arXiv:1811.03962* (2018).

[6] Sanjeev Arora, Nadav Cohen, and Elad Hazan. "On the Optimization of Deep Networks: Implicit Acceleration by Overparameterization". In: *Proc. Int. Conf. Mach. Learn.* 2018, pp. 244–253.

[7] Jimmy Ba and Rich Caruana. "Do deep nets really need to be deep?" In: *Proc. Advances in Neural Inf. Process. Syst.* 2014, pp. 2654–2662.

[8] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 12 (2017), pp. 2481–2495.

[9] Jonathan Baxter. "A Bayesian/information theoretic model of learning to learn via multiple task sampling". In: *Machine learning* (1997), pp. 7–39.

[10] Anil Bhattacharyya. "On a measure of divergence between two statistical populations defined by their probability distributions". In: *Bull. Calcutta Math. Soc.* 35 (1943), pp. 99–109.

[11] Gabriel J Brostow et al. "Segmentation and recognition using structure from motion point clouds". In: *Proc. Eur. Conf. Comp. Vis.* Springer. 2008, pp. 44–57.

[12] Han Cai et al. "Efficient architecture search by network transformation". In: *AAAI Conf. Artificial Intelligence*. 2018.

[13] Yuanzhouhan Cao, Zifeng Wu, and Chunhua Shen. "Estimating depth from monocular images as classification using deep fully convolutional residual networks". In: *IEEE Trans. Circuits Syst. Video Technol.* 28.11 (2017), pp. 3174–3182.

[14] Yue Cao et al. "Gcnet: Non-local networks meet squeeze-excitation networks and beyond". In: 2019, pp. 0–0.

[15] Siddhartha Chandra, Camille Couprie, and Iasonas Kokkinos. "Deep spatio-temporal random fields for efficient video segmentation". In: *IEEE Conf. Comput. Vis. Pattern Recog.* 2018, pp. 8915–8924.

[16] Guobin Chen et al. "Learning efficient object detection models with knowledge distillation." In: *Adv. Neural Inform. Process. Syst.* 2017, pp. 742–751.

[17] Liang-Chieh Chen et al. "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 40.4 (2018), pp. 834–848.

[18] Liang-Chieh Chen et al. "Encoder-decoder with atrous separable convolution for semantic image segmentation". In: *Proc. Eur. Conf. Comp. Vis.* (2018).

[19] Liang-Chieh Chen et al. "Rethinking Atrous Convolution for Semantic Image Segmentation". In: *IEEE Conf. Comput. Vis. Pattern Recog.* (2017).

[20] Liang-Chieh Chen et al. "Semantic image segmentation with deep convolutional nets and fully connected crfs". In: *Proc. Int. Conf. Learn. Representations*. 2015.

[21] Wuyang Chen et al. "FasterSeg: Searching for Faster Real-time Semantic Segmentation". In: *Int. Conf. Learn. Represent.* (2020).

[22] Yu Chen et al. "Adversarial PoseNet: A Structure-Aware Convolutional Network for Human Pose Estimation". In: *Proc. IEEE Int. Conf. Comp. Vis.* 2017, pp. 1212–1221.

[23] Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. "DarkRank: Accelerating Deep Metric Learning via Cross Sample Similarities Transfer". In: *Proc. Eur. Conf. Comp. Vis.* (2018).

[24] Jingchun Cheng et al. "Segflow: Joint learning for video object segmentation and optical flow". In: *Int. Conf. Comput. Vis.* 2017, pp. 686–695.

[25] Marius Cordts et al. "The Cityscapes Dataset for Semantic Urban Scene Understanding". In: *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.* 2016.

[26] Marius Cordts et al. "The cityscapes dataset for semantic urban scene understanding". In: *IEEE Conf. Comput. Vis. Pattern Recog.* (2016).

[27] Jifeng Dai et al. "Deformable convolutional networks". In: *Int. Conf. Comput. Vis.* 2017, pp. 764–773.

[28] Simon Jégou1 Michal Drozdzal, David Vazquez, and Adriana Romero1 Yoshua Bengio. "The One Hundred Layers Tiramisu: Fully Convolutional DenseNets for Semantic Segmentation". In: *Proc. Workshop of IEEE Conf. Comp. Vis. Patt. Recogn.* (2017).

[29] Romera Eduardo et al. "ERFNet: Efficient Residual Factorized ConvNet for Real-Time Semantic Segmentation". In: *IEEE Trans. Intell. Transportation Syst.* (2017).

[30] David Eigen and Rob Fergus. "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture". In: *Int. Conf. Comput. Vis.* 2015, pp. 2650–2658.

[31] David Eigen, Christian Puhrsch, and Rob Fergus. "Depth Map Prediction from a Single Image Using a Multi-Scale Deep Network". In: *Proc. Advances in Neural Inf. Process. Syst.* 2014.

[32] Mark Everingham et al. "The pascal visual object classes (voc) challenge". In: *Int. J. Comput. Vis.* (2010).

[33] Mohsen Fayyaz et al. "STFCN: spatio-temporal fully convolutional neural network for semantic segmentation of street scenes". In: *ACCV*. Springer. 2016, pp. 493–509.

[34] Xiaohan Fei, Alex Wang, and Stefano Soatto. "Geo-Supervised Visual Depth Prediction". In: *arXiv: Comp. Res. Repository*. Vol. abs/1807.11130. 2018.

[35] Cheng-Yang Fu et al. "Dssd: Deconvolutional single shot detector". In: *arXiv: Comp. Res. Repository*. Vol. abs/1701.06659. 2017.

[36] Huan Fu et al. "Deep Ordinal Regression Network for Monocular Depth Estimation". In: *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.* 2018, pp. 2002–2011.

[37] Raghudeep Gadde, Varun Jampani, and Peter V Gehler. "Semantic video cnns through representation warping". In: *Int. Conf. Comput. Vis.* 2017, pp. 4453–4462.

[38] Ross Girshick. "Fast r-cnn". In: *Proc. IEEE Int. Conf. Comp. Vis.* 2015, pp. 1440–1448.

[39] Ian J Goodfellow et al. "Generative Adversarial Nets". In: *Proc. Advances in Neural Inf. Process. Syst.* 3 (2014), pp. 2672–2680.

[40] Ishaan Gulrajani et al. "Improved training of wasserstein gans". In: *Proc. Advances in Neural Inf. Process. Syst.* 2017, pp. 5767–5777.

[41] Kin Gwn Lore et al. "Generative adversarial networks for depth map estimation from RGB video". In: *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.* 2018, pp. 1177–1185.

[42] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.* (2016), pp. 770–778.

[43] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *IEEE Conf. Comput. Vis. Pattern Recog.* (2016).

[44] Kaiming He et al. “Deep residual learning for image recognition”. In: *IEEE Conf. Comput. Vis. Pattern Recog.* 2016, pp. 770–778.

[45] Tong He et al. “Instance-Aware Embedding for Point Cloud Instance Segmentation”. In: *Eur. Conf. Comput. Vis.* Springer. 2020, pp. 255–270.

[46] Tong He et al. “Knowledge Adaptation for Efficient Semantic Segmentation”. In: *IEEE Conf. Comput. Vis. Pattern Recog.* (2019).

[47] Tong He et al. “Knowledge Adaptation for Efficient Semantic Segmentation”. In: *IEEE Conf. Comput. Vis. Pattern Recog.* 2019, pp. 578–587.

[48] Byeongho Heo et al. “A comprehensive overhaul of feature distillation.” In: *Int. Conf. Comput. Vis.* 2019, pp. 1921–19302.

[49] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. “Distilling the Knowledge in a Neural Network”. In: *Adv. Neural Inform. Process. Syst.* (2014).

[50] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. “Distilling the Knowledge in a Neural Network”. In: *arXiv: Comp. Res. Repository* abs/1503.02531 (2015).

[51] Yuenan Hou et al. “Inter-Region Affinity Distillation for Road Marking Segmentation”. In: *IEEE Conf. Comput. Vis. Pattern Recog.* 2020, pp. 12486–12495.

[52] Andrew G Howard et al. “Mobilenets: Efficient convolutional neural networks for mobile vision applications”. In: *arXiv: Comp. Res. Repository* abs/1704.04861 (2017).

[53] Junjie Hu et al. “Revisiting single image depth estimation: Toward higher resolution maps with accurate object boundaries”. In: *Proc. Winter Conf. on Appl. of Comp0 Vis.* IEEE. 2019, pp. 1043–1051.

[54] Gao Huang et al. “Densely Connected Convolutional Networks”. In: *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.* (2017), pp. 2261–2269.

[55] Xinyu Huang et al. “The ApolloScape Dataset for Autonomous Driving”. In: *arXiv: 1803.06184* (2018).

[56] Forrest N. Iandola et al. “SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <1MB model size”. In: *arXiv: Comp. Res. Repository* abs/1602.07360 (2016).

[57] Samvit Jain, Xin Wang, and Joseph E Gonzalez. “Accel: A Corrective Fusion Network for Efficient Semantic Segmentation on Video”. In: *IEEE Conf. Comput. Vis. Pattern Recog.* 2019, pp. 8866–8875.

[58] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. “Perceptual Losses for Real-Time Style Transfer and Super-Resolution”. In: *Proc. Eur. Conf. Comp. Vis.* (2016), pp. 694–711.

[59] Long Jonathan, Shelhamer Evan, and Darrell Trevor. "Fully Convolutional Networks for Semantic Segmentation". In: *IEEE Conf. Comput. Vis. Pattern Recog.* (2015).

[60] Tero Karras et al. "Progressive growing of gans for improved quality, stability, and variation". In: *Proc. Int. Conf. Learn. Representations* (2018).

[61] Alex Kendall, Yarin Gal, and Roberto Cipolla. "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics". In: *IEEE Conf. Comput. Vis. Pattern Recog.* 2018, pp. 7482–7491.

[62] Iasonas Kokkinos. "UberNet: Training a Universal Convolutional Neural Network for Low-, Mid-, and High-Level Vision Using Diverse Datasets and Limited Memory". In: *IEEE Conf. Comput. Vis. Pattern Recog.* 2017, pp. 5454–5463.

[63] Wei-Sheng Lai et al. "Learning blind video temporal consistency". In: *Eur. Conf. Comput. Vis.* 2018, pp. 170–185.

[64] Iro Laina et al. "Deeper depth prediction with fully convolutional residual networks". In: *Proc. Int. Conf. 3D Vision (3DV)*. IEEE. 2016, pp. 239–248.

[65] Chen-Yu Lee et al. "Deeply-supervised nets". In: *Artificial Intelligence and Statistics*. 2015, pp. 562–570.

[66] Hao Li et al. "Pruning filters for efficient convnets". In: *arXiv preprint arXiv:1608.08710* (2016).

[67] Quanquan Li, Shengying Jin, and Junjie Yan. "Mimicking Very Efficient Network for Object Detection". In: *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.* (2017), pp. 7341–7349.

[68] Quanquan Li, Shengying Jin, and Junjie Yan. "Mimicking very efficient network for object detection". In: *IEEE Conf. Comput. Vis. Pattern Recog.* (2017).

[69] Ruibo Li et al. "Deep attention-based classification network for robust depth prediction". In: *Proc. Asian Conf. Comp. Vis.* 2018.

[70] Stan Z Li. *Markov random field modeling in image analysis*. Springer Science & Business Media, 2009.

[71] Yule Li, Jianping Shi, and Dahua Lin. "Low-latency video semantic segmentation". In: *IEEE Conf. Comput. Vis. Pattern Recog.* 2018, pp. 5997–6005.

[72] Lin et al. "Focal Loss for Dense Object Detection". In: *Int. Conf. Comput. Vis.* 2017.

[73] Guosheng Lin et al. "Efficient piecewise training of deep structured models for semantic segmentation". In: *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.* 2016, pp. 3194–3203.

[74] Guosheng Lin et al. "Refinenet: Multi-path refinement networks for dense prediction". In: *IEEE Trans. Pattern Anal. Mach. Intell.* (2019).

[75] Guosheng Lin et al. "RefineNet: Multi-Path Refinement Networks for High-Resolution Semantic Segmentation". In: *IEEE Conf. Comput. Vis. Pattern Recog.* (2017).

[76] Tsung-Yi Lin et al. "Focal loss for dense object detection". In: *Proc. IEEE Int. Conf. Comp. Vis.* 2017, pp. 2980–2988.

[77] Tsung-Yi Lin et al. "Microsoft coco: Common objects in context". In: *Proc. Eur. Conf. Comp. Vis.* Springer. 2014, pp. 740–755.

[78] Fayao Liu, Chunhua Shen, and Guosheng Lin. "Deep Convolutional Neural Fields for Depth Estimation from a Single Image". In: *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.* 2015.

[79] Fayao Liu et al. "Learning Depth from Single Monocular Images Using Deep Convolutional Neural Fields". In: *IEEE Trans. Pattern Anal. Mach. Intell.* (2016).

[80] Hanxiao Liu et al. "Hierarchical representations for efficient architecture search". In: *Int. Conf. Learn. Represent.* 2018.

[81] Huijun Liu. *LightNet: Light-weight Networks for Semantic Image Segmentation*. https://github.com/ansleliu/LightNet. 2018.

[82] Si Liu et al. "Surveillance video parsing with single frame supervision". In: *IEEE Conf. Comput. Vis. Pattern Recog.* 2017, pp. 413–421.

[83] Wei Liu et al. "Ssd: Single shot multibox detector". In: *Proc. Eur. Conf. Comp. Vis.* Springer. 2016, pp. 21–37.

[84] Yifan Liu et al. "Auto-painter: Cartoon image generation from sketch by using conditional Wasserstein generative adversarial networks". In: *Neurocomputing* 311 (2018), pp. 78–87.

[85] Yifan Liu et al. "Structured Knowledge Distillation for Dense Prediction". In: *IEEE Trans. Pattern Anal. Mach. Intell.* (2019).

[86] Yifan Liu et al. "Structured Knowledge Distillation for Semantic Segmentation". In: *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.* 2019, pp. 2604–2613.

[87] Yifan Liu et al. "Structured Knowledge Distillation for Semantic Segmentation". In: *IEEE Conf. Comput. Vis. Pattern Recog.* (2019).

[88] Zhuang Liu et al. "Learning efficient convolutional networks through network slimming". In: *Int. Conf. Comput. Vis.* 2017, pp. 2736–2744.

[89] Zhuang Liu et al. "Rethinking the value of network pruning". In: *Int. Conf. Learn. Represent.* 2019.

[90] Jonathan Long, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation". In: *IEEE Conf. Comput. Vis. Pattern Recog.* 2015, pp. 3431–3440.

[91] Pauline Luc et al. “Semantic segmentation using adversarial networks”. In: *arXiv: Comp. Res. Repository* abs/1611.08408 (2016).

[92] Sachin Mehta et al. “ESPNet: Efficient Spatial Pyramid of Dilated Convolutions for Semantic Segmentation”. In: *Proc. Eur. Conf. Comp. Vis.* (2018).

[93] Mehdi Mirza and Simon Osindero. “Conditional Generative Adversarial Nets”. In: *arXiv: Comp. Res. Repository* abs/1411.1784 (2014).

[94] Sudhanshu Mittal, Maxim Tatarchenko, and Thomas Brox. “Semi-supervised semantic segmentation with high-and low-level consistency”. In: *IEEE Trans. Pattern Anal. Mach. Intell.* (2019).

[95] Taylor Mordan et al. “Revisiting Multi-Task Learning with ROCK: a Deep Residual Auxiliary Block for Visual Detection”. In: *Adv. Neural Inform. Process. Syst.* 2018, pp. 1310–1322.

[96] Arsalan Mousavian, Hamed Pirsiavash, and Jana Košecká. “Joint semantic segmentation and depth estimation with deep convolutional networks”. In: *Proc. Int. Conf. 3D Vision (3DV)*. IEEE. 2016, pp. 611–619.

[97] Vladimir Nekrasov, Chunhua Shen, and Ian Reid. “Light-weight refinenet for real-time semantic segmentation”. In: *arXiv preprint arXiv:1810.03272* (2018).

[98] Vladimir Nekrasov et al. “Fast Neural Architecture Search of Compact Semantic Segmentation Models via Auxiliary Cells”. In: *IEEE Conf. Comput. Vis. Pattern Recog.* 2019.

[99] Vladimir Nekrasov et al. “Real-Time Joint Semantic Segmentation and Depth Estimation Using Asymmetric Annotations”. In: *Proc. Int. Conf. on Robotics and Automation* (2018).

[100] Vladimir Nekrasov et al. “Real-Time Joint Semantic Segmentation and Depth Estimation Using Asymmetric Annotations”. In: *arXiv: Comp. Res. Repository.* Vol. abs/1809.04766. 2018.

[101] Trong Phong Nguyen et al. “Extragradient method in optimization: Convergence and complexity”. In: *Journal of Optimization Theory and Applications* (2018), pp. 137–162.

[102] Xuecheng Nie, Jiashi Feng, and Shuicheng Yan. “Mutual Learning to Adapt for Joint Human Parsing and Pose Estimation”. In: *Eur. Conf. Comput. Vis.* 2018, pp. 502–517.

[103] David Nilsson and Cristian Sminchisescu. “Semantic video segmentation by gated recurrent flow propagation”. In: *IEEE Conf. Comput. Vis. Pattern Recog.* 2018, pp. 6819–6828.

[104] Marin Orsic et al. “In defense of pre-trained imagenet architectures for real-time semantic segmentation of road-driving images”. In: *IEEE Conf. Comput. Vis. Pattern Recog.* 2019.

[105] Sangyong Park and Yong Seok Heo. "Knowledge distillation for semantic segmentation using channel and spatial correlations and adaptive cross entropy". In: *Sensors* 20.16 (2020), p. 4616.

[106] Adam Paszke et al. "ENet: A Deep Neural Network Architecture for Real-Time Semantic Segmentation". In: *arXiv: Comp. Res. Repository* abs/1606.02147 (2016).

[107] Hieu Pham et al. "Efficient neural architecture search via parameter sharing". In: *Proc. Int. Conf. Mach. Learn.* 2018.

[108] Fitsum Reda et al. *flownet2-pytorch: Pytorch implementation of FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks*. https://github.com/NVIDIA/flownet2-pytorch. 2017.

[109] Joseph Redmon and Ali Farhadi. "YOLO9000: better, faster, stronger". In: *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.* 2017, pp. 7263–7271.

[110] Joseph Redmon and Ali Farhadi. "Yolov3: An incremental improvement". In: *arXiv: Comp. Res. Repository*. Vol. abs/1804.02767. 2018.

[111] Joseph Redmon et al. "You only look once: Unified, real-time object detection". In: *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.* 2016, pp. 779–788.

[112] Shaoqing Ren et al. "Faster r-cnn: Towards real-time object detection with region proposal networks". In: *Proc. Advances in Neural Inf. Process. Syst.* 2015, pp. 91–99.

[113] Shaoqing Ren et al. "Faster R-CNN: towards real-time object detection with region proposal networks". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 39.6 (2016), pp. 1137–1149.

[114] Eduardo Romera et al. "Efficient convnet for real-time semantic segmentation". In: *IEEE Intelligent Vehicles Symp.* 2017, pp. 1789–1794.

[115] Adriana Romero et al. "Fitnets: Hints for thin deep nets". In: *arXiv: Comp. Res. Repository* abs/1412.6550 (2014).

[116] Mehta Sachin et al. "ESPNet: Efficient Spatial Pyramid of Dilated Convolutions for Semantic Segmentation". In: *Eur. Conf. Comput. Vis.* (2018).

[117] Mark Sandler et al. "MobileNetV2: Inverted Residuals and Linear Bottlenecks". In: *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.* 2018.

[118] Mark Sandler et al. "MobileNetV2: Inverted Residuals and Linear Bottlenecks". In: *IEEE Conf. Comput. Vis. Pattern Recog.* (2018).

[119] Ashutosh Saxena, Min Sun, and Andrew Y Ng. "Learning 3-d scene structure from a single still image". In: *Proc. IEEE Int. Conf. Comp. Vis.* IEEE. 2007, pp. 1–8.

[120] John Schulman et al. "Proximal policy optimization algorithms". In: *arXiv preprint arXiv:1707.06347* (2017).

[121] Ozan Sener and Vladlen Koltun. "Multi-task learning as multi-objective optimization". In: *Adv. Neural Inform. Process. Syst.* 2018, pp. 527–538.

[122] E Shelhamer, J Long, and T Darrell. "Fully Convolutional Networks for Semantic Segmentation." In: *IEEE Trans. Pattern Anal. Mach. Intell.* 39.4 (2017), p. 640.

[123] Evan Shelhamer et al. "Clockwork convnets for video semantic segmentation". In: *Eur. Conf. Comput. Vis.* Springer. 2016, pp. 852–868.

[124] Xingjian Shi et al. "Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting". In: *Adv. Neural Inform. Process. Syst.* 2015, pp. 802–810.

[125] Andrew Spek, Thanuja Dharmasiri, and Tom Drummond. "CReaM: Condensed Real-time Models for Depth Prediction using Convolutional Neural Networks". In: *Int. Conf. on Intell. Robots and Sys.* IEEE. 2018, pp. 540–547.

[126] Ke Sun et al. "Deep High-Resolution Representation Learning for Human Pose Estimation". In: *IEEE Conf. Comput. Vis. Pattern Recog.* 2019.

[127] Ke Sun et al. "High-Resolution Representations for Labeling Pixels and Regions". In: *arXiv: Comp. Res. Repository* abs/1904.04514 (2019).

[128] Christian Szegedy et al. "Going deeper with convolutions". In: *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.* 2015, pp. 1–9.

[129] Christian Szegedy et al. "Rethinking the Inception Architecture for Computer Vision". In: *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.* (2016), pp. 2818–2826.

[130] Mingxing Tan and Quoc Le. "Efficientnet: Rethinking model scaling for convolutional neural networks". In: *Proc. Int. Conf. Mach. Learn.* 2019, pp. 6105–6114.

[131] Tian et al. "FCOS: Fully Convolutional One-Stage Object Detection". In: *Int. Conf. Comput. Vis.* 2019.

[132] Zhi Tian et al. "Decoders Matter for Semantic Segmentation: Data-Dependent Decoding Enables Flexible Feature Aggregation". In: *IEEE Conf. Comput. Vis. Pattern Recog.* 2019, pp. 3126–3135.

[133] Zhi Tian et al. "FCOS: Fully Convolutional One-Stage Object Detection". In: *Proc. IEEE Int. Conf. Comp. Vis.* (2019).

[134] Michael Treml et al. "Speeding up semantic segmentation for autonomous driving". In: *Proc. Workshop of Advances in Neural Inf. Process. Syst.* 2016.

[135] Yi-Hsuan Tsai, Ming-Hsuan Yang, and Michael J Black. "Video segmentation via object flow". In: *IEEE Conf. Comput. Vis. Pattern Recog.* 2016, pp. 3899–3908.

[136] Gregor Urban et al. "Do Deep Convolutional Nets Really Need to be Deep (Or Even Convolutional)?" In: *Proc. Int. Conf. Learn. Representations.* 2016.

[137] Guo-Hua Wang, Yifan Ge, and Jianxin Wu. "In Defense of Feature Mimicking for Knowledge Distillation". In: *arXiv preprint arXiv:2011.01424* (2020).

[138] Heng Wang, Zengchang Qin, and Tao Wan. "Text Generation Based on Generative Adversarial Nets with Latent Variables". In: *Proc. Pacific-Asia Conf. Knowledge discovery & data mining*. 2018, pp. 92–103.

[139] Tao Wang et al. "Distilling object detectors with fine-grained feature imitation." In: *IEEE Conf. Comput. Vis. Pattern Recog.* 2019, pp. 4933–4942.

[140] Yikai Wang et al. "Deep Multimodal Fusion by Channel Exchanging". In: *Adv. Neural Inform. Process. Syst.* 33 (2020).

[141] Yujiang Wang et al. "Face mask extraction in video sequence". In: *Int. J. Comput. Vis.* 127.6 (2019), pp. 625–641.

[142] Yukang Wang et al. "Intra-class Feature Variation Distillation for Semantic Segmentation". In: *Eur. Conf. Comput. Vis.* (2020).

[143] Yin Wei et al. "Enforcing geometric constraints of virtual normal for depth prediction". In: *Proc. IEEE Int. Conf. Comp. Vis.* (2019).

[144] Diana Wofk et al. "FastDepth: Fast Monocular Depth Estimation on Embedded Systems". In: *Int. Conf. on Robotics and Automation* (2019).

[145] Tete Xiao et al. "Unified Perceptual Parsing for Scene Understanding". In: *Proc. Eur. Conf. Comp. Vis.* 2018.

[146] Jiafeng Xie et al. "Improving fast segmentation with teacher-student learning". In: *Brit. Mach. Vis. Conf.* (2018).

[147] Jiafeng Xie et al. "Improving Fast Segmentation With Teacher-student Learning". In: *Proc. British Machine Vis. Conf.* (2018).

[148] Dan Xu et al. "Pad-net: Multi-tasks guided prediction-and-distillation network for simultaneous depth estimation and scene parsing". In: *IEEE Conf. Comput. Vis. Pattern Recog.* 2018, pp. 675–684.

[149] Yu-Syuan Xu et al. "Dynamic video segmentation network". In: *IEEE Conf. Comput. Vis. Pattern Recog.* 2018, pp. 6556–6565.

[150] Yongxin Yang and Timothy Hospedales. "Deep multi-task representation learning: A tensor factorisation approach". In: *Int. Conf. Learn. Represent.* 2017.

[151] Ze Yang et al. "Reppoints: Point set representation for object detection". In: *Int. Conf. Comput. Vis.* 2019, pp. 9657–9666.

[152] Chun-Han Yao, Chia-Yang Chang, and Shao-Yi Chien. "Occlusion-aware video temporal consistency". In: *ACM Int. Conf. Multimedia*. ACM. 2017, pp. 777–785.

[153] Wei Yin et al. "Enforcing geometric constraints of virtual normal for depth prediction". In: *Int. Conf. Comput. Vis.* 2019, pp. 5684–5693.

[154] Changqian Yu et al. "Bisenet: Bilateral segmentation network for real-time semantic segmentation". In: *Eur. Conf. Comput. Vis.* 2018, pp. 325–341.

[155] Changqian Yu et al. "Representative Graph Neural Network". In: *Eur. Conf. Comput. Vis.* (2020).

[156] Fisher Yu and Vladlen Koltun. "Multi-scale context aggregation by dilated convolutions". In: *Proc. Int. Conf. Learn. Representations* (2016).

[157] Lantao Yu et al. "SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient." In: *Proc. AAAI Conf. Artificial Intell.* 2017, pp. 2852–2858.

[158] Ruichi Yu et al. "NISP: Pruning Networks Using Neuron Importance Score Propagation". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[159] Yuhui Yuan, Xilin Chen, and Jingdong Wang. "Object-contextual representations for semantic segmentation". In: *arXiv preprint arXiv:1909.11065* (2019).

[160] Yuhui Yuan, Xilin Chen, and Jingdong Wang. "Object-Contextual Representations for Semantic Segmentation". In: (2020).

[161] Yuhui Yuan and Jingdong Wang. "Ocnet: Object context network for scene parsing". In: *arXiv preprint arXiv:1809.00916* (2018).

[162] Yuhui Yuan and Jingdong Wang. "OCNet: Object Context Network for Scene Parsing". In: *arXiv: Comp. Res. Repository*. Vol. abs/1809.00916. 2018.

[163] Kaiyu Yue, Jiangfan Deng, and Feng Zhou. "Matching Guided Distillation". In: *Eur. Conf. Comput. Vis.* (2020).

[164] Sergey Zagoruyko and Nikos Komodakis. "Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer". In: *Int. Conf. Learn. Represent.* 2017.

[165] Sergey Zagoruyko and Nikos Komodakis. "Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer". In: *Int. Conf. Learn. Represent.* (2017).

[166] Sergey Zagoruyko and Nikos Komodakis. "Paying More Attention to Attention: Improving the Performance of Convolutional Neural Networks via Attention Transfer". In: *Proc. Int. Conf. Learn. Representations* (2017).

[167] Han Zhang et al. "Self-Attention Generative Adversarial Networks". In: *arXiv: Comp. Res. Repository*. Vol. abs/1805.08318. 2018.

[168] Linfeng Zhang and Kaisheng. Ma. "Improve Object Detection with Feature-based Knowledge Distillation: Towards Accurate and Efficient Detectors." In: *Int. Conf. Learn. Represent.* 2021.

[169] Ting Zhang et al. "Interleaved Group Convolutions". In: *Proc. IEEE Int. Conf. Comp. Vis.* 2017, pp. 4383–4392.

[170] Xiangyu Zhang et al. "ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices". In: *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.* (2018).

[171] Ying Zhang et al. "Deep mutual learning". In: *IEEE Conf. Comput. Vis. Pattern Recog.* (2018).

[172] Zhenyu Zhang et al. "Joint Task-Recursive Learning for Semantic Segmentation and Depth Estimation". In: *Eur. Conf. Comput. Vis.* 2018, pp. 235–251.

[173] Hengshuang Zhao et al. "ICNet for Real-Time Semantic Segmentation on High-Resolution Images". In: *Eur. Conf. Comput. Vis.* (2018).

[174] Hengshuang Zhao et al. "Icnet for real-time semantic segmentation on high-resolution images". In: *Proc. Eur. Conf. Comp. Vis.* (2018).

[175] Hengshuang Zhao et al. "Psanet: Point-wise spatial attention network for scene parsing". In: *Proc. Eur. Conf. Comp. Vis.* 2018, pp. 267–283.

[176] Hengshuang Zhao et al. "Pyramid scene parsing network". In: *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.* 2017, pp. 2881–2890.

[177] Hengshuang Zhao1 et al. "Pyramid Scene Parsing Network". In: *IEEE Conf. Comput. Vis. Pattern Recog.* (2020).

[178] Zhao Zhong et al. "Practical block-wise neural network architecture generation". In: *IEEE Conf. Comput. Vis. Pattern Recog.* 2018, pp. 2423–2432.

[179] Bolei Zhou et al. "Scene Parsing through ADE20K Dataset". In: *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.* 2017.

[180] Bolei Zhou et al. "Scene parsing through ade20k dataset". In: *IEEE Conf. Comput. Vis. Pattern Recog.* (2017).

[181] Zaida Zhou et al. "Channel Distillation: Channel-Wise Attention for Knowledge Distillation". In: *arXiv: Comp. Res. Repository* abs/2006.01683 (2020).

[182] Xizhou Zhu et al. "Deep feature flow for video recognition". In: *IEEE Conf. Comput. Vis. Pattern Recog.* 2017, pp. 2349–2358.

[183] Xizhou Zhu et al. "Towards high performance video object detection". In: *IEEE Conf. Comput. Vis. Pattern Recog.* 2018, pp. 7210–7218.

[184] Yi Zhu et al. "Improving Semantic Segmentation via Video Propagation and Label Relaxation". In: *IEEE Conf. Comput. Vis. Pattern Recog.* 2019, pp. 8856–8865.

[185] Zhen Zhu et al. "Asymmetric Non-local Neural Networks for Semantic Segmentation". In: *Int. Conf. Comput. Vis.* (2019).

[186] Bohan Zhuang et al. "Towards effective low-bitwidth convolutional neural networks". In: *IEEE Conf. Comput. Vis. Pattern Recog.* 2018, pp. 7920–7928.

[187] Zhuangwei Zhuang et al. "Discrimination-aware channel pruning for deep neural networks". In: *Adv. Neural Inform. Process. Syst.* 2018, pp. 875–886.