Learning Dense Prediction: from Correspondence to Segmentation

Feihu Zhang
Exeter College
University of Oxford

A thesis submitted for the degree of
Doctor of Philosophy

Michaelmas 2022

To my beloved parents, who are always supporting me.

Declaration

I hereby declare that this thesis is submitted to the Department of Engineering Science, University of Oxford, in fulfilment of the requirements for the degree of Doctor of Philosophy. This thesis is entirely my own work, and except where otherwise stated, describes my own research.

December 2022

Feihu Zhang

Acknowledgements

My utmost gratitude goes to Prof. Phillip Torr and Prof. Victor Prisacariu for giving me the opportunity to join one of the best research labs and for leading me in the research projects. Without their unwavering support and creative instructions, I would not be able to finish this thesis.

I want to thank Prof. Xiaojuan Qi, Prof. Song Bai, Prof. Li Zhang, and other TVG members for their unreserved sharing of thoughts and research ideas. I also appreciate Dr Qizhu Li's valuable suggestions when writing this thesis. Furthermore, my heartfelt gratitude goes to Ms Joanna Zapisek, the TVG project manager, for always supporting and taking good care of us.

My deepest gratitude goes to Dr Ruigang Yang for his kind and generous support before and after I join Oxford. My eternal appreciation goes to Dr Stephan R. Richter, Dr René Ranftl, and Dr Vladlen Koltun for their insightful instructions and suggestions when I interned at Intel. I am also immensely grateful for Dr Oliver Woodford and many other collaborators. Without their unremitting efforts and contributions to the research projects, I would not be able to get my papers published.

I hope to express my sincere appreciation to Prof. Andrew Zisserman, Prof. Andrea Vedaldi and Prof. Tao Xiang for serving as the examiners and providing constructive and insightful suggestions to improve the quality of this thesis significantly.

I owe a debt of gratitude to all my families, who have made great sacrifices and been incredibly supportive throughout my life. They have always encouraged me to make the most of every opportunity I have had. Without their constant love and care, I would not have become the person I am today.

Last but not least, I would like to thank Baidu Inc. and Snap Inc. for their generous funding support to make this thesis possible.

Abstract

Dense prediction is the task of predicting a label for each pixel in the image. Given 3D data (point clouds or RGB-D images) as input, dense prediction can also be extended to 3D space and assign each 3D point/location a label. According to the label type, dense prediction can be mainly categorized as depth estimation, motion prediction, segmentation, and other related tasks. There are four major challenges for learning dense predictions: i) how to significantly improve the accuracy and resolve the ambiguous regions, ii) high memory and computational costs, iii) the dependency on a large amount of labeled data for training, and iv) the poor cross-domain generalization to novel datasets.

This integrated thesis focuses on dense prediction tasks, from correspondence estimation (stereo matching and optical flow) to 2D/3D semantic segmentation. Seven robust deep neural network models are proposed to achieve state-of-the-art accuracy, to realize effective training with just synthetic data or unlabeled real data, and to boost the cross-domain generalization to various unseen datasets.

For the first task, traditional 3D geometry constraints are embedded into end-to-end trainable stereo matching networks to achieve state-of-the-art accuracy on two stereo matching benchmarks (by publication date). Based on this work, a domain-invariant stereo matching network is proposed. It is trained on the synthetic data but outperforms many models fine-tuned on real data. For the second task, a Separable Flow

network is developed for optical flow estimation, which ranks the first on two standard optical flow benchmarks (by the time of publication). It's also one of the best methods for predicting optical flow on various unseen datasets. Moreover, research is also conducted on unsupervised pre-training and domain adaptation for semantic image segmentation. Finally, the 2D image segmentation knowledge is further leveraged for tackling 3D segmentation. The proposed 3D segmentation networks achieve the leading position on large-scale point-cloud segmentation benchmarks (at the time of publication).

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Artificial Intelligence (AI) aims at developing the theory and computer systems to perform tasks that normally require human intelligence, such as visual perception, speech recognition, decision-making, and translation between languages [1]. In recent years, we are increasingly enjoying the convenience brought by AI technologies. AI products, such as self-driving car, health monitoring and diagnosis, and face recognition on our mobile phones, have made our daily life more and more comfortable.

As an important field of AI, computer vision system seeks to understand and automate tasks that the human visual system can do [269]. Humans are endowed with an innate visual ability to understand and make sense of the surrounding environment with high robustness. With little conscious effort, we are able to recognise objects, perceive three-dimensional distance and shapes, and make forecasts about the changes in the near future. In contrast to the human vision system, computers capture visual information in the form of images which consist of discrete grids of pixel color and intensity. Computer vision systems derive meaningful information by processing and analyzing these digital images or videos, which plays a vital role in many intelligent systems. For example, self-driving

cars rely on cameras to capture surrounding information for perception, motion prediction, and making driving decisions.

1.1 Dense Prediction

Given one or multiple images as input, pixel-wise dense prediction produces a label for each pixel in the image. Such dense pixel labels provide accurate information for detailed understanding of the surrounding scenes in many computer vision systems. For example, in self-driving systems, driveable regions can be accurately estimated through dense prediction algorithms.

According the type of the labels, dense prediction can be mainly categorized as depth estimation, motion prediction, segmentation, and other related tasks. Dense prediction can be widely used in many applications or industrial products, including but not limited to 3D reconstruction, remote sensing, image/video editing, robotics, and autonomous driving cars.

Dense prediction has been studied for many years. Many traditional algorithms [93, 97, 20, 59] have been developed to solve dense prediction problems. Then, deep neural networks [177, 64, 51, 219] were introduced to dense prediction applications and significantly improved the accuracy and robustness of the methods in solving dense prediction tasks. Learning end-to-end deep neural networks for solving dense prediction tasks [367, 37, 111, 274, 32, 128] has gradually superseded the traditional hand-craft designs in the dense prediction tasks.

1.2 Dense Prediction Tasks

This thesis focuses on three important dense prediction tasks: stereo matching, optical flow, and semantic segmentation. Among them, stereo matching and optical flow take a pair of images as input and output the dense pixel-to-pixel

correspondences across the two image views. The 2D image segmentation task takes a single image as input and outputs per-pixel category labels. In contrast, 3D point-cloud segmentation accepts a set of 3D points as input and predicts point-wise category labels.

Stereo Matching Stereo reconstruction is a fundamental problem in computer vision, robotics, and autonomous driving. It aims to estimate 3D geometry by computing per-pixel disparity between matching pixels in a stereo image pair. Stereo matching technique is fundamental in many real-world applications such as 3D reconstruction, remote sensing, and robotics.

Optical Flow Optical flow is the task of estimating per-pixel 2D motion between two images or video frames. This low-level vision task is a fundamental building block of many higher-level tasks, such as object tracking, scene reconstruction, and video editing and compression.

Semantic Segmentation Semantic segmentation [177, 59] that associates a label or category with every pixel in an image is used to recognize a collection of pixels that form distinct categories. Semantic segmentation of images is widely used in many industrial products, such as autonomous cars, medical diagnosis, and defect inspection. Given 3D data (*e.g.* point cloud or RGBD image) as input, the dense prediction can also be extended for 3D dense prediction [272] that assigns each 3D point/location a specific label. 3D Semantic segmentation that predicts a category label for each 3D point/location is widely applicable to many scenarios, including remote sensing, AR/VR, robotics, and self-driving cars.

1.3 Challenges in Dense Prediction

Despite the success of deep neural networks in solving dense prediction tasks, there are still many challenges in designing dense-prediction neural networks. This thesis focuses on four major challenges (targets):

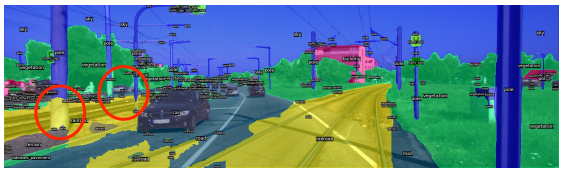
i) challenging regions, such as thin structures, small objects, occlusions and reflective regions,

ii) high memory and computational costs,

iii) dependency on a large amount of labeled data for training,

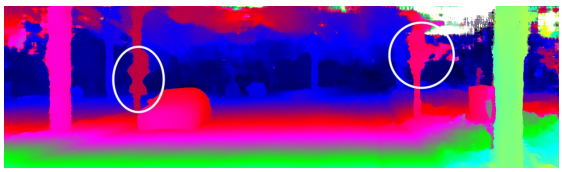
iv) poor cross-domain generalization to various unseen datasets.

(a) Input reference image



(b) Segmentation results of [142]

(c) Optical flow results of [268]



(d) Stereo matching results of [128]

Figure 1.1: Problem illustration of the poor dense prediction results in challenging regions of thin structures, small objects, occlusions and reflective regions (as highlighted by the red or white circles). (a) Input image of a driving scene. (b) Segmentation results of [142]. There are many wrong predictions for thin structures or small objects. (c) Optical flow results of [268]. It produces many ambiguous motion predictions in occlusions, image borders and thin objects. (d) Stereo depth estimation of [128]. Depth errors appear in the large smooth regions and the thin structures.

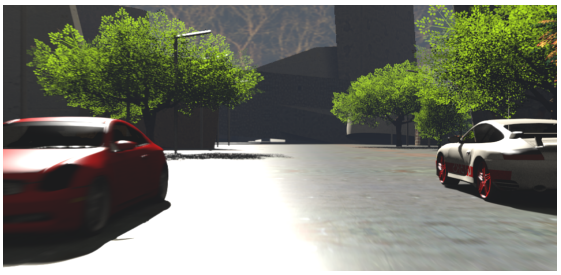
Challenging thin structures, small objects, occlusions and reflective regions. Existing dense prediction networks [128, 32, 369, 274, 38, 367] can produce good

predictions in most of the image areas (*e.g*. road and wall of the scenes). However, they still fail in many difficult but important regions (*e.g*. thin structures or small objects, occluded or reflective regions) and tend to assign them a false background label (Fig. 1.1 (b)). This is caused by 1) the imbalance between these thin structures and the large background areas, and 2) the smoothing and loss of fine details after a series of convolutional, downsampling, and upsampling layers. Furthermore, in occluded and reflective regions, missing pixel-matching information also leads to errors in dense correspondence estimation (Fig. 1.1 (c)$\sim$(d)). In order to predict accurate labels for these difficult regions or objects, new neural network modules or architectures must take full advantage of the adaptive pyramidal features and geometry information for achieving more accurate dense prediction results.

High memory and computational costs. Different from the visual classification networks [89, 256], dense prediction neural network models need to preserve a high-resolution feature map during the feature encoding and decoding in order to predict a label for each pixel in an image. In processing high-resolution images, the high memory and computational costs are crucial limitations in developing robust dense prediction models. Furthermore, in tasks of multi-view dense correspondence estimation (stereo matching and optical flow) and the 3D segmentation, many state-of-the-art methods [128, 274, 99] learn a 3D or even 4D feature representation to predict dense labels. The extremely high memory and computational costs in constructing and processing such high-dimensional feature tensors have become a bottleneck problem in dense correspondence estimation and 3D segmentation.

Dependency on plenty of labeled training data. The success of training existing dense prediction networks relies on massive amounts of manually labeled data. However, labeling pixels/3D points is extremely expensive and difficult. Without enough labeled data for training, it's impossible to train accurate neural network

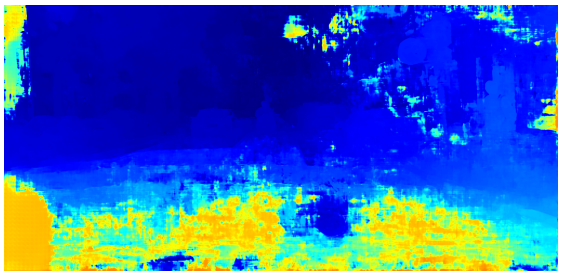
models for dense prediction tasks. This thesis utilizes two strategies to resolve this issue. Firstly, synthetic data are effectively used in the proposed methods. Synthetic data can be easily and cheaply generated. It can also produce more accurate labels and avoid labeling errors or biases introduced by human annotators. Secondly, this thesis proposes unsupervised methods for the semantic image segmentation task that leverage unlabeled real data for neural network pre-training and domain adaptation. The proposed method can be pre-trained on unlabeled data and then fine-tuned on synthetic data (without using any real-data labels) to achieve results competitive with those of models trained or fine-tuned on real data.



(a) Example of the training data

(b) Example of the test data

(c) Optical flow results of [274]



(d) Stereo matching results of [32]

Figure 1.2: Problem illustration of the poor cross-domain generalization abilities. Models are trained on the synthetic dataset [187] (a) and tested on the real images [58] (b). There are significant domain differences between the real test scenes (b) and the source training data (a). (c) Optical flow estimation results of [274]. (d) Stereo depth estimation of [32]. There is a large area of wrong predictions for both stereo matching and optical flow due to the domain differences.

Poor cross-domain generalization. The neural network models trained on one specific dataset cannot generalize well to unseen data without fine-tuning or

adaptation. Namely, when tested on a new dataset, the accuracy drops significantly. The poor generalization abilities are usually caused by domain differences (such as different distributions of color, illumination, contrast, texture, and object contents). For example, synthetic data can be cheaply generated with accurate labels to train dense prediction networks. While the synthetic data have significant domain differences (as illustrated in Fig. 1.2 (a)∼(b)). Models trained on synthetic data produce poor predictions on the real datasets (Fig. 1.2 (c)∼(d)).

Many existing methods have explored ideas for solving the above challenges. To improve the accuracy in challenging regions, interpolation is widely used on the low-resolution feature representation to restore the details in the dense prediction results [246, 32]. Smoothness penalties are implemented in many traditional dense matching algorithms [19, 325, 11, 352] to improve the performance in occluded and reflective regions for optical flow and stereo matching. However, these usually lead to blur predictions around object edges.

Existing deep neural network models compress the 3D space or representation into 2D representation to reduce memory and computational costs. For example, [187, 64, 115, 268] simply utilize 2D convolutional networks to learn dense matching results. Without 3D geometry constraints, such methods take more training time for convergence and have poor generalization abilities when testing on novel datasets. [330, 264, 220] project the 3D point cloud space into 2D image-like grids and use 2D convolutions for more efficient processing. Such space compression leads to significant loss of point-wise information and will produce ambiguous 3D segmentation results.

Self-supervised learning has been studied to avoid the dependency on manually annotated data. For example, many contrastive learning works adopt instance discrimination [86, 41] for pre-training on unlabeled images and can be transferred to image classification and object detection via fine-tuning. However, dense pixel-

level segmentation poses unique challenges. Few works have studied unsupervised learning or pre-training for accurate dense image labeling tasks.

Moreover, to successfully apply the pre-trained dense prediction networks to other unseen datasets, domain adaptation and transfer learning methods (*e.g.* [282, 82, 22]) attempt to transfer or adapt the pre-trained models from one source domain to another new domain. A large amount of images from the new domain are required for the adaptation. However, these cannot be easily obtained in many real scenarios. Moreover, it is impossible to adapt the model one by one to every domain, which costs plenty of time and resources.

1.4 Approach

To address the aforementioned issues in dense prediction, we explore different strategies and develop various effective deep neural network models for stereo matching, optical flow, and semantic segmentation tasks.

Firstly, to achieve better dense prediction accuracy, we mainly develop two novel techniques. 1) We propose new neural network modules to construct more effective dense prediction networks. New local and non-local matching aggregation modules that aim at capturing local and the whole-image cost dependencies are proposed respectively for the dense correspondence estimation (stereo matching and optical flow). A deep fusion network module with a unique voxel-based "mini-PointNet" point cloud representation is designed to learn more accurate point-wise predictions in 3D semantic segmentation. 2) We utilize unsupervised pre-training strategies to obtain a good initialization for dense prediction networks. For example, we introduce an unsupervised contrastive pre-training pipeline to boost the performance on several semantic segmentation datasets (Cityscapes [58], ADE20K [376] and NYUv2 [254]) and utilize supervised pre-training on synthetic

datasets (FlyingThing3D [187]) to improve the accuracy of the stereo matching and optical flow networks.

Secondly, to reduce the memory and computational costs of the dense prediction networks, we 1) propose a more effective matching cost aggregation module to replace the use of a large number of 3D convolutional layers, 2) split the high-dimensional 2D optical flow matching into two simple 1D matching problems which can be implemented with low memory and computational costs and 3) design novel sparse and memory-efficient data structures to represent the irregular 3D points for accurate 3D segmentation.

Furthermore, to avoid the reliance on manual labels for training dense prediction networks, we propose to use unlabeled real data and high-quality synthetic data (with synthetic labels) for pre-training or fine-tuning the dense prediction models. For example, in semantic segmentation, we show that the proposed contrastive pre-training can take full advantage of the unlabeled real images (Cityscapes [58]) and the synthetic scenes [234] to develop robust models for driving scene parsing. Moreover, for dense correspondence estimation (stereo matching and optical flow), we use synthetic data to train accurate dense matching models that achieve similar or better accuracy when comapared to the deep neural network models fine-tuned on real data [268, 64, 51].

Finally, to address the poor cross-domain generalization, we design two effective strategies. i) We demonstrate that learning more robust non-local structural and geometric representations can make the deep neural network models perform better when there are significant domain shifts (*e.g.* in colors, texture styles and local contrasts). This is especially effective in stereo matching and optical flow. ii) We show the proposed contrastive pre-training strategy can make the deep neural network models generalize well to unseen datasets. The contrastive pre-training pipeline can take unlabeled real data as input for learning more representative

and generalizable features that can avoid overfitting on one specific dataset when training semantic segmentation models.

1.5 Contributions

This thesis focuses on applying deep neural networks to 2D/3D dense prediction tasks, including stereo matching, optical flow, semantic image segmentation, and 3D point cloud segmentation. The main research contributions are summarised below, with each part corresponding to 1$\sim$2 published papers.

1.5.1 Stereo Matching

Chapters 3 and 4 describe two proposed stereo matching networks which can be used for robust and accurate disparity/depth estimation using binocular vision.

Chapter 3 focuses on matching cost aggregation in stereo matching, which is crucial in deep neural network models in order for them to accurately estimate disparities. We propose two novel neural net layers aimed at capturing local and whole-image cost dependencies, respectively. The first is a semi-global aggregation layer which is a differentiable approximation of the semi-global matching, and the second is the locally guided aggregation layer which follows a traditional cost filtering strategy to refine thin structures. These two layers can be used to replace the widely used 3D convolutional layer which is computationally costly and memory-consuming as it has cubic computational/memory complexity. In the experiments, we show that nets with a two-layer guided aggregation block easily outperform the state-of-the-art GC-Net [128], which has nineteen 3D convolutional layers.

We observed that GA-Net performs better than existing stereo matching networks [128, 32] in the cross-domain generalization settings (from synthetic to real).

However, it still has difficulties in generalizing to new, unseen environments due to significant domain differences, such as in color, illumination, contrast, and texture. We aim at designing a domain-invariant stereo matching network that generalizes well to unseen data. To achieve this goal, Chapter 4 proposes i) a novel "domain normalization" approach that regularizes the distribution of learned feature representations to make them invariant to domain differences, and ii) an end-to-end trainable structure-preserving graph-based filter for extracting robust structural and geometric representations that can further enhance domain-invariant generalizations. When trained on synthetic data and generalized to real test sets, the proposed model performs significantly better than all state-of-the-art models. It even outperforms some deep neural network models (*e.g.* MC-CNN [352] and DispNet [187]) that are fine-tuned with test-domain data.

1.5.2 Optical Flow

Chapter 5 proposes a new, separable, cost-volume module, which can be plugged into existing cost-volume-based optical flow frameworks, with two key innovations that address existing challenges for optical flow. The first contribution is to separate the problem of estimating 2D motion into two independent 1D problems, namely, the horizontal and vertical motions, by compressing the 4D cost volume into two smaller 3D volumes using a self-adaptive separation layer. This factored representation significantly reduces the memory and computing resources required to learn the cost volumes, making them linear in the range of motion, without a loss in accuracy. Moreover, it enables the second of our innovations: the use of non-local aggregation layers to learn a refined cost volume. Such layers have previously been used for 1D stereo problems (Chapters 3 and 4), where they improve both accuracy in ambiguous regions, and cross-domain generalization. We apply them to optical flow for the first time, learning cost volumes with non-local, prior knowledge via a one-step

motion regression that is able to predict a low-resolution (*i.e*. 1/8), but high-quality motion. This prediction also serves as a better initial input to the interpolation and refinement module.

The Separable Flow module are trained and evaluated on the standard Sintel and KITTI optical flow datasets. By the date of publication, it achieved the best accuracy among all published optical flow methods on the two standard optical flow benchmarks. Moreover, in the cross-domain case of training on synthetic data and testing on real data (*i.e*. KITTI), the method improves the state-of-the-art by a great margin, even outperforming some DNN models (*e.g*. FlowNet2 [115] and PWC-Net [268]) fine-tuned on the target KITTI scenes.

1.5.3 Semantic Image Segmentation

In Chapter 6, we present an approach to contrastive representation learning for semantic segmentation. Our approach leverages the representational power of existing feature extractors to find corresponding regions across images. These cross-image correspondences are used as auxiliary labels to guide the pixel-level selection of positive and negative samples for more effective contrastive learning in semantic segmentation. We show that auxiliary labels can be generated from a variety of feature extractors, ranging from image classification networks that have been trained using unsupervised contrastive learning to segmentation models that have been trained on a small amount of labeled data or synthetic data. We additionally introduce a novel metric for rapidly judging the quality of a given auxiliary-labeling strategy, and empirically analyze various factors that influence the performance of contrastive learning for semantic segmentation. We demonstrate the effectiveness of our method both in the low-data as well as in the high-data regime on various datasets. Experiments show that contrastive learning with the proposed auxiliary-labeling approach consistently boosts semantic segmentation

accuracy when compared to standard ImageNet pre-training and outperforms existing approaches of contrastive and semi-supervised semantic segmentation.

Furthermore, we also introduce contrastive learning to the problem of unsupervised domain adaptation for semantic image segmentation (Chapter 7). The proposed method leverages the pixel category information from the synthetic data and the image feature distributions from the unlabeled real data for joint training and adaptation. We discover both in-domain contrastive pairs as well as cross-domain contrastive samples to learn the domain-invariant representation. Our framework utilizes pseudo-labels to guide the selection of contrastive pairs. The pseudo labels are generated by a label expansion strategy that specializes in discovering hard-class contrastive pairs. The proposed method consistently outperforms state-of-the-art methods for domain adaptation across synthetic and real datasets. It significantly improves the accuracy on hard classes that are small or suffer from significant class-specific domain shifts between the source (synthetic) and the target (real) domain.

1.5.4 3D Point Cloud Segmentation

Chapter 8 proposes a deep fusion network architecture (FusionNet) with a unique voxel-based "mini-PointNet" point cloud representation and a new feature aggregation module (fusion module) for large-scale 3D semantic segmentation. The method learns more accurate point-wise predictions when compared to voxel-based convolutional networks. It also realizes more effective feature aggregations with lower memory and computational complexity for large-scale point cloud segmentation, when compared to the popular point-wise convolutions. In the experiments, FusionNet achieves state-of-the-art accuracy on the large-scale Semantic KITTI benchmark.

Chapter 9 describes a robust baseline method for the segmentation of large-scale

LiDAR point clouds. We propose a novel dense feature encoding technique for the localization and segmentation of small, far-away objects and design effective strategies for handling severe class imbalances. Since there is no public dataset for the study of LiDAR instance segmentation, we also build a new synthetic LiDAR point cloud dataset which contains both precise 3D bounding boxes and point-wise labels for both instance and semantic segmentation. The dataset is about 3$\sim$20 times as large as other existing LiDAR datasets.

1.6 Thesis outline

The remaining chapters of this thesis are organized as follows:

Chapter 2 presents a review of relevant literature in the area of stereo matching, optical flow, 2D image and 3D point cloud segmentation.

Chapter 3 and **Chapter 4** describe the proposed deep stereo matching networks that can achieve more accurate stereo reconstruction and better cross-domain generalization performance.

Chapter 5 seeks to extend the effective strategies and other discoveries from the 1D stereo matching problem to the 2D optical flow estimation.

Chapter 6 and **Chapter 7** introduce contrastive learning methods to address the pre-training and domain adaptation for the semantic image segmentation task.

Furthermore, **Chapter 8** and **Chapter 9** further introduce the knowledge of 2D image segmentation to 3D segmentation by proposing more accurate 3D segmentation networks that are particularly effective for 3D point clouds and RGB-D data.

Finally, in **Chapter 10**, we conclude this thesis by summarising the key contributions and discussing questions that remain unanswered and potential future work.

1.7 Publications

The following publications form the individual chapters of this thesis, including six published conference papers and one unpublished arXiv pre-print.

Chapter 3

> Feihu Zhang, Victor Prisacariu, Ruigang Yang, Philip H.S. Torr. GANet: Guided Aggregation Net for End-to-end Stereo matching. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, (oral & best paper finalist)*.

Chapter 4

> Feihu Zhang, Xiaojuan Qi, Ruigang Yang, Victor Prisacariu, Benjamin Wah, Philip Torr. Domain-invariant Stereo Matching Newtorks. *European Conference on Computer Vision (ECCV), 2020, (oral)*.

Chapter 5

> Feihu Zhang, Oliver Woodford, Victor Prisacariu, Philip Torr. Separable Flow: Learning Motion Cost Volumes for Optical Flow Estimation. *IEEE/CVF International Conference on Computer Vision (ICCV), 2021*.

Chapter 6

> Feihu Zhang, Philip Torr, René Ranftl, Stephan R. Richter. Looking Beyond Single Images for Contrastive Semantic Segmentation. *Conference on Neural Information Processing Systems (NeurIPS), 2021*.

Chapter 7

Feihu Zhang, Vladlen Koltun, Philip H. S. Torr, René Ranftl, Stephan R. Richter. Unsupervised Contrastive Domain Adaptation for Semantic Segmentation. *arXiv preprint, 2022*.

Chapter 8

Feihu Zhang, Jin Fang, Benjamin Wah, Philip Torr. Deep FusionNet for Point Cloud Semantic Segmentation. *European Conference on Computer Vision (ECCV), 2020*.

Chapter 9

Feihu Zhang, Chenye Guan, Jin Fang, Song Bai, Ruigang Yang, Philip Torr, Victor Prisacariu. Instance Segmentation of LiDAR Point Clouds. *International Conference on Robotics and Automation (ICRA), 2020*.

Chapter 2

Background

Dense prediction has been studied for decades, with many traditional algorithms [93, 97, 20, 59] proposed. Early neural networks with backpropagation mechanism have also been proposed since 1980s [240, 148]. However, until recently, deep neural networks [177, 64, 51, 219] start to be successfully introduced to dense prediction tasks, which significantly improve the accuracy and robustness of the dense prediction models. Since then, training end-to-end deep neural networks for dense prediction tasks [367, 37, 111, 274, 32, 128] has gradually superseded the traditional hand-craft designs in the dense prediction tasks.

This chapter briefly introduces the advances and developments for deep neural networks in solving dense prediction problems. We mainly focus on the three major dense prediction tasks: stereo matching, optical flow and 2D/3D segmentation. In each of the tasks, we concentrate on four aspects: i) performance in the challenging thin structures, small objects, occlusions and reflective regions, ii) reducing the memory and computational costs of dense prediction networks, iii) addressing the reliance on plenty of manually labeled data in training deep neural networks, and iv) boosting the cross-domain generalization abilities to novel dataset.

2.1 Stereo Matching

Stereo reconstruction aims to estimate 3D geometry by computing disparities between matching pixels in a stereo image pair. It is challenging due to a variety of real-world problems, such as occlusions, large textureless areas, reflective surfaces, thin structures, and repetitive textures (*challenge i*). Traditionally, stereo reconstruction can be decomposed into three important steps: feature extraction (for matching cost computation), matching cost aggregation, and disparity prediction [243, 93]. Recently, deep neural network has been applied for disparity estimation. Many powerful stereo matching networks have been proposed and gradually replaced the traditional hand-craft stereo matching algorithms.

2.1.1 Deep Neural Networks for Stereo Matching

Deep neural networks are first introduced to learn image features and compute patch-wise similarity scores in [351, 358, 51, 71]. They use traditional cost aggregation and disparity computation methods [93, 98] to compute the disparity maps. Limited by the abilities of the traditional matching cost aggregation step, these approaches need many post-refinement ways and often produce wrong predictions in occluded regions, thin structures, large textureless/reflective regions, and around object edges (*challenge i*). There are also some other methods tries to improve the performance of the traditional cost aggregation. For example, SGM-Nets [248] predicts the penalty-parameters for SGM [93] using a neural net, whereas Schönberger *et al*. [244] learn to fuse proposals by optimization in stereo matching.

Recently, end-to-end deep neural network models have been proposed for stereo matching. Mayer *et al*. create a large synthetic dataset to train end-to-end deep neural network for disparity estimation (*e.g*. DispNet) [187]. Pang *et al*. [207] first estimate and then refine the disparity maps by building a two-stage convolutional

neural network. Many other methods, including EdgeStereo [261], HD^3 [349], iResNet [162], SegStereo [333], and MADNet [282], create the color or feature correlations between the left and right views for disparity estimation.

GC-Net [128] incorporates the feature extraction, matching cost aggregation, and disparity estimation into a single end-to-end deep neural model. It constructs a 3D cost volume and uses 3D convolutional layers for cost aggregations. PSMNet [32] uses pyramid feature extraction and a stacked hourglass block [202] with twenty-five 3D convolutional layers to improve the GC-Net further. Similarly, AnyNet [305], StereoNet [129] and EMCUA [203] all utilize a similarity cost as the third dimension to build the cost volume in which the real geometric context is maintained. However, the 3D cost volume and 3D convolutional layer take plenty of memory and computational costs (*challenge ii*).

The common feature of these neural network models is that they all require a large number of training samples with ground-truth depths. These depth ground truths are usually difficult to achieve. Moreover, a model trained on one dataset cannot generalize well to new datasets without fine-tuning or retraining.

2.1.2 Self-supervised Learning and Adaptation in Stereo Matching

Many self-supervised stereo matching and domain adaptation methods have been proposed to address *challenge iii* – the reliance on plenty of manually labeled data in training.

Self-supervised Learning: Training stereo matching networks in an unsupervised manner relies on image reconstruction losses that warp the left and right views [377, 374]. However, they cannot solve the occlusions and reflective regions where there is no correspondence between the views. Moreover, they always overfit one specific dataset and cannot generalize well to other novel datasets.

Domain Adaptation: Some methods pre-train the models on synthetic data and then explore the domain knowledge to adapt [82, 208] for a new domain. Many online or offline adaptation methods [281, 282, 280, 216] have been proposed. The offline adaptation requires re-training on the target dataset before application. MADNet [282] proposes to adapt the pre-trained model online and in real-time. However, it has poor accuracy even after the adaptation. Moreover, all existing domain adaptation approaches need a large number of stereo pairs from the target domain for adaptation. It's impossible to adapt the model to every domain, which costs plenty of time and resources. Also, these stereo images cannot be easily obtained in many real scenarios. Thus, we need a reliable disparity estimation approach that can be directly applied to various scenes without re-training or adaptation.

2.1.3 Cross-domain Generalization

In contrast to domain adaptation, *challenge iv* – domain generalization [14, 76] is a much harder problem that assumes no access to target information for adaptation or fine-tuning. The ideas of domain-invariant feature learning have been widely explored. Many approaches focus on learning invariant features from multiple source domains [196, 76, 151]. Recent methods [14, 152] utilize meta-learning and take variations from multiple source domains to generalize to novel test distributions. Choy *et al*. [57] develop a universal feature learning framework for visual correspondences using deep metric learning. In contrast to these approaches, there are methods that try to improve the batch or instance normalization layers to improve the generalization and robustness [199, 206].

For stereo matching, work is seldom done to improve the generalization ability of the deep neural network models, especially for developing the domain-invariant stereo matching networks.

2.1.4 Summary of Stereo Matching

As discussed above, stereo matching task typically suffers from all four challenges. While state-of-the-art methods have limitations in effectively solving them.

In Chapter 3, we propose more effective matching cost aggregation layers and novel design of stereo matching networks to address *challenges i) and ii)*. The proposed guided aggregation net significantly improves the disparity estimation results in the challenging thin structures, small objects, occlusions and reflective regions. It considerably increases accuracy, while decreasing both memory and computational costs.

Furthermore, in Chapter 4, we propose domain-invariant networks to solve *challenges iii) and iv)*. We leverage synthetic data that can be easily generated to replace the expensive real stereo dataset in training our stereo matching model. More importantly, the model trained on one dataset (*e.g*. synthetic data) can be easily generalized to other unseen datasets to achieve accurate disparity estimation results. There is no need for any re-training, fine-tuning or adaptation.

2.2 Optical Flow

Optical flow is the task of estimating per-pixel 2D motion between two images or video frames. This low-level vision task is a fundamental building block of many higher-level tasks, such as object tracking, scene reconstruction, video editing and compression. This section reviews prior optical flow work, including traditional approaches and learning-based flow networks.

2.2.1 Traditional Approaches

There are many traditional optical flow methods rely on local filtering [98], interpolation [232, 102, 342], nearest neighbor search [11, 246, 186, 103, 182] or dense inverse

search [137]. These methods are very sensitive to local variations since they cannot capture non-local dependencies. Other methods [97, 20, 350, 310, 23, 223, 226] uses gradient-based solvers to optimizes global energy functions that consist of a local matching cost data term and an MRF-based smoothness regularization term. Furthermore, discrete solvers are also introduced in [192, 40, 324] to find more globally optimal solutions. However, large motion ranges require a huge search space, and each pixel must be paired with thousands of discrete correspondences to find the best matches. To address this issue, Menez *et al*. [192] prune the search space using feature descriptors and optimize using message passing, whereas Chen *et al*. [40] use a distance transform to solve the global optimization problem over the entire search space.

2.2.2 Deep Neural Networks for Optical Flow

Recently, a multitude of deep neural networks have been proposed to compute optical flow for videos or a pair of frames. Many different strategies have been introduced, including robust loss functions [73, 12], occlusion handling [369], feature representations [247, 358], uncertainty estimation [114], lightweight architecture[111], refinement/interpolation [386, 113, 274], data resampling [15], and motion estimation in dark scenes [372].

Several works jointly learn segmentation and optical flow [249, 9, 316, 52, 316], segmenting the image into objects or backgrounds and computing motion depending on the region type. Coarse-to-fine processing has emerged as a popular ingredient in many recent works [268, 349, 224, 111, 112, 113, 332, 96, 15, 369].

Among these methods, explicit cost volumes appear frequently [98, 268, 274, 110, 96, 295, 332, 178], storing the data matching costs for each pixel's potential correspondences, and thus playing an important role in generating accurate flow fields. For example, PWC-Net [268] develops a DNN model using image pyramids,

warping, and cost volumes. Xiao *et al*. [319] learn cost volumes using the Cayley representation, but without effective cost aggregations. Hui *et al*. [110] address the ambiguous matching challenge by improving the cost volume through an adaptive modulation prior, exploiting local flow consistency. Hofinger *et al*. [96] improve the cost volume construction process via a sampling-based strategy that revises the gradient flow across pyramid levels. Wang *et al*. [295] reshape a 4D cost volume into 3D via a displacement-aware projection (DAP) layer, learning the high-dimensional cost volume with low-dimensional convolutions. However, it can only process a fixed and small displacement range (*e.g*. -3,..,3). Yang *et al*. [332] propose a 5D volumetric encoder-decoder architecture with separable volumetric filtering.

There are also some other cost-volume based methods which have issues of memory and computational costs (*challenge ii*). Xu *et al*. [324] construct a 4D cost volume using DNN features and apply improved semi-global matching [93] for cost aggregations. This strategy is impractical for end-to-end training of DNNs, since the cost aggregation step is not differentiable, and incurs huge memory and computational costs. The current state-of-the-art optical flow model, RAFT [274] also builds multi-scale 4D correlation volumes for all pairs of pixels. However, limited by its huge memory and computational costs, RAFT does not apply any cost aggregation to the 4D volume.

2.2.3 Self-supervised Optical Flow

Self-supervised optical flow networks [343, 307, 118, 170, 125, 375, 168, 116] have also been explored to address *challenge iii*. However, these self-supervised flow networks heavily rely on the feature or color correspondences between the image pairs. These correspondences are unavailable in the image border and occlusion regions, leading to poor estimations. As a result, they cannot achieve competitive accuracy as those in supervised training.

2.2.4 Datasets and Domain Generalization

It's extremely difficult to achieve optical flow ground truth of the real image pairs. Existing real optical flow datasets [13, 136, 191] all relies on accurate laser scan to achieve 3D reconstruction and correspondence. The datasets in outdoor scenes [136, 191] also need human labor to remove label outliers manually. Thus, all these real datasets are limited to a small scale (20$\sim$1K image pairs) and are collected in a limited number of scenes. Models trained on these real datasets do not have good generalization abilities and can not generalize to different application scenarios (*challenge iv*).

State-of-the-art optical flow algorithms [268, 274] introduces the large-scale synthetic datasets [187, 25] for pre-training or joint training to improve the models' accuracy and generalizations. These synthetic datasets can be easily generated with more accurate correspondence labels. However, these synthetic data have significant domain shifts compared with real data. Training on these synthetic data does not have good generalizations to real scenes.

Teed *et al*. [274] proposes the recurrent transformer architecture for coarse-to-fine optical flow estimation and uses the mixture of real and synthetic datasets for training which has greatly improved the generalization abilities to unseen real data. In contrast to this method, we propose a separable flow model that can learn and refine a full-range matching cost volume over the whole motion space using non-local aggregations. As a result, our separable flow model achieves higher accuracy in the specific dataset and improves the cross-domain generalization abilities to novel scenes/datasets.

2.2.5 Summary of Optical Flow

Optical flow task naturally has difficulties in accurate motion estimation of large occlusion regions (*challenge i*). Many state-of-the-art methods create 4D cost volumes

to achieve accurate predictions of large motions. But, they incur high memory and computational costs (*challenge ii*). Moreover, all these existing optical flow networks require accurate motion labels of real scenes in training or fine-tuning (*challenge iii*).

In this thesis, Chapter 5 proposes a new optical flow network that effectively solves the four challenges in motion estimation. It simplifies the dense matching mechanism to predict more accurate motion results while reducing the memory cost. We can train it on synthetic data instead of real labeled images. Moreover, it can be easily generalized to novel datasets to achieve accurate motion prediction without re-training or adaptation.

2.3 Semantic Image Segmentation

Semantic image segmentation is the task of dense per-pixel classification of the content of an input image and is one of the fundamental problems of computer vision. Training deep neural networks for semantic segmentation typically requires a large amount of densely annotated images. However, it's extremely laborious and costly to produce such dense annotations for an adequate number of images at sufficient quality (*challenge iii)*.

Self-supervised learning techniques offer the possibility to significantly reduce the manual labeling effort that is required to train high-performance machine learning models. Furthermore, given synthetic source domain data and the unlabeled target real data, unsupervised domain adaptation aims at mitigating these performance drops by augmenting existing labeled training data from a source domain with unlabeled images from the target domain. Synthetic data can be easily generated without human labor, and unlabeled real images can be easily collected. Thus, unsupervised learning and adaptation can fundamentally resolve the dependency on manual data labels.

This section introduces the related research on unsupervised representation learning and domain adaptation for semantic segmentation.

2.3.1 Unsupervised Contrastive Learning

Contrastive learning [86, 41] has recently emerged as a promising technique for self-supervised representation learning and demonstrated impressive performance compared to fully supervised methods in a wide variety of tasks [86, 44, 41, 42, 91]. Contrastive learning leverages large amounts of unlabeled data for augmenting and extending existing labeled datasets without costly manual annotation. Similar visual concepts can be mapped to nearby representations in latent space, whereas dissimilar concepts will be pushed further apart. The effectiveness of contrastive learning heavily relies on the selection of similar and dissimilar samples.

Many existing works adopt instance discrimination [86, 41], where positive samples are generated from the same image via different transformations (views). All other images available for training represent potential negative samples. Recent work investigated different strategies to generate more informative positive and negative pairs, including mutual view selection[278, 277], hard mining[236, 126, 301], adversarial perturbations[132, 94, 100, 124], and using large mini-batches or a momentum memory bank [86, 194, 315].

The instance-level contrastive learning pre-training can be transferred to image classification and object detection via fine-tuning and demonstrated impressive performance. However, dense pixel-level segmentation poses unique challenges. Few works have studied contrastive learning in this dense image labeling task.

Contrastive learning for image segmentation. Recent works adapted contrastive learning to dense prediction tasks such as semantic segmentation [31, 304, 322, 383]. DenseCL applied the contrastive objective to the pixel level [304]. ReSim learned

spatial features from sliding windows [318]. Xie *et al*. generated views by smoothing over nearby feature representations to aggregate local context [322]. Chen *et al*. exploited foreground/backward masks [35] while Van Gansbeke *et al*. leveraged object masks from pre-trained saliency detectors to group samples [289]. All of the approaches mentioned above generate positive views from the same image. This limits the diversity of positive samples severely, as instances of the same category in different images cannot be paired. In contrast to existing works, our auxiliary-labeling strategy (Chapter 6) allows us to establish positive correspondences across images and pair more diverse examples of the same concept.

2.3.2 Unsupervised Domain Adaptation for Segmentation

Adversarial training for adaptation. In an adversarial training, a discriminator is trained to distinguish images [108, 379], intermediate representations [48, 50, 308, 294], or predicted labels [284, 285] from different domains. The discriminator provides a supervisory signal for aligning the distributions of different domains. A line of work is to train the category-aware discriminator for more fine-grained supervision [48, 294].

Others [49, 285] leverage local region or patch invariances for adversarial feature alignment. Style or contextual transfer [198, 344, 335], semantic image synthesis [334] and non-transferable features [33, 63] are utilized for domain adaptation.

Moreover, SIM aligns the feature representation from stuff and things of the target domain to specific samples in the source domain [308]. ADVENT finds that the entropy of predictions is higher on the target domain and introduces an entropy minimization loss [291]. CrDoCo trains separate networks in the task domains and leverages a cross-domain consistency [50]. Similarly, Hang *et al*. propose several domain-invariant regularizers to improve the consistency of predictions [364].

Self-training for adaptation. State-of-the-art self-training relies on pseudo labels for the target domain, which are generated by a teacher network [271]. To balance the selection of easy and harder cases in pseudo labels, adaptive thresholds are commonly used for different classes [4, 265] or instances [189]. Consistent predictions across scales are also enforced to generate pseudo labels [265].

Furthermore, IAST smooths pseudo labels in high-confidence regions and sharpens them for low-confidence regions [189]. Zheng and Yang exploit auxiliary classifiers to obtain a confidence measure for the pseudo labels [373]. ProDA weights pseudo labels by their distance to prototypical features to reduce the influence of outliers [359]. Ma *et al*. propose a triplet loss to align average feature representations for different pseudo categories [181].

Contrastive learning is also introduced for domain adaptation. Liu *et al*. [173] identify patches to pair via simplified spatial pyramid matching. ASSUDA uses a contrastive loss to minimize the distance of predictions to adversarial samples [336]. Masden *et al*. construct contrastive pairs from mean feature representations in the source and target domain to features averaged over target images [184].

2.3.3 Summary of Semantic Image Segmentation

For semantic segmentation, many studies have been done to improve accuracy while reducing memory and computation costs. However, all these approaches heavily rely on a large amount of labeled real images.

To address the reliance on plenty of manually labeled data in training deep neural networks (*challenge iii*), we design a self-supervised contrastive pre-training framework for semantic segmentation (Chapter 6) to reduce the requirements of the real-data labels. Moreover, Chapter 7 proposes a contrastive domain adaptation method that takes full use of the synthetic data and the unlabeled real images to achieve accurate image segmentation results.

2.4 3D Point-cloud Segmentation

Segmentation of 3D point clouds or RGB-D images that gives each 3D point/location a category label is widely applicable in many scenarios, including remote sensing, AR/VR, robotics, and self-driving cars. Many deep neural network models have been proposed for this important task. This section focuses on the introduction of recent research progress in different types of 3D segmentation methods. These approaches can be roughly categorized into three types: regular voxel-based networks, irregular point-based networks and others.

2.4.1 Voxel-based Networks

Previous approaches convert point clouds to 2D or 3D volumetric grids/voxels (or similar slices/lattices) [314, 185, 220, 354]. For example, 3D point clouds or shapes have been projected into several 2D image-like grids with 2D convolutions for shape classification and retrieval tasks [264, 220]. Other studies [314, 185, 220, 147, 99] voxelize point clouds into 3D volumetric voxels and apply 3D convolution networks for point-cloud processing and understanding.

Among these studies, VV-Net [190] uses a kernel-based interpolated variational autoencoder (VAE) on regular voxel grids. Instance segmentation models have been proposed on dense 3D voxels/2D grids [140, 354]. Panoptic labels can be predicted using a spatially hashed volumetric map [200]. Efficient feature aggregations have been explored in PVCNN [176] through a voxel-based regular CNN in low resolution to avoid random memory accesses.

To extend voxel-based networks to high-resolution scene segmentation tasks [60, 17], a set of unbalanced octrees have been used to improve the resolution [235]. Sparse Submanifold CNN [56, 80] computes the convolutions only at activated points to design high-resolution volumetric inputs.

2.4.2 Point-based Networks

Recent work takes raw points as input for feature learning and label predictions. PointNet aggregate features from different points by shared multi-layer perceptrons (MLP) [219]. PointNet++ improves its network by adding a hierarchical structure [221]. Wang *et al*. associate instances and semantics segmentation [297] with feature encoder of stacked PointNet layers. He *et al*. learn deep geodesic-aware representations for PointNet/PointNet++ [90]. The issue with the MLP module is that they keep only the most significant activation on features, leading to the loss of some useful detailed information for segmentation.

Other recent work explores the extension of convolutions on irregular and unordered point-cloud inputs [175, 326, 155, 276, 81, 290].

For unordered points, PointCNN performs convolutions on transformed points [155]. Zhang *et al*. use statistics from concentric spherical shells to resolve point-order ambiguities [365]. FeaStNet generalizes convolution layers by adding a soft-assignment matrix [290]. KPConv learns flexible and deformable point convolutions [276]. Point Conv [104, 312], GeoCNN [143] and annular convolution [135] query the nearest neighbors for convolutions with different kernels.

Some work explores contiguous convolutions [205, 312, 299] for point clouds. For instance, PointConv [312] treats convolution kernels as nonlinear functions of the coordinates and uses (MLP) modules to learn continuous weight functions for point-wise convolutions. Hermosilla *et al*. develop Monte Carlo convolutions for learning non-uniformly sampled point clouds [92]. Similarly, Mao *et al*. propose to interpolate discrete convolutional weights for convolutions on points [183].

Other work explores the ideas of convolutions on unique surfaces. Rao *et al*. map 3D points onto a discretized sphere for convolution definition [225]. Huang *et al*. utilize a 4-rotational symmetric field to define a domain for convolution on a surface [106]. Tangent convolution projects local surface geometry on a tangent

plane around every point, producing planar convolution-able tangent images [272].

There are also graph or tree-based convolutions for point clouds, including local spectral graph convolutions [292], graph attention convolutions [297], regularized graph CNN (RGCNN) [273], and octree guided CNN with spherical kernels [149].

2.4.3 Other Point Networks

There are also special networks developed for various applications to directly take raw points as input for sampling [338, 348], semantic [303, 366, 123, 346, 368, 231], and instance segmentation [213, 331]. Some employ new and special data structures for point cloud processing, including 3D point capsule encoder and decoder networks [371], over segmentation on graph structure [144], basis point sets [217], multi-resolution tree-structured networks [72], bilateral convolutions on a sparse lattice of the point cloud [263], recurrent neural networks (RNN) on slices of a point cloud [107], and the superpoint graph-based semantic segmentation [145].

2.4.4 Datasets and Domain Generalization

There are several popular point cloud datasets for semantic segmentation. Most of them provide RGB-D data. For example, Armeni *et al*. [7, 6] provides an indoor dataset with 3D meshes and semantic annotations for 265 rooms and ScanNet [60], a RGB-D video dataset, contains 2.5M views in 1513 scenes with 3D semantic and instance labels. Furthermore, Semantic Kitti [17] is a large-scale LiDAR point cloud dataset in driving scenes. It has 22 sequences with 23K frames, and each scan contains 10–13k points in a large space of $160m \times 160m \times 20m$. These 3D segmentation datasets are usually collected by laser or depth scans and then manually labeled.

To address *challenges iii* and *iv*, boost the study of cross-domain generalization in 3D segmentation and reduce the dependency on manual labels in training 3D segmentation models, we create a large-scale synthetic point-cloud dataset that

consists of 100k LiDAR frames for the driving scenes. We also show that it's possible to train 3D segmentation method (*e.g.* our LiDARSeg in Chapter 9) on the synthetic data to achieve good generalization results on real data.

2.4.5 Summary of 3D Segmentation

For 3D point cloud segmentation, difficulties mainly come from *challenges i∼iii*. It is hard to balance the accuracy and the costs of memory and computations. Many existing methods try to reduce the memory and computational costs in the 3D segmentation networks. However, they neglect the thin structures and small objects and easily produce ambiguous predictions around object edges. Moreover, training existing 3D segmentation networks heavily relies on expensive 3D point labels.

In this thesis, Chapter 8 contributes a new efficient data structure design and neural network modules to store and process the 3D point clouds and significantly reduce the memory and computational costs of the 3D segmentation methods. More importantly, it resolves the ambiguous prediction problem around the object edges and thin structures by assigning each point a distinct label. In Chapter 9, we contribute a large-scale synthetic point-cloud dataset to boost the study of cross-domain generalization in 3D segmentation and reduce the dependency on manual labels in training 3D segmentation models.

2.5 Summary

Different ideas have been explored in solving the four major challenges for dense prediction tasks. While existing approaches have many limitations in solving these challenges. In the following chapters, we propose new approaches to address the four challenges in stereo matching, optical flow and 2D/3D segmentation.

To improve the performance in the challenging thin structures, small objects,

occlusions and reflective regions (*challenge i*), we propose novel neural network layers and design new effective architectures to achieve state-of-the-art accuracy for stereo matching (in Chapter 3), optical flow (Chapter 5) and 3D point cloud segmentation (Chapter 8 and 9).

We also develop more effective matching cost aggregation layers for stereo matching (Chapter 3), a simplified matching mechanism (Chapter 5) for optical flow and the novel data structure for efficient 3D point cloud storage and processing (in Chapter 8). All of these significantly reduce the memory and computational costs in dense prediction networks (*challenge ii*).

Moreover, to address the reliance on manual labels in training deep neural networks (*challenge iii*), we design a self-supervised contrastive pre-training framework for semantic segmentation (Chapter 6 and Chapter 7) and take full use of the synthetic data in training optical flow (Chapter 5) and stereo matching (Chapter 4).

Finally, we develop domain-invariant stereo matching (Chapter 4) and optical flow (Chapter 5) networks that can be trained on synthetic data and directly applied to other unseen real datasets without re-training or fine-tuning (*challenge iv*).

Chapter 3

GA-Net: Guided Aggregation Net for End-to-end Stereo Matching

Feihu Zhang

University of Oxford

Victor Prisacariu

University of Oxford

Ruigang Yang

Baidu Research, Baidu Inc.

Philip H. S. Torr

University of Oxford

Abstract

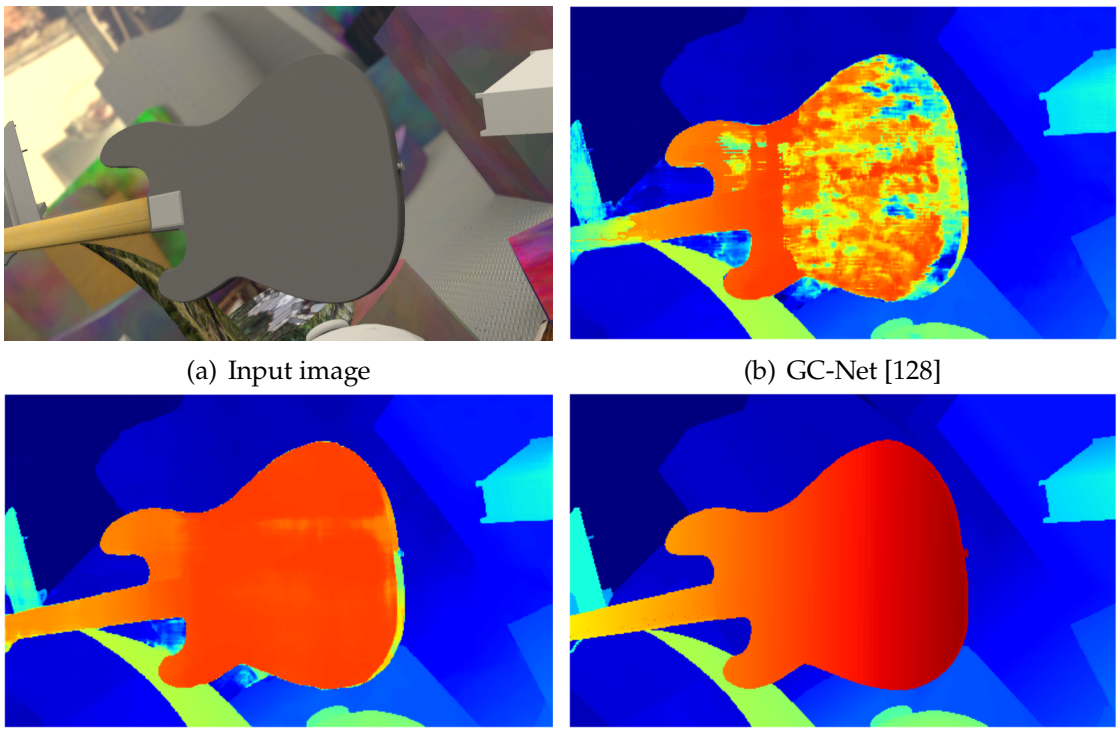
In the stereo matching task, matching cost aggregation is crucial in both traditional methods and deep neural network models in order to accurately estimate disparities. We propose two novel neural net layers, aimed at capturing local and the whole-image cost dependencies respectively. The first is a semi-global aggregation layer which is a differentiable approximation of the semi-global matching, the second is the local guided aggregation layer which follows a traditional cost filtering strategy to refine thin structures.

These two layers can be used to replace the widely used 3D convolutional layer which is computationally costly and memory-consuming as it has cubic computational/memory complexity. In the experiments, we show that nets with a two-layer guided aggregation block easily outperform the state-of-the-art GC-Net which has nineteen 3D convolutional layers. We also train a deep guided aggregation network (GA-Net) which gets better accuracies than state-of-the-art methods on both Scene Flow dataset and KITTI benchmarks.

3.1 Introduction

Stereo reconstruction is a major research topic in computer vision, robotics and autonomous driving. It aims to estimate 3D geometry by computing disparities between matching pixels in a stereo image pair. It is challenging due to a variety of real-world problems, such as occlusions, large textureless areas (*e.g.* sky, walls *etc.*), reflective surfaces (*e.g.* windows), thin structures and repetitive textures.

Traditionally, stereo reconstruction is decomposed into three important steps: feature extraction (for matching cost computation), matching cost aggregation



(a) Input image (b) GC-Net [128]

(c) Our GA-Net-2 (d) Ground truth

Figure 3.1: Performance illustrations. (a) a challenging input image. (b) Result of the state-of-the-art method GC-Net [128] which has nineteen 3D convolutional layers for matching cost aggregation. (c) Result of our GA-Net-2, which only uses two proposed GA layers and two 3D convolutional layers. It aggregates the matching information into the large textureless region and is an order of magnitude faster than GC-Net. (d) Ground truth.

and disparity prediction [243, 93]. Feature-based matching is often ambiguous, with wrong matches having a lower cost than the correct ones, due to occlusions, smoothness, reflections, noise etc. Therefore, cost aggregation is a key step needed to obtain accurate disparity estimations in challenging regions.

Deep neural networks have been used for matching cost computation in, e.g, [352, 358], with (i) cost aggregation based on traditional approaches, such as cost filtering [98] and semi-global matching (SGM) [93] and (ii) disparity computation with a separate step. Such methods considerably improve over traditional pixel matching, but still struggle to produce accurate disparity results in textureless, reflective and occluded regions. End-to-end approaches that link matching with

disparity estimation were developed in e.g. DispNet [187], but it was not until GC-Net [128] that cost aggregation, through the use of 3D convolutions, was incorporated in the training pipeline. The more recent work of [32], PSMNet, further improves accuracy by implementing the stacked hourglass backbone [202] and considerably increasing the number of 3D convolutional layers for cost aggregation. The large memory and computation cost incurred by using 3D convolutions is reduced by down-sampling and up-sampling frequently, but this leads to a loss of precision in the disparity map.

Among these approaches, traditional semi-global matching (SGM) [93] and cost filtering [98] are all robust and efficient cost aggregation methods which have been widely used in many industrial products. But, they are not differentiable and cannot be easily trained in an end-to-end manner.

In this work, we propose two novel cost aggregation layers for end-to-end stereo reconstruction to replace the use of 3D convolutions. Our solution considerably increases accuracy, while decreasing both memory and computation costs.

First, we introduce a semi-global guided aggregation layer (SGA) which implements a differentiable approximation of semi-global matching (SGM) [93] and aggregates the matching cost in different directions over the whole image. This enables accurate estimations in occluded regions or large textureless/reflective regions.

Second, we introduce a local guided aggregation layer (LGA) to cope with thin structures and object edges in order to recover the loss of details caused by down-sampling and up-sampling layers.

As illustrated in Fig. 3.1, a cost aggregation block with only two GA layers and two 3D convolutional layers easily outperforms the state-of-the-art GC-Net [128], which has nineteen 3D convolutional layers. More importantly, one GA layer has only 1/100 computational complexity in terms of FLOPs (floating-point operations)

as that of a 3D convolution. This allows us to build a real-time GA-Net model, which achieves better accuracy compared with other existing real-time algorithms and runs at a speed of 15∼20 fps.

We further increase the accuracy by improving the network architectures used for feature extraction and matching cost aggregation. The full model, which we call "GA-Net", achieves the state-of-the-art accuracy on both the Scene Flow dataset [187] and the KITTI benchmarks [75, 191].

3.2 Related Work

Feature based matching cost is often ambiguous, as wrong matches can easily have a lower cost than correct ones, due to occlusions, smoothness, reflections, noise etc. To deal with this, many cost aggregation approaches have been developed to refine the cost volume and achieve better estimations. This section briefly introduces related work in the application of deep neural networks in stereo reconstruction with a focus on the existing matching cost aggregation strategies, and briefly reviews approaches for traditional local and semi-global cost aggregations.

3.2.1 Deep Neural Networks for Stereo Matching

Deep neural networks were used to compute patch-wise similarity scores in [351, 358, 51, 71], with traditional cost aggregation and disparity computation/refinement methods [93, 98] used to get the final disparity maps. These approaches achieved state-of-the-art accuracy, but, limited by the traditional matching cost aggregation step, often produced wrong predictions in occluded regions, large textureless/reflective regions and around object edges. Some other methods looked to improve the performance of traditional cost aggregation, with, e.g. SGM-Nets [248] predicting the penalty-parameters for SGM [93] using a neural net, whereas

Schönberger *et al*. [244] learned to fuse proposals by optimization in stereo matching and Yang *et al*. proposed to aggregate costs using a minimum spanning tree [339].

Recently, end-to-end deep neural network models have become popular. Mayer *et al*. created a large synthetic dataset to train end-to-end deep neural network for disparity estimation (*e.g*. DispNet) [187]. Pang *et al*. [207] built a two-stage convolutional neural network to first estimate and then refine the disparity maps. Tulyakov *et al*. proposed end-to-end deep stereo models for practical applications [287]. GC-Net [128] incorporated the feature extraction, matching cost aggregation and disparity estimation into a single end-to-end deep neural model to get state-of-the-art accuracy on several benchmarks. PSMNet [32] used pyramid feature extraction and a stacked hourglass block [202] with twenty-five 3D convolutional layers to further improve the accuracy.

3.2.2 Cost Aggregation

Traditional stereo matching algorithms [93, 325, 19] added an additional constraint to enforce smoothness by penalizing changes of neighboring disparities. This can be both local and (semi-)global, as described below.

3.2.2.1 Local Cost Aggregation

The cost volume C is formed of matching costs at each pixel's location for each candidate disparity value d. It has a size of $H \times W \times D_{max}$ (with H: image height, W: image width, D_{max}: maximum of the disparities) and can be sliced into D_{max} slices for each candidate disparity d. An efficient cost aggregation method is the local cost filter framework [98, 353], where each slice of the cost volume $C(d)$ is filtered independently by a guided image filter [88, 279, 353]. The filtering for pixel's location $\mathbf{p} = (x, y)$ at disparity d is a weighted average of all neighborhoods

$\mathbf{q} \in N_{\mathbf{p}}$ in the same slice $C(d)$:

$$C^A(\mathbf{p}, d) = \sum_{\mathbf{q} \in N_{\mathbf{p}}} \omega(\mathbf{p}, \mathbf{q}) \cdot C(\mathbf{q}, d) \tag{3.1}$$

Where $C(\mathbf{q}, d)$ means the matching cost at location $\mathbf{p}$ for candidate disparity d. $C^A(\mathbf{p}, d)$ represents the aggregated matching cost. Different image filters [88, 279, 353] can be used to produce the guided filter weights ω. Since these methods only aggregate the cost in a local region $N_{\mathbf{p}}$, they can run at fast speeds and reach real-time performance.

3.2.2.2 Semi-Global Matching

When enforcing (semi-)global aggregation, the matching cost and the smoothness constraints are formulated into one energy function $E(D)$ [93] with the disparity map of the input image as D. The problem of stereo matching can now be formulated as finding the best disparity map D^* that minimizes the energy $E(D)$:

$$\begin{aligned} E(D) = \textstyle\sum_{\mathbf{p}} \{ C_{\mathbf{p}}(D_{\mathbf{p}}) + \sum_{\mathbf{q} \in N_{\mathbf{p}}} P_1 \cdot \delta(|D_{\mathbf{p}} - D_{\mathbf{q}}| = 1) \\ + \textstyle\sum_{\mathbf{q} \in N_{\mathbf{p}}} P_2 \cdot \delta(|D_{\mathbf{p}} - D_{\mathbf{q}}| > 1) \}. \end{aligned} \tag{3.2}$$

The first term $\sum_{\mathbf{p}} C_{\mathbf{p}}(D_{\mathbf{p}})$ is the sum of matching costs at all pixel locations $\mathbf{p}$ for disparity map D. The second term is a constant penalty P_1 for locations $\mathbf{q}$ in the neighborhood of $\mathbf{p}$ if they have small disparity discontinuities in disparity map D ($|D_{\mathbf{p}} - D_{\mathbf{q}}| = 1$). The last term adds a larger constant penalty P_2, for all larger disparity changes ($|D_{\mathbf{p}} - D_{\mathbf{q}}| > 1$).

Hirschmuller proposed to aggregate matching costs in 1D from sixteen directions to get a approximate solution with $O(KN)$ time complexity, which is well known as semi-global matching (SGM) [93]. The cost $C^A_{\mathbf{r}}(\mathbf{p}, d)$ of a location $\mathbf{p}$ at disparity d aggregates along a path over the whole image in the direction $\mathbf{r}$, and is defined

recursively as:

$$C_{\mathbf{r}}^{A}(\mathbf{p},d)=C(\mathbf{p},d)+\min\begin{cases}C_{\mathbf{r}}^{A}(\mathbf{p}-\mathbf{r},d),\\ C_{\mathbf{r}}^{A}(\mathbf{p}-\mathbf{r},d-1)+P_1,\\ C_{\mathbf{r}}^{A}(\mathbf{p}-\mathbf{r},d+1)+P_1,\\ \min\limits_{i} C_{\mathbf{r}}^{A}(\mathbf{p}-\mathbf{r},i)+P_2.\end{cases} \tag{3.3}$$

Where $\mathbf{r}$ is a unit direction vector. The same aggregation steps were used in MC-CNN [352, 248], and similar iterative steps were employed in [19, 21, 171].

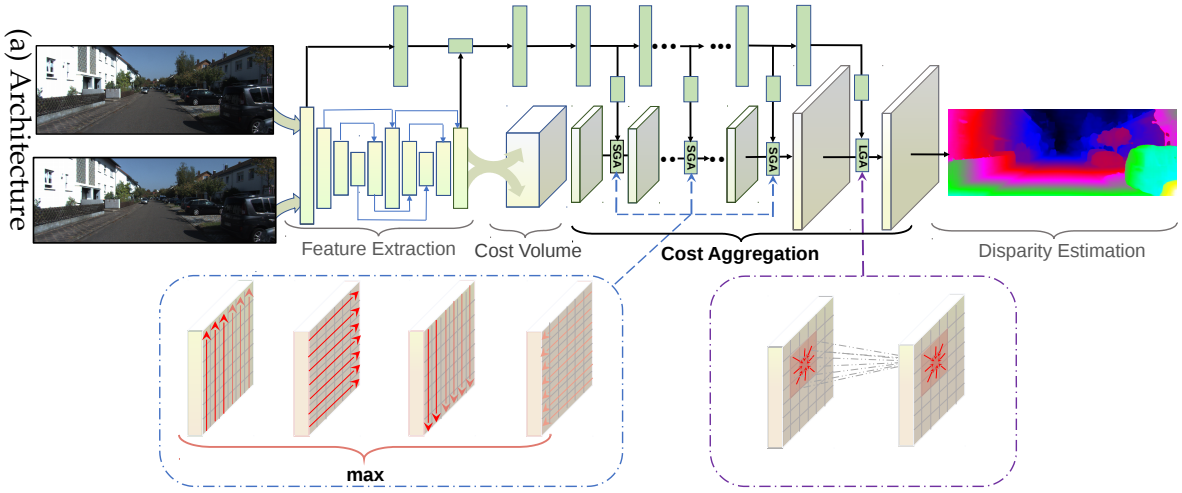
In the following section, we detail our much more efficient guided aggregation (GA) strategies, which include a semi-global aggregation (SGA) layer and a local guided aggregation (LGA) layer. Both GA layers can be implemented with back propagation in end-to-end models to replace the low-efficient 3D convolutions and obtain higher accuracy.

3.3 Guided Aggregation Net

In this section, we describe our proposed guided aggregation network (GA-Net), including the guided aggregation (GA) layers and the improved network architecture.

3.3.1 Guided Aggregation Layers

State-of-the-art end-to-end stereo matching neural nets such as [128, 32] build a 4D matching cost volume (with size of $H \times W \times D_{max} \times F$, H: height, W: width, D_{max}: max disparity, F: feature size) by concatenating features between the stereo views, computed at different disparity values. This is next refined by a cost aggregation stage, and finally used for disparity estimation. Different from these approaches, and inspired by semi-global and local matching cost aggregation methods [93, 98], we propose our semi-global guided aggregation (SGA) and local guided aggregation



(b) Semi-Global Guided Aggregation (SGA) (c) Local Guided Aggregation (LGA)

Figure 3.2: (a) Architecture overview. The left and right images are fed to a weight-sharing feature extraction pipeline. It consists of a stacked hourglass CNN and is connected by concatenations. The extracted left and right image features are then used to form a 4D cost volume, which is fed into a cost aggregation block for regularization, refinement and disparity regression. The guidance subnet (green) generates the weight matrices for the guided cost aggregations (SGA and LGA). (b) SGA layers semi-globally aggregate the cost volume in four directions. (c) The LGA layer is used before the disparity regression and locally refines the 4D cost volume for several times.

(LGA) layers, as outlined below.

3.3.1.1 Semi-Global Aggregation

Traditional SGM [93] aggregates the matching cost iteratively in different directions (Eq. (3.3)). There are several difficulties in using such a method in end-to-end trainable deep neural network models.

First, SGM has many user-defined parameters (P_1, P_2), which are not straightforward to tune. All of these parameters become unstable factors during neural network training. Second, the cost aggregations and penalties in SGM are fixed for all pixels, regions and images without adaptation to different conditions. Third, the hard-minimum selection leads to a lot of fronto parallel surfaces in depth estimations.

We design a new semi-global cost aggregation step which supports backpropagation. This is more effective than the traditional SGM and can be used repetitively in a deep neural network model to boost the cost aggregation effects. The proposed aggregation step is:

$$
\begin{aligned}
C_{\mathbf{r}}^{A}(\mathbf{p}, d) = C(\mathbf{p}, d) \\
+ \text{ sum} \begin{cases}
\mathbf{w}_1(\mathbf{p}, \mathbf{r}) \cdot C_{\mathbf{r}}^{A}(\mathbf{p} - \mathbf{r}, d), \\
\mathbf{w}_2(\mathbf{p}, \mathbf{r}) \cdot C_{\mathbf{r}}^{A}(\mathbf{p} - \mathbf{r}, d - 1), \\
\mathbf{w}_3(\mathbf{p}, \mathbf{r}) \cdot C_{\mathbf{r}}^{A}(\mathbf{p} - \mathbf{r}, d + 1), \\
\mathbf{w}_4(\mathbf{p}, \mathbf{r}) \cdot \max\limits_{i} C_{\mathbf{r}}^{A}(\mathbf{p} - \mathbf{r}, i).
\end{cases}
\end{aligned}
\tag{3.4}
$$

This is different from the SGM in three ways. First, we make the user-defined parameters learnable and add them as penalty coefficients/weights of the matching cost terms. These weights would therefore be adaptive and more flexible at different locations for different situations. Second, we replace the first/external minimum selection in Eq. (3.3) with a weighted sum, without any loss in accuracy. This change was proven effective in [262], where convolutions with strides were used to replace the max-pooling layers to get an all convolutional network without loss of accuracy. Third, the internal/second minimum selection is changed to a maximum. This is because the learning target in our models is to maximize the probabilities at the ground truth depths instead of minimizing the matching costs. Since $\max\limits_{i} C_{\mathbf{r}}^{A}(\mathbf{p} - \mathbf{r}, i)$ in Eq. (3.4) can be shared by $C_{\mathbf{r}}^{A}(\mathbf{p}, d)$ for d different locations, here, we do not use another weighted summation to replace it in order to reduce the computational complexity.

For both Eq. (3.3) and Eq. (3.4), the values of $C_{\mathbf{r}}^{A}(\mathbf{p}, d)$ increase along the path, which may lead to very large values. We normalize the weights of the terms to

avoid such a problem. This leads to our new semi-global aggregation:

$$
\begin{aligned}
C_{\mathbf{r}}^{A}(\mathbf{p}, d) \;=\; \text{sum} & \begin{cases} \mathbf{w}_0(\mathbf{p}, \mathbf{r}) \cdot C(\mathbf{p}, d) \\ \mathbf{w}_1(\mathbf{p}, \mathbf{r}) \cdot C_{\mathbf{r}}^{A}(\mathbf{p} - \mathbf{r}, d), \\ \mathbf{w}_2(\mathbf{p}, \mathbf{r}) \cdot C_{\mathbf{r}}^{A}(\mathbf{p} - \mathbf{r}, d-1), \\ \mathbf{w}_3(\mathbf{p}, \mathbf{r}) \cdot C_{\mathbf{r}}^{A}(\mathbf{p} - \mathbf{r}, d+1). \\ \mathbf{w}_4(\mathbf{p}, \mathbf{r}) \cdot \max\limits_{i} C_{\mathbf{r}}^{A}(\mathbf{p} - \mathbf{r}, i). \end{cases} \\
s.t. \quad & \sum_{i=0,1,2,3,4} \mathbf{w}_i(\mathbf{p}, \mathbf{r}) = 1
\end{aligned}
\tag{3.5}
$$

$C(\mathbf{p}, d)$ is known as the cost volume (with a size of $H \times W \times D_{max} \times F$). Same as the traditional SGM [93], the cost volume can be sliced into D_{max} slices at the third dimension for each candidate disparity d and each of these slices repeats the aggregation operation of Eq. (3.5) with the shared weight matrices ($\mathbf{w}_{0...4}$). All the weights $\mathbf{w}_{0...4}$ can be achieved by a guidance subnet (as shown in Fig. 3.2). Different to the original SGM which aggregates in sixteen directions, in order to improve the efficiency, the proposed aggregations are done in totally four directions (left, right, up and down) along each row or column over the whole image, namely $\mathbf{r} \in \{(0,1), (0,-1), (1,0), (-1,0)\}$.

The final aggregated output $C^{A}(\mathbf{p})$ is obtained by selecting the maximum between the four directions:

$$
C^{A}(\mathbf{p}, d) = \max_{\mathbf{r}} \; C_{\mathbf{r}}^{A}(\mathbf{p}, d) \tag{3.6}
$$

The last maximum selection keeps the best message from only one direction. This guarantees that the aggregation effects are not blurred by the other directions. The backpropagation for $\mathbf{w}$ and $C(\mathbf{p}, d)$ in the SGA layer can be done inversely as Eq. (3.5) (details are available in the supplementary materials). Our SGA layer can be repeated several times in the neural network model to obtain better cost aggregation effects (as illustrated in Fig. 3.2).

3.3.1.2 Local Aggregation

We now introduce the local guided aggregation (LGA) layer which aims to refine the thin structures and object edges. Down-sampling and up-sampling are widely used in stereo matching models which blurs thin structures and object edges. The LGA layer learns several guided filters to refine the matching cost and aid in the recovery of thin structure information. The local aggregation follows the cost filter definition [98] (Eq. (3.1)) and can be written as:

$$\begin{aligned} C^A(\mathbf{p},d) &= \text{sum}\begin{cases} \sum_{\mathbf{q}\in N_\mathbf{p}} \omega_0(\mathbf{p},\mathbf{q})\cdot C(\mathbf{q},d), \\ \sum_{\mathbf{q}\in N_\mathbf{p}} \omega_1(\mathbf{p},\mathbf{q})\cdot C(\mathbf{q},d-1), \\ \sum_{\mathbf{q}\in N_\mathbf{p}} \omega_2(\mathbf{p},\mathbf{q})\cdot C(\mathbf{q},d+1). \end{cases} \\ s.t. &\sum_{\mathbf{q}\in N_\mathbf{p}} \omega_0(\mathbf{p},\mathbf{q}) + \omega_1(\mathbf{p},\mathbf{q}) + \omega_2(\mathbf{p},\mathbf{q}) = 1 \end{aligned} \tag{3.7}$$

Different slices (totally D_{max} slices) of cost volume share the same filtering/aggregation weights in LGA. This is the same as the original cost filter framework [98] and the SGA (Eq.(3.5)) in this paper. While, different with the traditional cost filter [98] which uses a $K \times K$ filter kernel to filter the cost volume in a $K \times K$ local/neighboor region $N_\mathbf{p}$, the proposed LGA layer has three $K \times K$ filters (ω_0, ω_1 and ω_2) at each pixel location $\mathbf{p}$ for disparities d, $d-1$ and $d+1$ respectively. Namely, it aggregates with a $K \times K \times 3$ weight matrix in a $K \times K$ local region for each pixel location $\mathbf{p}$. The setting of the weight matrix is also similar to [121], but, weights and filters are shared during the aggregation as designed in [98].

3.3.1.3 Efficient Implementation

We use several 2D convolutional layers to build a fast guidance subnet (as illustrated in Fig. 3.2). The implementation is similar to [357]. It uses the reference image as input and outputs the aggregation weights $\mathbf{w}$ (Eq. (3.5)). For a 4D cost volume C

with size of $H \times W \times D \times F$ (H: height, W: width, D: max disparity, F: feature size), the output of the guidance subnet is split, reshaped and normalized as four $H \times W \times K \times F$ ($K = 5$) weight matrices for four directions' aggregation using Eq. (3.5). Note that aggregations for different disparities corresponding to a slice d share the same aggregation weights. Similarly, the LGA layer need to learn a $H \times W \times 3K^2 \times F$ ($K = 5$) weight matrix and aggregates using Eq. (3.7).

Even though the SGA layer involves an iterative aggregation across the width or the height, the forward and backward can be computed in parallel due to the independence between elements in different feature channels or rows/columns. For example, when aggregating in the left direction, the elements in different channels or rows are independent and can be computed simultaneously. The elements of the LGA layer can also be computed in parallel by simply decomposing it into element-wise matrix multiplications and summations. In order to increase the receptive field of the LGA layer, we repeat the computation of Eq. (3.7) twice with the same weight matrix, which is similar to [53].

3.3.2 Network Architecture

As illustrated in Fig. 3.2, the GA-Net consists of four parts: the feature extraction block, the cost aggregation for the 4D cost volume, the guidance subnet to produce the cost aggregation weights and the disparity regression. For the feature extraction, we use a stacked hourglass network which is densely connected by concatenations between different layers. The feature extraction block is shared by both left and right views. The extracted features for left and right images are then used to form a 4D cost volume. Several SGA layers are used for the cost aggregation and LGA layers can be implemented before and after the softmax layer of the disparity regression. It refines the thin-structures and compensate for the accuracy loss caused by the down-sampling done for the cost volume. The weight matrices (in Eq. (3.5) and

Eq. (3.7)) are generated by an extra guidance subnet which uses the reference view (*e.g.* the left image) as input. The guidance subnet consists of several fast 2D convolutional layers and the outputs are reshaped and normalized into required weight matrices for these GA layers.*

3.3.3 Loss Function

We adopt the smooth L_1 loss function to train our models. Smooth L_1 is robust at disparity discontinuities and has low sensitivity to outliers or noises, as compared to L_2 loss. The loss function for training our models is defined as:

$$\begin{aligned} L(\hat{d}, d) &= \tfrac{1}{N} \sum_{n=1}^{N} l(|\hat{d} - d|) \\ l(x) &= \begin{cases} x - 0.5, & x \geq 1 \\ x^2/2, & x < 1 \end{cases} \end{aligned} \tag{3.8}$$

where, $|\hat{d} - d|$ measures the absolute error of the disparity predictions, N is the number of valid pixels with ground truths for training.

For the disparity estimation, we employ the disparity regression proposed in [128]:

$$\hat{d} = \sum_{d=0}^{D_{max}} d \times \sigma(-C^A(d)) \tag{3.9}$$

The disparity prediction $\hat{d}$ is the sum of each disparity candidate weighted by its probability. The probability of each disparity d is calculated after cost aggregation via the softmax operation $\sigma(\cdot)$. The disparity regression is shown more robust than classification based methods and can generate sub-pixel accuracy.

*The parameter settings of "GA-Net-15" used in our experiments are detailed in the supplementary material (available at www.feihuzhang.com).

Table 3.1: Evaluations of GA-Nets with different settings. Average end point error (EPE) and threshold error rate are used for evaluations.

Feature	Cost Aggregation			Scene Flow		KITTI 2015
Stacked Block	Concate	SGA	LGA	EPE Error	Error Rates (%)	Error Rates (%)
				1.26	13.4	3.39
√				1.19	13.0	3.31
√	√			1.14	12.5	3.25
√	√	+1		1.05	11.7	3.09
√	√	+2		0.97	11.0	2.96
√	√	+3		0.90	10.5	2.85
√	√	+4		0.89	10.4	2.83
√	√	+3	√	**0.84**	**9.9**	**2.71**

3.4 Experiments

In this section, we evaluate our GA-Nets with different settings using Scene Flow [187] and KITTI [75, 191] datasets. We implement our architectures using pytorch or caffe [122] (only for real-time models' implementation). All models are optimized with Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$). We train with a batch size of 16 on eight GPUs using 240×576 random crops from the input images. The maximum of the disparity is set as 192. Before training, we normalize each channel of the image by subtracting their means and dividing their standard deviations. We train the model on Scene Flow dataset for 10 epochs with a constant learning rate of 0.001. For the KITTI datasets, we fine-tune the models pre-trained on Scene Flow dataset for a further 640 epochs. The learning rate for fine-tuning begins at 0.001 for the first 300 epochs and decreases to 0.0001 for the remaining epochs.

3.4.1 Ablation Study

We evaluate the performance of GA-Nets with different settings, including different architectures and different number (0-4) of GA layers. As listed in Table 3.1, The guided aggregation models significantly outperform the baseline setting which only has 3D convolutional layers for cost aggregation. The new architectures for feature extraction and cost aggregation improve the accuracy by 0.14% on KITTI dataset

and 0.9% on Scene Flow dataset. Finally, the best setting of GA-Net with three SGA layers and one LGA layer gets the best 3-pixel threshold error rate of 2.71% on KITTI 2015 validation set. It also achieves the best average EPE of 0.84 pixel and the best 1-pixel threshold error rate of 9.9% on the Scene Flow test set.

3.4.2 Effects of Guided Aggregations

In this section, we compare the guided aggregation strategies with other matching cost aggregation methods. We also analyze the effects of the GA layers by observing the post-softmax probabilities output by different models.

Firstly, our proposed GA-Nets are compared with the cost aggregation architectures in GC-Net (with nineteen 3D convolutions) and PSMNet (with twenty-five 3D convolutions). We fixed the feature extraction architecture as proposed above. As shown in Table 3.2, GA-Nets have fewer parameters, run at a faster speed and achieve better accuracy. *E.g.*, with only two GA layers and two 3D convolutions, our GA-Net-2 outperforms the GC-Net by 0.29 pixel in average EPE. Also, the GA-Net-7 with three GA layers and seven 3D convolutions outperforms the current best PSMNet [32] which has twenty-five 3D convolutional layers.

We also study the effects of the GA layers by comparing with the same architectures without GA steps. These baseline models "GA-Nets*" have the same network architectures and all other settings except that there is no GA layer implemented. As shown in Fig. 3.3, for all these models, GA layers have significantly improved the models' accuracy (by 0.5-1.0 pixels in average EPE). For example, the GA-Net-2 with two 3D convolutions and two GA layers produces lower EPE (1.51) compared with GA-Net*-11 (1.54) which utilizes eleven 3D convolutions. This implies that two GA layers are more effective than nine 3D convolutional layers.

Finally, in order to observe and analyze the effects of GA layers, in Fig. 3.4, we plot the post-softmax probabilities with respect to a range of candidate disparities.

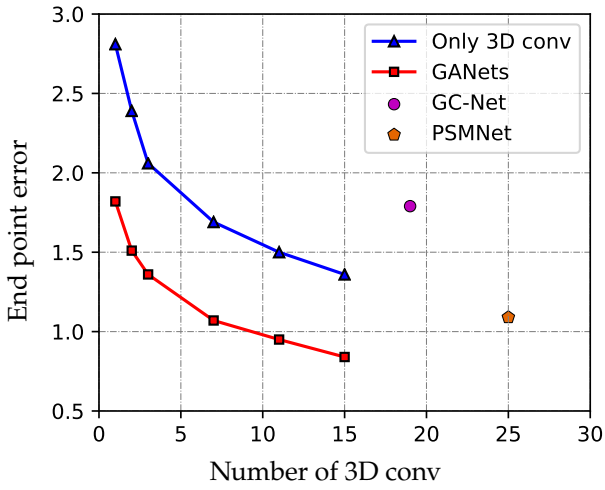


Figure 3.3: Illustration of the effects of guided aggregations. GA-Nets are compared with the same architectures without GA Layers. Evaluations are on Scene Flow dataset using average EPE.

These probabilities are directly used for disparity estimation using Eq. (3.9) and can reflect the effectiveness of the cost aggregation strategies. The data samples are all selected from some challenging regions, such as a large textureless region (sky), the reflective region (window of a car) and pixels around the object edges. Three different models are compared. The first model (first row of Fig. 3.4) only has 3D convolutions (without any GA layers), the second model (second row of Fig. 3.4) has SGA layers and the last model (last row of Fig. 3.4) has both SGA layers and LGA layer.

As illustrated in Fig. 3.4(a), for large textureless regions, there would be a lot of noise since there is no any distinctive features in these regions for correct matching. The SGA layers successfully suppress these noise in the probabilities by aggregating surrounding matching information. The LGA layer further concentrates the probability peak on the ground truth value. It could refine the matching results. Similarly, in the sample of reflective region (Fig. 3.4(b)), the SGA and LGA layers correct the wrong matches and concentrate the peak on the correct disparity value. For the samples around the objects edges (Fig. 3.4(c)), there are usually two peaks

Table 3.2: Comparisons of different cost aggregation methods. Average end point error (EPE) and 1-pixel threshold error rate are used for evaluations on Scene Flow dataset.

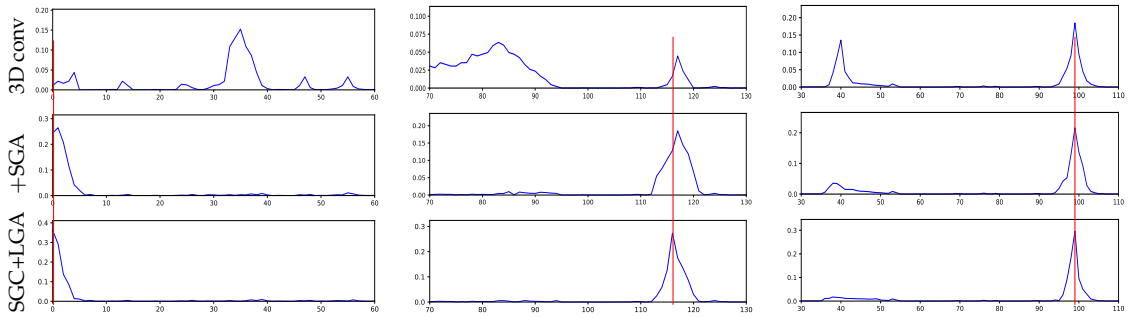
Models	3D Conv Number	Param	Time(s)	EPE Error	Error Rates
GC-Net	19	2.9M	4.4	1.80	15.6
PSMNet	25	3.5M	2.1	1.09	12.1
GA-Net-1	1	0.5M	0.17	1.82	16.5
GA-Net-2	2	0.7M	0.35	1.51	15.0
GA-Net-3	3	0.8M	0.42	1.36	13.9
GA-Net-7	7	1.3M	0.62	1.07	11.9
GA-Net-11	11	1.8M	0.95	0.95	10.8
GA-Net-15	15	2.3M	1.5	**0.84**	**9.9**

in the probability distribution which are influenced by the background and the foreground respectively. The SGA and LGA use spatial aggregation along with appropriate maximum selection to cut down the aggregation of the wrong matching information from the background and therefore suppress the false probability peak appeared at the background's disparity value.

3.4.3 Comparisons with SGMs and 3D Convolutions

The SGA layer is a differentiable approximation of the SGM [93]. But, it produces far better results compared with both the original SGM with handcrafted features and the MC-CNN [352] with CNN based features (as shown in Table 3.5). This is because 1) SGA does not have any user-defined parameters that are all learned in an end-to-end fashion. 2) The aggregation of SGA is fully guided and controlled by the weight matrices. The guidance subnet learns effective geometrical and contextual knowledge to control the directions, scopes and strengths of the cost aggregations.

Moreover, compared with original SGM, most of the fronto-parallel approximations in large textureless regions have been avoided. (Example is in Fig. 3.5.) This might be benefited from: 1) the use of the soft weighted sum in Eq. (3.5) (instead of the hard min/max selection in Eq. (3.3)); and 2) the regression loss of Eq. (3.9)

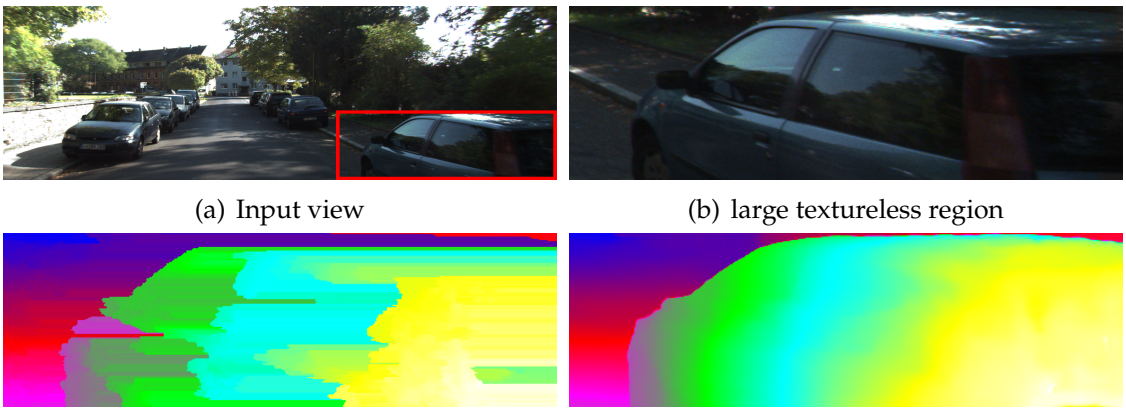


(a) large textureless region (sky) (b) reflective region (window) (c) object edges

Figure 3.4: Post-softmax probability distributions with respect to disparity values. Red lines illustrate the ground truth disparities. Samples are selected from three challenging regions: (a) the large smooth region (sky), (b) the reflective region from one car window and (c) one region around the object edges. The *first row* shows the probability distributions without GA layers. The *second row* shows the effects of semi-global aggregation (SGA) layers and the *last row* is the refined probabilities with one extra local guided aggregation (LGA) layer.

which helps achieve the subpixel accuracy.

Our SGA layer is also more efficient and effective than the 3D convolutional layer. This is because the 3D convolutional layer could only aggregate in a local region restricted by the kernel size. As a result, a series of 3D convolutions along with encoder and decoder architectures are indispensable in order to achieve good results. As a comparison, our SGA layer aggregates semi-globally in a single layer which is more efficient. Another advantage of the SGA is that the aggregation's direction, scope and strength are fully guided by variable weights according to different geometrical and contextual information in different locations. *E.g.*, the SGA behaves totally different in the occlusions and the large smoothness regions. But, the 3D convolutional layer has fixed weights and always perform the same for all locations in the whole image.



(a) Input view (b) large textureless region

(c) result of traditional SGM (d) result of our GA-Net-15

Figure 3.5: Comparisons with traditional SGM. More results and comparisons are avaiable at GA-Net-15 and SGM.

Table 3.3: Comparisons with existing real-time algorithms

Methods	End point error	Error rates	Speed (fps)
Our GA-Net	**0.7 px**	**3.21 %**	15 (GPU)
DispNet [187]	1.0 px	4.65 %	22 (GPU)
Toast [222]	1.4 px	7.42 %	25 (CPU)

3.4.4 Complexity and Real-time Models

The computational complexity of one 3D convolutional layer is $O(K^3CN)$, where N is the elements number of the output blob. K is the size of the convolutional kernel and C is the channel number of the input blob. As a comparison, the complexity of SGA is $O(4KN)$ or $O(8KN)$ for four or eight-direction aggregations. In GC-Net [128] and PSMNet [32], $K = 3$, $C = 32, 64$ or 128 and in our GA-Nets, K is used as 5 (for SGA layer). Therefore, the computational complexity in terms of floating-point operations (FLOPs) of the proposed SGA step is less than $1/100$ of one 3D convolutional layer.

The SGA layer are much faster and more effective than 3D convolutions. This allows us to build an accurate real-time model. We implement one caffe [122] version of the GA-Net-1 (with only one 3D convolutional layer and without LGA layers). The model is further simplified by using $4\times$ down-sampling and up-sampling for

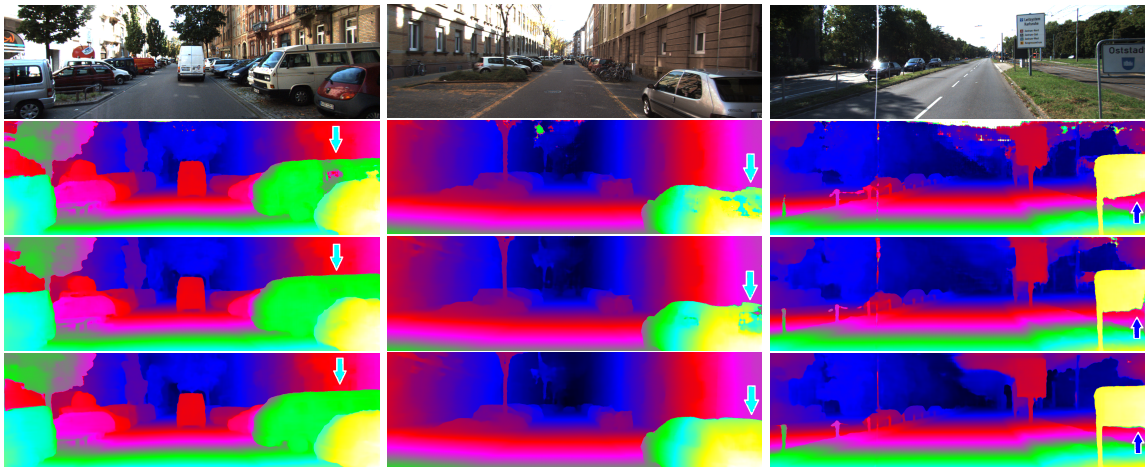


Figure 3.6: Results visualization and comparisons. *First row:* input image. *Second row:* Results of GC-Net [128]. *Third row:* Results of PSMNet [32]. *Last row:* Results of our GA-Net. Significant improvements are pointed out by blue arrows. The guided aggregations can effectively aggregate the disparity information to the large textureless regions (*e.g.* the cars and the windows) and give precise estimations. It can also aggregate the object knowledge and preserve the depth structure very well (last column).

cost volume. The real-time model could run at a speed of 15∼20 fps for 300×1000 images on a TESLA P40 GPU. We also compare the accuracy of the results with the state-of-the-art real-time models. As shown in Table 3.3, the real-time GA-Net far outperforms other existing real-time stereo matching models.

3.4.5 Evaluations on Benchmarks

For the benchmark evaluations, we use the GA-Net-15 with full settings for evaluations. We compare our GA-Net with the state-of-the-art deep neural network models on the Scene Flow dataset and the KITTI benchmarks.

3.4.5.1 Scene Flow Dataset

The Scene Flow synthetic dataset [187] contains 35,454 training and 4,370 testing images. We use the "final" set for training and testing. GA-Nets are compared with

Table 3.4: Evaluation Results on KITTI 2012 Benchmark

Models	error rates (2 pixels)	error rates (3 pixels)	Reflective regions	Avg-All (end point)
Our GA-Net	**2.18** %	**1.36** %	**7.87**%	**0.5 px**
PSMNet [32]	2.44 %	1.49 %	8.36%	0.6 px
GC-Net [128]	2.71 %	1.77 %	10.80%	0.7 px
MC-CNN [352]	3.90 %	2.43 %	17.09%	0.9 px

Table 3.5: Evaluation Results on KITTI 2015 Benchmark

Models	Non Occlusion		All Areas	
	Foreground	Avg All	Foreground	Avg All
Our GA-Net-15	**3.39**%	**1.84%**	**3.91%**	**2.03%**
PSMNet [32]	4.31%	2.14 %	4.62%	2.32%
GC-Net [128]	5.58%	2.61%	6.16%	2.87%
SGM-Nets [248]	7.43%	3.09%	8.64%	3.66%
MC-CNN [352]	7.64%	3.33%	8.88%	3.89%
SGM [93]	11.68%	5.62%	13.00%	6.38%

other state-of-the-art DNN models by evaluating with the average end point errors (EPE) and 1-pixel threshold error rates on the test set. The results are presented in Table 3.2. We find that our GA-Net outperforms the state-of-the-arts on both of the two evaluation metrics by a noteworthy margin (2.2% improvement in error rate and 0.25 pixel improvement in EPE compared with the current best PSMNet [32].).

3.4.5.2 KITTI 2012 and 2015 Datasets

After training on Scene Flow dataset, we use the GA-Net-15 to fine-tune on the KITTI 2015 and KITTI 2012 data sets respectively. The models are then evaluated on the test sets. According to the online leader board, as shown in Table 3.4 and Table 3.5, our GA-Net has fewer low-efficient 3D convolutions but achieves better accuracy. It surpasses current best PSMNet in all the evaluation metrics. Examples are shown in Fig. 3.6. The GA-Nets can effectively aggregate the correct matching information into the challenging large textureless or reflective regions to get precise estimations. It also keeps the object structures very well.

3.5 Conclusion

In this paper, we developed much more efficient and effective guided matching cost aggregation (GA) strategies, including the semi-global aggregation (SGA) and the local guided aggregation (LGA) layers for end-to-end stereo matching. The GA layers significantly improve the accuracy of the disparity estimation in challenging regions, such as occlusions, large textureless/reflective regions and thin structures. The GA layers can be used to replace computationally costly 3D convolutions and get better accuracy.

Acknowledgement

Research is mainly supported by Baidu's Robotics and Auto-driving Lab and in part by the ERC grant ERC-2012-AdG 321162-HELIOS, EPSRC grant Seebibyte EP/M013774/1 and EPSRC/MURI grant EP/N019474/1. We would also like to acknowledge the Royal Academy of Engineering. Victor Adrian Prisacariu would like to thank the European Commission Project Multiple-actOrs Virtual Empathic CARegiver for the Elder (MoveCare).

Chapter 4

Domain-invariant Stereo Matching Networks

Feihu Zhang

University of Oxford

Xiaojuan Qi

University of Hong Kong

Ruigang Yang

Baidu Research, Baidu Inc.

Victor Prisacariu

University of Oxford

Benjamin Wah

The Chinese University of Hong Kong

Philip H. S. Torr

University of Oxford

Abstract

State-of-the-art stereo matching networks have difficulties in generalizing to new unseen environments due to significant domain differences, such as color, illumination, contrast, and texture. In this paper, we aim at designing a domain-invariant stereo matching network (DSMNet) that generalizes well to unseen scenes. To achieve this goal, we propose i) a novel "domain normalization" approach that regularizes the distribution of learned representations to allow them to be invariant to domain differences, and ii) an end-to-end trainable structure-preserving graph-based filter for extracting robust structural and geometric representations that can further enhance domain-invariant generalizations. When trained on synthetic data and generalized to real test sets, our model performs significantly better than all state-of-the-art models. It even outperforms some deep neural network models (*e.g*. MC-CNN and DispNet) fine-tuned with test-domain data.

4.1 Introduction

Stereo reconstruction is a fundamental problem in computer vision, robotics and autonomous driving. It aims to estimate 3D geometry by computing disparities between matching pixels in a stereo image pair. Recently, many end-to-end deep neural network models (*e.g*. [355, 32, 128]) have been developed for stereo matching that achieve impressive accuracy on several datasets or benchmarks.

However, state-of-the-art stereo matching networks (supervised [355, 32, 128] and unsupervised [377, 282]) cannot generalize well to unseen data without fine-tuning or adaptation. Their difficulties lie in the large domain differences (such as color, illumination, contrast and texture). As illustrated in Fig. 4.1, the pre-trained

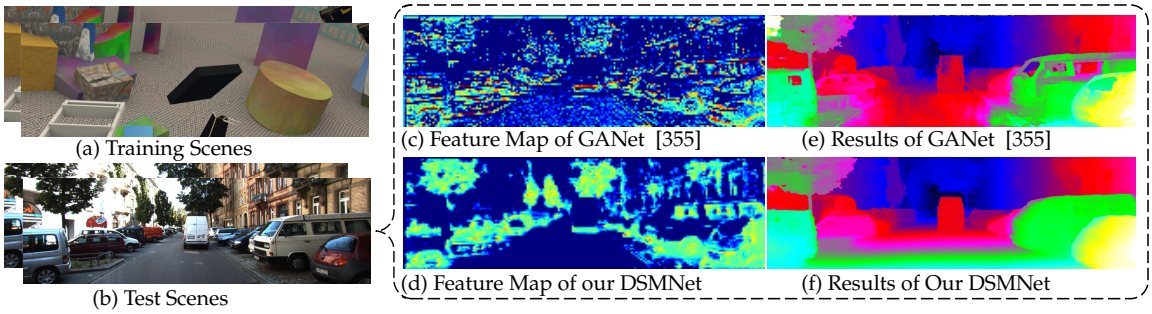


Figure 4.1: Visualization of the feature maps and disparity results. GANet [355] is used for comparisons. The features used for matching (outputs of the feature extraction networks) are visualized in (c) and (d). Models are trained on synthetic data (Sceneflow [187]) and tested on novel real scenes (KITTI [191]). The feature maps from GANet has many artifacts (*i.e.* noise). Our DSMNet mainly captures the structure and shape information as robust features, and there is no distortions or artifacts in the feature map. It can produce accurate disparity estimations in the novel test scenes.

models on one specific dataset produce poor results on other real and unseen scenes.

Domain adaptation and transfer learning methods (*e.g.* [282, 82, 22]) attempt to transfer or adapt from one source domain to another new domain. Typically, a large number of stereo images from the new domain are required for the adaptation. However, these cannot be easily obtained in many real scenarios. Yet, we still need a good method for disparity estimation even without data from the new domain for adaptation.

We focus on the more challenging but crucial domain generalization [14] problem that assumes no access to target information for adaptation or fine-tuning. Namely, we are trying to design a model that can generalize well to unseen data without any re-training or adaptation. The difficulties in developing such a domain-invariant stereo matching network (DSMNet) come from the significant domain differences (Fig. 4.1(a)-(b)) which can be roughly categorized as i) image-level styles (*e.g.* color, illumination), ii) local variations (*e.g.* contrast), iii) texture patterns, details and noise conditions and iv) other complicated domain shifts (*e.g.* uncommon/non-linear

contents). They can be approximated by:

$$f(p) = \alpha_I(\alpha_p \cdot \phi(p) + \beta_p) + \beta_I. \tag{4.1}$$

Here, p is the feature of each pixel (e.g. RGB). Without domain shifts, $f(p) = p$ for different datasets. In practice, domain shifts are varying in different datasets.
The i) image-level style differences can be represented as α_I and β_I. The ii) local variations (e.g. contrasts) are α_p. β_p represents the iii) image details/noise. Pixels of an image have the same α_I and β_I. The local shifts α_p and β_p are varying in different regions/pixels. And ϕ is the expression of iv) other uncommon domain differences that cannot be easily formulated as specific models.

Fig. 4.1 visualizes the features learned by state-of-the-art stereo matching model [355]. Such domain differences make the learned features unstable, distorted and noisy, leading to many wrong matching results (Fig. 4.1(e)) when applied to the novel test data (Fig. 4.1(c)).

In this paper, we propose two novel trainable neural network layers for constructing the DSMNet for cross-domain generalization without fine-tuning or adaptation. The proposed novel **domain normalization (DN)** layer fully regulates the distribution of the feature in both the image-level spatial (height and width) and the pixel-level channel dimensions. It can therefore reduce the domain shifts/differences of i) image-level styles (α_I and β_I in Eq. (4.1)) and ii) local contrast variations (α_p in Eq. (4.1)) between different datasets/scenes. Our non-local **structure-preserving graph-based filtering (SGF)** layer can further smooth and reduce the iii) domain-sensitive local details/noise (β_p in Eq. (4.1)). It also helps capture more robust structural and geometric representations (*e.g.* shape and structure, as in Fig. 4.1(d)) that are more robust to iv) many other complicated domain differences (ϕ in Eq. (4.1)) for stereo reconstruction.

We formulate our method as an end-to-end deep neural network and train it only with synthetic data. In experiments, without any fine-tuning or adaptation on the real test data, our DSMNet far outperforms: 1) almost all state-of-the-art stereo matching models (*e.g*. GANet [355]) trained on the same synthetic dataset, 2) most of the traditional methods (*e.g*. Cosfter filter, SGM [93] *et al*.), 3) most of the unsupervised/self-supervised models trained on the target test domains. Our model even surpasses some of the fine-tuned (on the target domains) supervised neural network models (*e.g*. MC-CNN [352], content-CNN [179], DispNetC [187]). Also, it doesn't sacrifice fine-tuned accuracy for generalization. After fine-tuning on the target scenes, it can achieve state-of-the-art accuracy (*e.g*. on KITTI benchmark). Moreover, our method can be easily extended to the optical flow task. It also significantly improves the generalization abilities of the optical flow networks (*e.g*. FlowNew2 [115], PwcNet [268]).

4.2 Related Work

4.2.1 Deep Neural Networks for Stereo Matching

In recent years, deep neural networks have seen great success in stereo matching [187, 248, 128, 32, 355]. These models can be categorized into three types: 1) learning better features for traditional stereo matching algorithms, 2) correlation-based deep neural networks, 3) cost-volume based stereo matching networks.

In the first category, deep neural networks have been used to compute patch-wise similarity scores as the matching costs [358, 352]. The costs are then fed into the traditional cost aggregation and disparity computation/refinement methods [93] to get the final disparity maps. The models are, however, limited by the traditional matching cost aggregation step and often produce wrong predictions in occluded regions, large textureless/reflective regions and around object edges.

DispNetC [187], a typical method in the second category, computes the correlations by warping between stereo views and attempts to predict the per-pixel disparity by minimizing a regression training loss. Many other sate-of-the-art methods, including iResNet [162], CRL [207], SegStereo [333], EdgeStereo [261], HD^3 [349], and MADNet [282], are all based on color or feature correlations between the left and right views for disparity estimation.

The recently developed cost-volume based models explicitly learn feature extraction, cost volume, and regularization function all end to end. Examples include GC-Net [128], PSM-Net [32], StereoNet [129], AnyNet [305], GANet [355] and EMCUA [203]. They all utilize a similarity cost as the third dimension to build the 4D cost volume in which the real geometric context is maintained.

Others, like [83], combine the correlation and cost volume strategies.

The common feature of these models is that they all require a large number of training samples with ground truths. More importantly, a model trained on one domain cannot generalize well to new scenes without fine-tuning or retraining.

4.2.2 Adaptation and Self-supervised Learning

Self-supervised Learning: A recent trend of training stereo matching networks in an unsupervised manner relies on image reconstruction losses that are achieved by warping left and right views [377, 374]. However, they cannot solve the occlusions and reflective regions where there is no correspondence between the left and the right views. Also, they cannot generalize well to other new domains.

Domain Adaptation: Some methods pre-train the models on synthetic data and then explore the cross-domain knowledge to adapt [82, 208] for a new domain. Others focus on the online or offline adaptations [281, 282, 280, 216]. For example, MADNet [282] is proposed to adapt the pre-trained model online and in real time. But, it has poor accuracy even after the adaptation. Moreover, the domain

adaptation approaches require a large number of stereo images from the target domain for adaptations. However, these cannot be easily obtained in many real scenarios. And, in this case, we still need a good method for disparity estimation even without data from the new domain for adaptation.

4.2.3 Cross-domain Generalization

In contrast to domain adaptation, domain generalization [14, 76] is a much harder problem that assumes no access to target information for adaptation or fine-tuning. There are many approaches that explore the idea of domain-invariant feature learning. Previous approaches focus on developing data-driven strategies to learn invariant features from different source domains [196, 76, 151]. Some recent methods utilize meta-learning that takes variations in multiple source domains to generalize to novel test distributions [14, 152]. Other approaches [157, 153] employ an invariant adversarial network to learn domain-invariant representations for image recognition. Choy *et al*. [57] develop a universal feature learning framework for visual correspondences using deep metric learning.

In contrast to the above approaches, there are methods that try to improve the batch or instance normalization in order to improve the generalization and robustness for style transfer or image recognition [199, 206].

In summary, for stereo matching, work is seldom done to improve the generalization ability of the end-to-end deep neural network models, especially when developing the domain-invariant stereo matching networks.

4.3 Proposed DSMNet

To address the challenges of domain shifts (Eq. 4.1), we propose 1) a novel domain normalization (DN) to remove the influence of the image-level domain shifts (α_I and β_I: *e.g*. color, style, illuminance) and the local contrast variations (α_p in Eq.4.1),

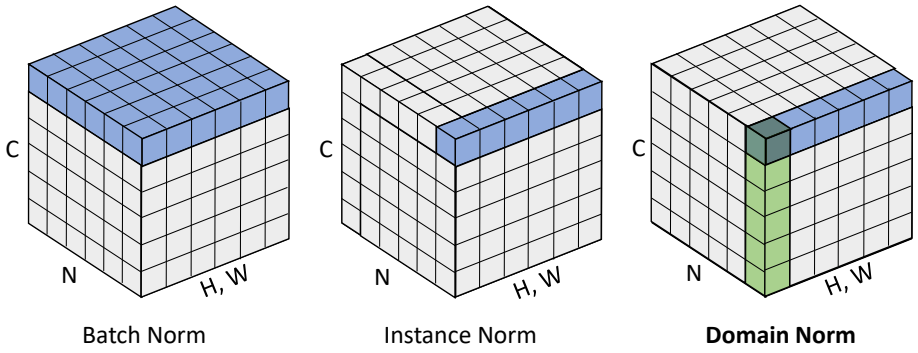


Figure 4.2: Normalization methods. Each subplot shows a feature map tensor, with N as the batch axis, C as the channel axis, and (H, W) as the spatial axes. The blue elements in set S are normalized by the same mean and variance. The proposed domain normalization consists of image-level normalization (blue, Eq. 4.2) and pixel-level normalization of each C-channel feature vector (green, Eq. 4.4).

as well as 2) the trainable structure-preserving graph-based filtering (SGF) layer to smooth the domain-sensitive local noise/details (β_p) and capture the structural and geometric context as robust features for domain-invariant stereo reconstruction.

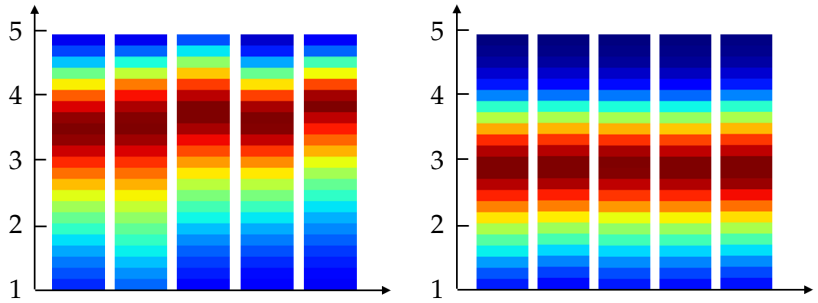
4.3.1 Domain Normalization

Batch normalization (BN) has become the default feature normalization operation for constructing end-to-end deep stereo matching networks [128, 32, 355, 261, 282, 187]. Although it can reduce the internal covariate shift effects in training deep networks, it is domain-dependent and has negative influence on the cross-domain generalization ability.

BN normalizes the features as follows:

$$\hat{x}_i = \frac{1}{\sigma}(x_i - \mu_i). \tag{4.2}$$

Here x and $\hat{x}$ are the input and output features, respectively, and i indexes elements in a tensor (*i.e.* feature maps, as illustrated in Fig. 4.2) of size $N \times C \times H \times W$ (N: batch size, C: channels, H: spatial height, W: spatial width). μ_i and σ_i are the



(a) Instance Norm (b) Domain Norm

Figure 4.3: Norm (α_p of Eq.4.1) distributions of the features for different datasets (left to right: synthetic SceneFlow, KITTI, Middlebury, CityScapes and ETH 3D). The output of the feature extraction network is used for the study. The norm (α_p) of the feature vector at each pixel is counted. Instance normalization can only reduce the image-level differences, but does not normalize the C-channel feature vectors at pixel level.

corresponding channel-wise mean and standard deviation (std) and are computed by:

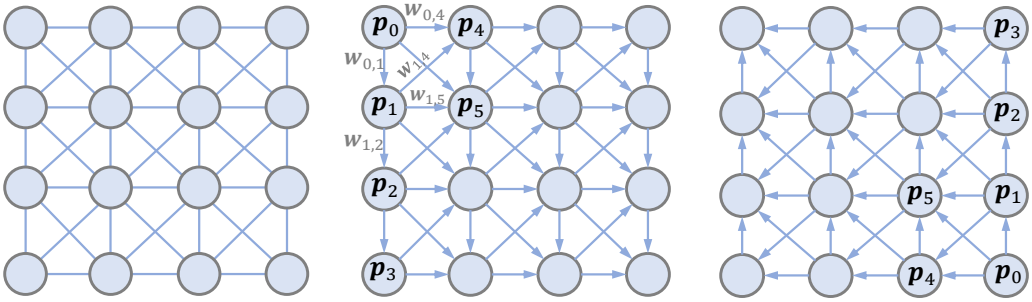
$$\mu_i = \frac{1}{m}\sum_{k\in S_i} x_k, \quad \sigma_i = \sqrt{\frac{1}{m}\sum_{k\in S_i}(x_k - \mu_i)^2 + \epsilon}, \tag{4.3}$$

where S_i is the set of elements in the same channel as element i (Fig. 4.2), and ϵ is a small constant to avoid dividing by zeros.

Mean μ and standard deviation σ are computed per batch in the training phase, and the accumulated values of the training set are utilized for inference. However, different domains may have different μ and σ caused by color shifts, contrast, and illumination. (Fig. 4.1(a)–(b)). Thus μ and σ computed for one dataset are not transferable to others.

Instance normalization (IN) [199, 209] overcomes the dependency on data-set statistics by normalizing each sample separately, where elements in S_i are confined to be from the same sample as illustrated in Fig. 4.2. In theory, IN is domain-invariant, and normalization across the spatial dimensions (H, W) reduces image-level style variations.

However, matching of stereo views is realized at the pixel level by finding an



(a) 8-connected graph (b) directed graph G_1 (c) directed graph G_2

Figure 4.4: Illustration of the graph construction. The 8-way connected graph is separated into two directed graphs G_1 and G_2.

accurate correspondence for each pixel using its C-channel feature vector. Any inconsistence of the feature norm and scaling will significantly influence the matching cost and similarity measurements.

Fig. 4.3 illustrates that IN cannot regulate the norm distribution of pixel-wise feature vectors that vary in datasets/domains.

We propose in Fig. 4.2 our **domain-invariant normalization (DN)**. Our method normalizes features along the spatial axis (H, W) to induce style-invariant representations similar to IN as well as along the channel dimension (C) to enhance the local invariance.

Our DN is realized as follows:

$$\hat{x}_i' = \frac{\hat{x}_i}{\sqrt{\sum_{i \in S_i'} |\hat{x}_i|^2 + \epsilon}}, \tag{4.4}$$

where S_i' (green region in Fig. 4.2) includes C elements from the same example (N axis) and the same spatial location (H, W axis). $\hat{x}_i$ is computed as Eq. (4.2) and (4.3) with elements in S_i from the same channel and sample (blue in Fig. 4.2).

In addition to the normalization across spatial dimension, we also employ L_2 normalization to normalize features along the channel axis. They collaborate with each other to address the sensitivity to both image-level domain shift (α_I and β_I in

Eq. (4.1)) and the local contrast variations (α_p). As illustrated in Fig. 4.3, it helps regulate the norm (α_p) distribution of the features in different datasets and improves the robustness to local contrast variations.

Finally, the trainable per-channel scale γ and shift β are added to enhance the discriminative representation ability as BN and IN. The final formulation is:

$$y_i = \gamma_i \, \hat{x}'_i + \beta_i. \tag{4.5}$$

4.3.2 Structure-preserving Graph-based Filtering

We propose a trainable Structure-preserving Graph-based Filter (SGF) that exploits contextual information and avoid solely memorizing local domain-sensitive texture patterns, details or noise (see Fig. 4.1(c)) for robust stereo matching.

Our inspiration comes from traditional graph-based filters that are remarkably effective in employing structural and geometric information for structure-preserving texture and detail removing/smoothing [353], denoising [353, 43], as well as depth-aware estimation and enhancement [169, 339].

Formulation For a 2D image/feature map I, we construct an 8-connected graph by connecting pixel $\mathbf{p}$ to its eight neighborhoods (see Fig. 4.4). To avoid loops and achieve fast information aggregation over the graph, we split it into two reverse directed graphs G_1, G_2 (see Fig. 4.4(b) and 4.4(c)).

We assign weight ω_e to each edge $e \in G$, and a feature (or color) vector $C(\mathbf{p})$ to each node $\mathbf{p} \in G$. We also allow $\mathbf{p}$ to propagate information to itself with weight $\omega_e(\mathbf{p}, \mathbf{p})$. For graph G_i ($i = 0, 1$), our SGF is defined as follows:

$$C_i^A(\mathbf{p}) \;=\; \frac{\sum\limits_{\mathbf{q}\in G_i} W(\mathbf{q},\mathbf{p})\cdot C(\mathbf{q})}{\sum\limits_{\mathbf{q}\in G_i} W(\mathbf{q},\mathbf{p})}, \qquad W(\mathbf{q},\mathbf{p}) \;=\; \sum_{l_{\mathbf{q},\mathbf{p}}\in G_i} \prod_{e\in l_{\mathbf{q},\mathbf{p}}} \omega_e. \tag{4.6}$$

Here, $l_{\mathbf{q},\mathbf{p}}$ is a feasible path from $\mathbf{q}$ to $\mathbf{p}$. Note that $e(\mathbf{q}, \mathbf{q})$ is included in the path and

counts for the start node $\mathbf{q}$. Unlike traditional geodesic filters, we consider all valid paths from source node $\mathbf{q}$ to target node $\mathbf{p}$. The propagation weight along path $l_{\mathbf{q},\mathbf{p}}$ is the product of all edge weights ω_e along the path. Here weight $W(\mathbf{q},\mathbf{p})$ is defined as the sum of the weights of all feasible paths from $\mathbf{q}$ to $\mathbf{p}$, which determines how much information is diffused to $\mathbf{p}$ from $\mathbf{q}$.

For the edge weight $\omega_{(\mathbf{q},\mathbf{p})}$, we define it in a self-regularized manner as follows:

$$\omega_e(\mathbf{q},\mathbf{p}) = \frac{{\mathbf{x_p}}^T \mathbf{x_q}}{\|\mathbf{x_p}\|_2 \cdot \|\mathbf{x_q}\|_2}, \tag{4.7}$$

where $\mathbf{x_p}$ and $\mathbf{x_q}$ represent the feature vectors of $\mathbf{p}$ and $\mathbf{q}$, respectively.

Compared to other local filters, such as Gaussian filter, median filter, and bilateral filter that can only propagate information in a local region determined by the filter kernel size, our SGF allows the propagation of long-range information over the whole image. More importantly, the filtering weights is defined as a spatial accumulation along all feasible paths in a graph. Similar to Geodesic filter [169] and tree filter [340, 257], this path-based filtering kernel helps better preserve the structures of the feature maps.

For stable training and to avoid extreme values, we further add a normalization constraint to the weights associated with $\mathbf{p}$ in the graph G_i as:

$$\sum_{\mathbf{q}\in N_\mathbf{p}} \omega_{e(\mathbf{q},\mathbf{p})} = 1. \tag{4.8}$$

Here, $N_\mathbf{p}$ is the set of the connected neighbors of $\mathbf{p}$ (including itself), and $e(\mathbf{q},\mathbf{p})$ is the directed edge connecting $\mathbf{q}$ and $\mathbf{p}$. For example, in Fig. 4.4(b), for node $\mathbf{p}_0$, $\omega_{e(\mathbf{p}_0,\mathbf{p}_0)} = 1$; and for node $\mathbf{p}_4$, $\omega_{0,4} + \omega_{1,4} + \omega_{e(\mathbf{p}_4,\mathbf{p}_4)} = 1$.

If Eq. (4.8) holds, we can further derive $\sum_{\mathbf{q}\in G_i} W(\mathbf{q},\mathbf{p}) = 1$*.Eq. (4.6) can then be

*Proof is in the supplementary material: `https://github.com/feihuzhang/DSMNet`

simplified as follows:

$$C_i^A(\mathbf{p}) = \sum_{\mathbf{q}\in G_i} W(\mathbf{q},\mathbf{p})\cdot C(\mathbf{q}), \qquad W(\mathbf{q},\mathbf{p}) = \sum_{l_{\mathbf{q},\mathbf{p}}\in G_i} \prod_{e\in l_{\mathbf{q},\mathbf{p}}} \omega_e. \tag{4.9}$$

Such a transformation not only increases the robustness in training but also reduces the computational costs.

Linear implementation. Eq. (4.9) can be realized as an iterative linear aggregation, where the node is sequentially updated following the direction of the graph (*e.g.* top to bottom, then left to right in G_1). In each step, $\mathbf{p}$ is updated as:

$$\begin{aligned} C_i^A(\mathbf{p}) &= \omega_{e(\mathbf{p},\mathbf{p})}\cdot C(\mathbf{p}) + \sum_{\mathbf{q}\in N_{\mathbf{p}},\mathbf{q}\neq\mathbf{p}} \omega_{e(\mathbf{q},\mathbf{p})}\cdot C_i^A(\mathbf{q}) \\ s.t. &\sum_{\mathbf{q}\in N_{\mathbf{p}}} \omega_{e(\mathbf{q},\mathbf{p})} = 1. \end{aligned} \tag{4.10}$$

Finally, we repeat the aggregation process for G_1 and G_2 where the updated representation with G_1 is used as the input for aggregation with G_2. The aggregation of Eq. (4.10) is a linear process with time complexity of $O(n)$ (with n nodes in the graph). During training, backpropagation can be realized by reversing the propagation which is also a linear process (*refer to the supplementary material*).

Relations to existing approaches. We show that the recently proposed semi-global aggregation (SGA) layer [355] and affinity-based propagation approach [171] are special cases of our SGF (Eq. (4.9)). In addition, we compare it with non-local strategies, [302, 320], graph neural networks [361] and the attention mechanism [109].

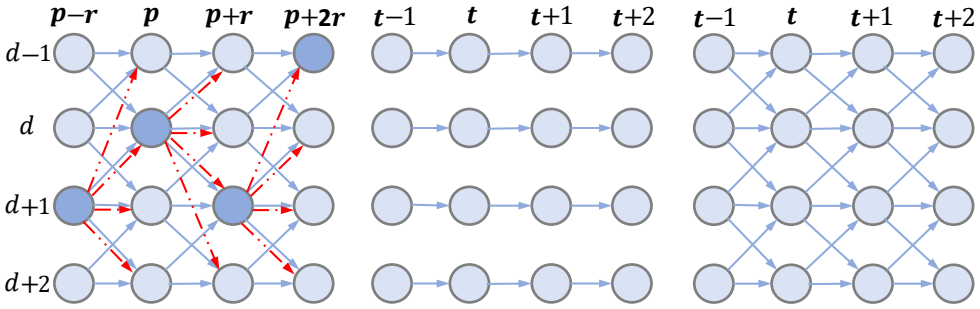


Figure 4.5: Special cases of our graph-based filter. (a) Semi-global aggregation (SGA) layer [355]. The dark blue node represents the maximum of each column. (b) and (c) are the affinity-based spatial propagations [171]. They aggregate from column t to $t+1$.

a) Semi-global Aggregation (SGA) [355] is proposed as a differentiable approximation of SGM [93] and can be presented as follows:

$$C_{\mathbf{r}}^{A}(\mathbf{p},d) \;=\; \text{sum}\begin{cases} \omega_0(\mathbf{p},\mathbf{r})\cdot C(\mathbf{p},d) \\ \omega_1(\mathbf{p},\mathbf{r})\cdot C_{\mathbf{r}}^{A}(\mathbf{p}-\mathbf{r},d) \\ \omega_2(\mathbf{p},\mathbf{r})\cdot C_{\mathbf{r}}^{A}(\mathbf{p}-\mathbf{r},d-1) \\ \omega_3(\mathbf{p},\mathbf{r})\cdot C_{\mathbf{r}}^{A}(\mathbf{p}-\mathbf{r},d+1) \\ \omega_4(\mathbf{p},\mathbf{r})\cdot \max\limits_{i} C_{\mathbf{r}}^{A}(\mathbf{p}-\mathbf{r},i) \end{cases} \quad s.t. \sum_{i=0,1,2,3,4}\omega_i(\mathbf{p},\mathbf{r})=1 \tag{4.11}$$

The aggregations are in four directions, namely $\mathbf{r}=\{(0,1),(0,-1),(1,0),(-1,0)\}$. Taking the right to left propagation ($\mathbf{r}=(0,1)$) as an example, we can construct a propagation graph in Fig. 4.5(a). The y-coordinate represents disparity d, and the x-coordinate is the indexes of the pixels/nodes. Compared to our graph in Fig. 4.4(b), edges connecting top and bottom nodes are removed, and the maximum of each column is densely connected to every node of the next column (red edges). Eq. (4.11) can then be realized by our SGF of Eq. (4.9). Here, $(\mathbf{p}-\mathbf{r},d\pm1)$ are the neighborhood nodes of $\mathbf{p}$, and $\omega_{0,\dots4}$ are the corresponding edge weights.

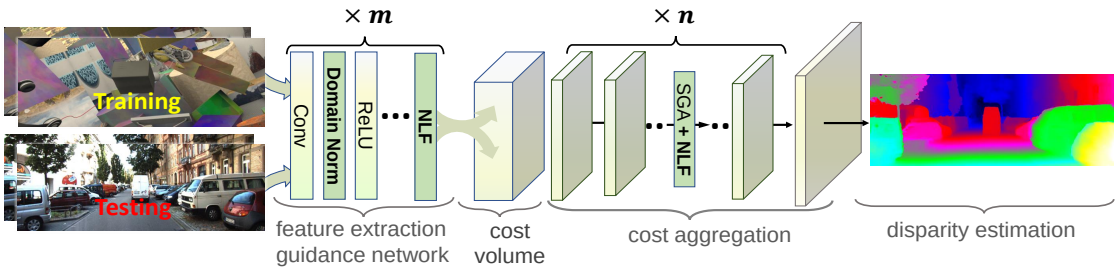


Figure 4.6: Overview of the network architecture. Synthetic data are used for training, while using data from other new domains (*e.g.* real KITTI dataset) for testing. The backbone of GANet [355] is used as the baseline. The proposed DN layer is used after each convolutional layer in the feature extraction and guidance network. Several SGF layers are implemented for both feature extraction and cost aggregation.

b) Affinity-based Spatial Propagation in [171] can be achieved as:

$$C^A(\mathbf{p},d) \;=\; \left(1-\sum_{\mathbf{q}\in N_\mathbf{p},\mathbf{q}\neq\mathbf{p}}\omega_{e(\mathbf{q},\mathbf{p})}\right)C(\mathbf{p}) \;+\; \sum_{\mathbf{q}\in N_\mathbf{p},\mathbf{q}\neq\mathbf{p}}\omega_{e(\mathbf{q},\mathbf{p})}C^A(\mathbf{q}), \tag{4.12}$$

where $\omega_{e(\mathbf{q},\mathbf{p})}$ are the learned affinities. $1-\sum_{\mathbf{q}\in N_\mathbf{p}}\omega_{e(\mathbf{q},\mathbf{p})}$ is equal to our weight $\omega_{\mathbf{e}(\mathbf{p},\mathbf{p})}$ for $\mathbf{p}$. The graphs for filtering can be constructed as in Fig. 4.5(b) and 4.5(c) for the one-way and three-way propagations [171], respectively.

c) Non-local Strategies, Graph Neural Networks and Attentions [302, 361, 320, 109, 250] can be used for non-local feature aggregation. But, they are implemented without spatial and structural awareness. Existing attentions and GNNs used in image segmentation task only consider the feature similarity for aggregation which treat pixel locations equally. In geometric problem (*e.g.* stereo matching), spatial proximity is crucial for learning accurate depths since pixels in the same object/class (with similar features) must be spatially close enough to have similar depth values. Therefore, these similarity/affinity based attentions and non-local networks will easily smooth out depth edges and thin structures (as illustrated in the supplementary material). Our SGF utilizes both the feature affinity and the spatial proximity for non-local graph-based filtering. It spatially aggregates the

features along the paths which can better preserve the structure of the disparity maps. More importantly, Our graph filter has lower (linear) complexity in both memory requirement and computation since it is realized by the linear spatial propagation and the weight matrix is only $5 \times N$.

4.3.3 Network Architecture

As illustrated in Fig. 4.6, we utilize the backbone of GANet as the baseline architecture. The LGA layer in [355] is removed since it's domain-dependent and captures a lot of local patterns that are very sensitive to domain shifts.

We replace the original batch normalization layer by our proposed domain normalization layer for feature extraction. For the feature extraction network, we utilize a total of seven proposed filtering layers. For 3D cost aggregation of the cost volume, two SGF layers are further added for cost volume filtering in each channel/depth. *Details of the architecture are in the supplementary.*

4.4 Experimental Results

In our experiments, we train our method only with synthetic data and test it on four real datasets to evaluate its domain generalization ability. During training, we use disparity regression [128] for disparity prediction, and the smooth L_1 loss to compute the errors for back-propagation (the same as in [355, 32]). All the models are optimized with Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$). We train with a batch size of 8 on four GPUs using 288×624 random crops from the input images. The maximum of the disparity is set as 192. We train the model on the synthetic dataset for 10 epochs with a constant learning rate of 0.001. All other training settings are kept the same as those in [355].

4.4.1 Datasets

KITTI stereo 2012 [75] and 2015 [191] datasets provide about 400 image pairs of outdoor driving scenes for training, where the disparity labels are transformed from Velodyne LiDAR points. The **Cityscapes** dataset [58] provides a large amount of high-resolution ($1k \times 2k$) stereo images collected from city driving scenes. The disparity labels are pre-computed by SGM [93] which is not accurate enough for training deep neural network models. The **Middlebury** stereo dataset [242] is designed for indoor scenes with higher resolution (up to $2k \times 3k$). But it provides no more than 50 image pairs that are not enough to train robust deep neural networks. In addition, **ETH 3D** dataset [245] provides 27 pairs of gray images for training.

These existing real datasets are all limited by their small quantity or poor ground-truth labels, making them insufficient for training. Hence, we use them as test sets for evaluating our models' cross-domain generalization ability.

We mainly use synthetic data to train our domain-invariant models. The existing Scene Flow synthetic dataset [187] contains 35k training image pairs with a resolution of 540×960. This dataset has a limited number of the outdoor driving scenes that provide stereo pairs with a few settings of the camera baselines and image resolutions. We use CARLA [65] to generate a new supplementary synthetic dataset (with 20k stereo pairs) with more diverse settings, including two kinds of image resolutions (720×1080 and 1080×1920), three different focal lengths, and five different camera baselines (in a range of 0.2–1.5m). This supplementary dataset* can significantly improve the diversity of the training set.

The two advantages in using synthetic data are that it can avoid all the difficulties of labeling a large amount of real data, and that it can eliminate the negative influence of wrong depth values in real datasets.

*Available at `https://github.com/feihuzhang/DSMNet`.

Table 4.1: Ablation study. Models are trained on synthetic data (SceneFlow). Threshold error rates (%) are used for evaluations.

Normalization	SGF		Backbone	Middlebury	KITTI
	feature	cost volume		2-pixel	3-pixel
BN			ours	30.3	9.4
DN			ours	27.1	7.9
DN	+3		ours	24.2	7.1
DN	+7		ours	22.9	6.8
DN	+9		ours	22.4	6.8
DN	+7	+2	ours	**21.8**	**6.5**
BN			PSMNet	39.5	16.3
BN			GANet	32.2	11.7
DN	+7	+2	PSMNet	26.1	8.5
DN	+7	+2	GANet	23.7	7.3

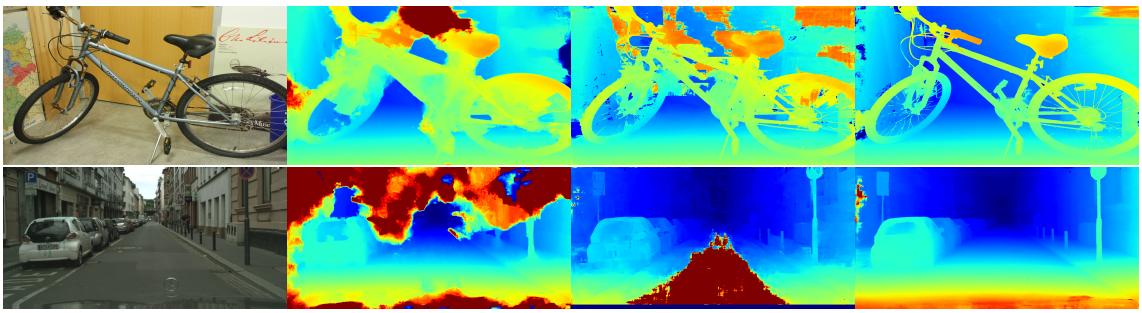
4.4.2 Ablation Study

We evaluate the performance of our DSMNet with numerous settings, including different architectures, normalization strategies and numbers (0–9) of the proposed SGF layers. As listed in Table 4.1, the full-setting DSMNet far outperforms the baseline in accuracy by 3% on the KITTI and 8% on the Middlebury datasets. Our proposed domain normalization improves the accuracy by about 1.5%, and the SGF layers contribute another 1.4% on the KITTI dataset.

Moreover, our proposed layers are generic and could be seamlessly integrated into other deep stereo matching models. Here, we replace our backbone model with GANet [355] and PSMNet [32]. The accuracies are improved by 4–8% on KITTI dataset and 8–13% on Middlebury dataset for coss-domain evaluations compared with the original PSMNet and GANet.

4.4.3 Component Analysis and Comparisons

To further validate the superiorities of the proposed layers, we compare each of them with other related normalization and attention/affinity strategies.



(a) Input view (b) HD^3 [349] (c) PSMNet [32] (d) Our DSMNet

Figure 4.7: Comparisons with state-of-the-art models. Models are trained on synthetic data and evaluated on high-resolution real datasets (Middlebury and CityScapes). Our DSMNet can produce much more accurate disparity estimation.

Table 4.2: Comparisons with Existing Normalization and Filtering/Attention Strategies

Normalization	Middlebury	KITTI	Affinity/Attention	Middlebury	KITTI
Batch Norm	29.1	7.3	Attention [109]	25.2	5.9
Instance Norm	27.1	6.4	Denoising [320]	25.9	6.1
Adaptive Norm [199]	28.2	6.8	Affinity [171]	23.1	5.2
Our Domain Norm	20.1	4.1	Our Graph Filter	20.1	4.1

Normalization strategies. Table 4.2 compares our domain normalization with batch normalization [117], instance normalization [288], and the recently proposed adaptive batch-instance normalization [199]. There are also some adaptive BNs [158, 159] for domain adaptation, but they require the full access to the target dataset and cannot be used for more challenging domain generalization task. We keep all other settings the same as our DSMNet and only replace the normalization method for training and evaluation. Our domain normalization is superior to others for domain-invariant stereo matching because it can fully regulate the distribution of the feature vectors and remove both image-level and local contrast differences for cross-domain generalization.

Attentions and non-local approaches. Finally, we compare our SGF with attention and non-local networks, including affinity-based propagation [171], non-local neural network denoising [320], and non-local attention [109] (in Table 4.2). Our

Table 4.3: Evaluations on the KITTI, Middlebury, and ETH 3D validation datasets. Threshold error rates (%) are used.

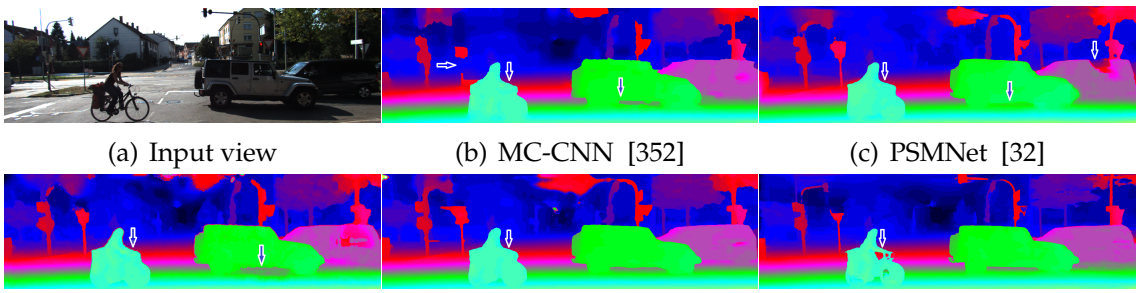
Models	KITTI		Middlebury			ETH3D	Carla
	2012	2015	full	half	quarter		
CostFilter [98]	21.7	18.9	57.2	40.5	17.6	31.1	41.1
PatchMatch [21]	20.1	17.2	50.2	38.6	16.1	24.1	30.1
SGM [93]	7.1	7.6	38.1	25.2	10.7	12.9	20.2
Training set	SceneFlow						
HD^3 [349]	23.6	26.5	50.3	37.9	20.3	54.2	35.7
gwcnet [83]	20.2	22.7	47.1	34.2	18.1	30.1	33.2
PSMNet [32]	15.1	16.3	39.5	25.1	14.2	23.8	25.9
GANet [355]	10.1	11.7	32.2	20.3	11.2	14.1	18.8
Our DSMNet	**6.2**	**6.5**	**21.8**	**13.8**	**8.1**	**6.2**	**9.8**
Training set	SceneFlow + Carla						
HD^3 [349]	19.1	19.5	47.3	35.2	19.5	45.2	–
gwcnet [83]	17.2	18.1	45.2	31.8	17.2	29.4	–
PSMNet [32]	10.3	11.0	35.5	23.7	13.8	20.3	–
GANet [355]	7.2	7.6	31.9	19.7	11.4	13.5	–
Our DSMNet	**3.9**	**4.1**	**20.1**	**13.6**	**8.2**	**6.0**	–

SGF layer is better for capturing the structural and geometric context for robust domain-invariant stereo matching. The non-local neural network denoising [320] and non-local attention [109] do not have spatial constraints that usually lead to smoothness of the depth edges (as shown in the supplementary material). Affinity-based propagations [171] are special cases of our proposed SGF and are not as effective in feature and cost volume aggregations for stereo matching.

4.4.4 Cross-domain Evaluations

In this section, we compare our DSMNet with state-of-the-art stereo matching models by training with synthetic data and evaluating on real test sets.

Comparisons with state-of-the-art models. In Table 4.3 and Fig. 4.7, we compare our DSMNet with other state-of-the-art neural network models on the four real datasets. All models are trained on synthetic data (either SceneFlow or a mixture of SceneFlow and Carla). We find that DSMNet far outperforms the state-of-the-art models by 3$\sim$30% in error rates for all these datasets. It is also far better than



(a) Input view (b) MC-CNN [352] (c) PSMNet [32]

(d) HD^3 [349] (e) GANet-deep [355] (f) Our DSMNet-synthetic

Figure 4.8: Comparisons with the fine-tuned state-of-the-art models. Our model is trained only with synthetic data. All others are fine-tuned on the KITTI target scenes. As pointed by arrows, our DSMNet can produce more accurate object boundaries.

Table 4.4: Cross-domain evaluation on KITTI 2015 benchmark (all area). DSMNet is trained only with synthetic data.

Models	Training Set	Error Rate (%)
Our DSMNet	**Synthetic**	**3.71**
MC-CNN-acrt [352]	Kitti-gt	3.89
DispNetC [187]	Kitti-gt	4.34
Content-CNN [179]	Kitti-gt	4.54
MADNet-finetune[282]	Kitti-gt	4.66
Weak Supervise [286]	Kitti-gt	4.97
MADNet [282]	Kitti (no gt)	8.23
OASM-Net [150]	Kitti (no gt)	8.98
Unsupervised [377]	Kitti (no gt)	9.91

Table 4.5: In-domain (after fine-tuning) evaluation (error rates: %) on the KITTI 2015 Benchmark

Models	Non-Occluded	All Area
GANet + **Our SGF**	**1.58**	**1.77**
GANet-deep [355]	1.63	1.81
DSMNet-finetune	1.71	1.90
AcfNet [363]	1.72	1.89
GANet-15 [355]	1.73	1.93
HD^3 [349]	1.87	2.02
gwcnet-g [83]	1.92	2.11
PSMNet [32]	2.14	2.32
GCNet [128]	2.61	2.87

traditional algorithms, like SGM [93], costfilter [98] and patchmatch [21].

Evaluation on the KITTI benchmark. Table 4.4 presents the performance of our DSMNet on the KITTI benchmark [191]. Our model far outperforms most of the unsupervised/self-supervised models trained on the KITTI domain. It is even better than supervised stereo matching networks (including, MC-CNN [352], content-CNN [179], and DispNetC [187]) trained or fine-tuned on the KITTI dataset. When compared with other fine-tuned state-of-the-art models (*e.g.* PSMNet [32], HD^3 [349], GANet-deep [355]), our DSMNet (without fine-tuning) produces more accurate object boundaries (Fig. 4.8).

Table 4.6: Evaluations of the Optical Flow Networks for Cross-domain Generalization

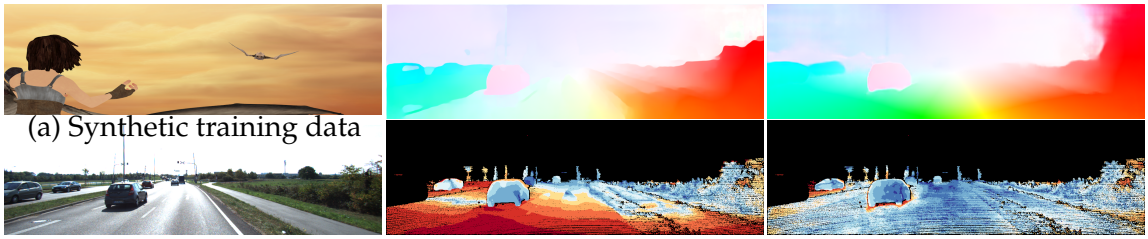
Original Models	Error Rates (%)	Improved Models	Error Rates (%)
FlowNet2 [355]	34.1	Domain-invaraint FlowNet2	16.2
PwcNet [268]	16.9	Domain-invaraint PwcNet	11.2

4.4.5 Fine-tuning

In this section, we show the best performance of our DSMNet when fine-tuned on the target domain. We fine-tune the model pre-trained on synthetic data for a further 700 epochs using the KITTI 2015 training set. The learning rate begins at 0.001 for the first 300 epochs and decreases to 0.0001 for the rest. The results of the test set are submitted to KITTI 2015 benchmark for evaluations.

Table 4.5 compares the results of the fine-tuned DSMNet and those of other state-of-the-art DNN models. We find that DSMNet outperforms most of the recent models (including PSMNet [32], HD^3 [349], GwcNet [83] and GANet-15 [355]) by a noteworthy margin. This implies that DSMNet can achieve the same accuracy by fine-tuning on one specific dataset, without sacrificing accuracy to improve its generalization ability.

We also separately test the effectiveness of our SGF layer. Using the current best "GANet-deep" [355] (including the Local Guided Aggregation layer) as the baseline, we add five filtering layers for feature extraction. All other settings are kept the same as the original GANet. After training on synthetic data and fine-tuning on the KITTI training dataset, the new model got a new state-of-the-art accuracy (1.58%) and **ranked No. 1** on KITTI 2015 benchmark (non-occluded area, by the time of submission). This shows that our SGF can improve not only cross-domain generalization but also the accuracy on the test domains.



(a) Synthetic training data

(b) Real test view (c) FlowNet2 [115] (d) Domain-invariant FlowNet2

Figure 4.9: Performance illustration with optical flow. Models are trained only with synthetic data. (a) Example of the synthetic training data (MPI Sintel [25]), (b) the real test view from KITTI 2015 dataset, (c) the result (top) and the error map (bottom) of the original FlowNet2, (d) the result (top) and the error map (bottom) of the domain-invariant FlowNet2 powered by our DSMNet.

4.5 Extension for Optical Flow

Similar to stereo matching, optical flow is also based on pixel-to-pixel similarity measurement for dense correspondence matching between two different images. Therefore, our domain-invariant matching network can be easily extended to the optical flow task. We use FlowNet2 [115] and PwcNet [268] as baselines and employ our DN and graph filtering to to realize the domain-invariant optical flow networks. The models are trained on synthetic FlyingThings3D [187] and MPI Sintel [25] datasets and evaluated on real flow dataset (KITTI 2015). As shown in Table 4.6 and Fig. 4.9, Accuracies are significantly improved by 5.7–17% in the cross-domain evaluations. This further demonstrates the effectiveness of our proposed domain-invariant network.

4.6 Conclusion

In this paper, we proposed two end-to-end trainable neural network layers for our domain-invariant stereo matching network. Our novel domain normalization can fully regulate the distribution of learned features to address significant domain shifts, and our SGF can capture more robust non-local structural and geometric

features for accurate disparity estimation in cross-domain situations. We have verified our model on four real datasets and shown its superior accuracy when compared to other state-of-the-art models in the cross-domain generalization.

Acknowledgement

Research is supported by Baidu, the ERC grant ERC-2012-AdG 321162-HELIOS, EPSRC grant Seebibyte EP/M013774/1 and EPSRC/MURI grant EP/N019474/1. We would also like to acknowledge the Royal Academy of Engineering.

Chapter 5

Separable Flow: Learning Motion Cost Volumes for Optical Flow Estimation

Feihu Zhang

University of Oxford

Oliver J. Woodford

Victor Prisacariu

University of Oxford

Philip H. S. Torr

University of Oxford

Abstract

Full-motion cost volumes play a central role in current state-of-the-art optical flow methods. However, constructed using simple feature correlations, they lack the ability to encapsulate prior, or even non-local knowledge. This creates artifacts in poorly constrained ambiguous regions, such as occluded and textureless areas. We propose a separable cost volume module, a drop-in replacement to correlation cost volumes, that uses non-local aggregation layers to exploit global context cues and prior knowledge, in order to disambiguate motions in these regions. Our method leads both the now standard Sintel and KITTI optical flow benchmarks in terms of accuracy, and is also shown to generalize better from synthetic to real data.

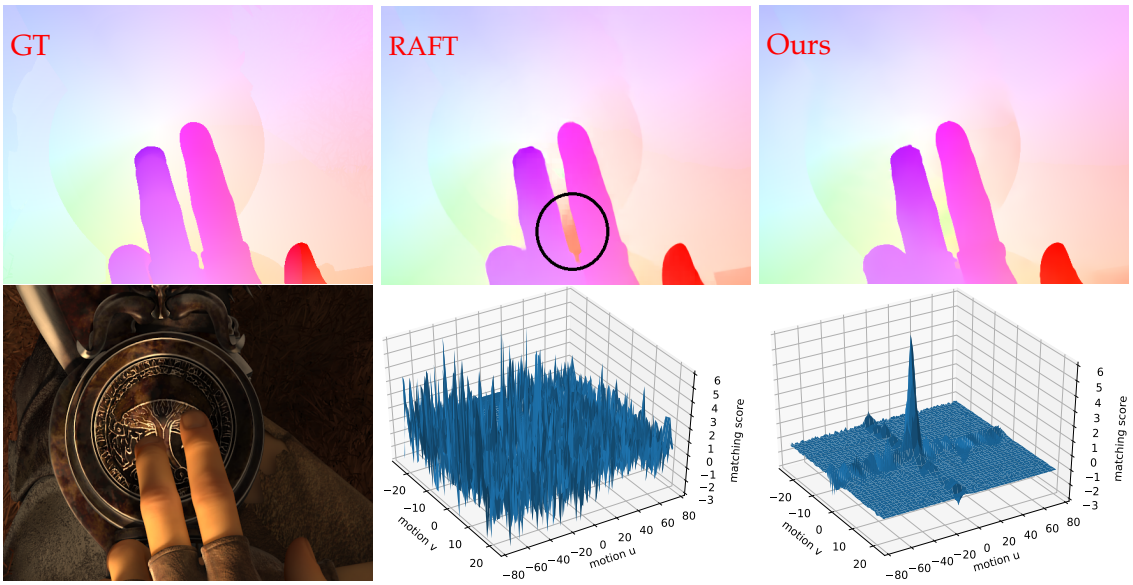
5.1 Introduction

Optical flow is the task of estimating per-pixel 2D motion between two images or video frames. This low-level vision task is a fundamental building block of many higher level tasks, such as object tracking, scene reconstruction and video compression. A common approach to this task, used in both hand designed [97, 20] and more modern deep-learning methods [274, 268], is to first compute a cost volume for motions of all pixels, then use this to infer or refine a motion per pixel. While state-of-the-art methods [274, 332] tend to use this approach, it suffers from two key challenges. First, the cost volume size is exponential in the dimensionality of the search space. Therefore memory and computation requirements for optical flow, with its 2D search space, grow quadratically with the range of motion. In contrast, such costs for the 1D stereo matching task grow only linearly with the range of disparity. Secondly, resolving ambiguities caused by occlusion, lack of

texture, or other such issues requires a more global, rather than local, understanding of the scene, as well as prior knowledge. Cost volumes generally do not encapsulate such information, leaving the job of resolving such ambiguities to the second stage of each method. As Fig. 5.1 & 5.4 illustrate, this makes it harder to compute accurate motion in such regions.

This work proposes a new separable cost volume computation module, which plugs into existing cost-volume-based optical flow frameworks, with two key innovations that address these challenges. The first is to separate the 2D motion of optical flow into two independent 1D problems, horizontal and vertical motion, compressing the 4D cost volume into two smaller 3D volumes using a self-adaptive separation layer. This factored representation significantly reduces the memory and computing resources required to infer (and thus also learn) the cost volumes, making them linear in the range of motion, without loss in accuracy. Moreover, it enables the second innovation: the use of non-local aggregation layers to learn a refined cost volume. Such layers have previously been used for 1D stereo problems [355, 356], where they improve both accuracy in ambiguous regions, and cross-domain generalization. We apply them here to optical flow for the first time, learning cost volumes with non-local, prior knowledge via a one-step motion regression that is able to predict a low-resolution (*i.e.* 1/8), but high-quality motion. This prediction also serves as a better input to the interpolation and refinement module.

We train and evaluate our Separable Flow module on the standard Sintel [25] and KITTI [74] optical flow datasets. We achieve the current best accuracy among all published optical flow methods on both these benchmarks. Moreover, in the cross-domain case of training on synthetic and testing on real data (*i.e.* KITTI), our results improve the previous state of the art by a greater margin, even outperforming some DNN models (*e.g.* FlowNet2 [115] and PWC-Net [268]) finetuned on the target KITTI scenes. We provide an ablation study to show how much of this improvement



(a) Input & ground truth (b) RAFT output & cost volume (c) Our output & cost volume

Figure 5.1: Performance illustrations. (a) Input view from Sintel and the ground truth optical flow. (b) The optical flow result and 2D motion cost volume (for a single pixel in the circled region) of the state of the art, RAFT [274]. (c) Result and cost volume (for the same pixel) learned by our Separable Flow. RAFT does not predict motion accurately in the ambiguous regions, such as occlusions (highlighted by the circle). Indeed, there are many false peaks in the cost volume for this region. In contrast, Separable Flow predicts accurate flow results in these challenging regions, by integrating separable, non-local matching cost aggregations. The resulting learned cost volume has one large peak, that correctly matches the ground truth. See sec. 5.4.2 for more details.

is attributable to each of our contributions. We reiterate that any optical flow framework that computes a cost volume can benefit from these improvements.

5.2 Related Work

We now review prior work related to our method, with a focus on traditional and neural-network-based optical flow, and cost aggregation methods in stereo.

5.2.1 Traditional Approaches

There are three main types of traditional optical flow method. The first is usually based on local filtering [98], interpolation [232, 102, 342], nearest neighbor search [11, 246, 186, 103, 182] or dense inverse search [137].

The second usually optimizes a global energy function that consists of a local matching cost data term and an MRF-based smoothness regularization term, using gradient-based solvers [97, 20, 350, 310, 23, 223, 226].

Methods of the third type use discrete solvers [192, 40, 324] to find more globally optimal solutions to the global energy function. However, large motion ranges mean each pixel can be paired with any of thousands of discrete correspondences, leading to a huge search space. To address this issue, Menez *et al*. [192] prune the search space using feature descriptors, and optimize using message passing, whereas Chen *et al*. [40] use a distance transform to solve the global optimization problem over the full search space.

5.2.2 Deep Neural Networks for Optical Flow

A multitude of deep neural networks (DNNs) have been proposed to infer optical flow between a pair of frames, addressing many different aspects of the task. These include occlusion handling [369], robust loss functions [73, 12], feature representations [247, 358], refinement/interpolation [386, 113, 274], uncertainty estimation [114], lightweight architecture [111], data resampling [15], and motion estimation in dark scenes [372]. Several works jointly learn segmentation and optical flow [249, 9, 316, 52, 316], segmenting the image into objects or backgrounds and computing motion depending on the region type. Coarse-to-fine processing has emerged as a popular ingredient in many recent works [268, 349, 224, 111, 112, 113, 332, 96, 15, 369]. Self-supervised optical flow networks [343, 307, 118, 170, 125, 375, 168, 116] and semi-supervised frameworks [327, 141] have also been explored.

Among these methods, explicit cost volumes appear frequently, [98, 268, 274, 110, 96, 295, 332, 178], storing the data matching costs for each pixel's potential correspondences, and thus playing an important role in generating accurate flow fields. For example, PWC-Net [268] develops a DNN model using image pyramids, warping, and cost volumes. Xiao *et al*. [319] learn cost volumes using the Cayley representation, but without effective cost aggregations. Hui *et al*. [110] address the ambiguous matching challenge by improving the cost volume through an adaptive modulation prior, exploiting local flow consistency. Hofinger *et al*. [96] improve the cost volume construction process via a sampling-based strategy that revises the gradient flow across pyramid levels. Wang *et al*. [295] reshape a 4D cost volume into 3D via a displacement-aware projection (DAP) layer, learning the high-dimensional cost volume with low-dimensional convolutions. However, it can only process a fixed and small displacement range (*e.g*. -3,..,3). Yang *et al*. [332] propose a 5D volumetric encoder-decoder architecture with separable volumetric filtering. Designed for a local search window (*e.g*. -9,..,9), it cannot capture non-local knowledge in the cost volume.

In contrast to these methods, ours can learn and refine a full-range cost volume over the whole motion space, using non-local aggregations, as a result of our Separable Flow model. This is similar to Xu *et al*. [324], who construct a 4D cost volume using DNN features and apply improved semi-global matching [93] for cost aggregations. This strategy is impractical for end-to-end training of DNNs, since the cost aggregation step is not differentiable, and incurs huge memory and computational costs. The current state-of-the-art optical flow model, RAFT [274] also builds multi-scale 4D correlation volumes for all pairs of pixels. However, limited by its huge memory and computational costs, RAFT does not apply any cost aggregation to the 4D volume.

5.2.3 Cost Volumes in Stereo Matching

Full-range cost volumes built over the whole displacement space have been widely used in state-of-the-art stereo matching DNNs [355, 32, 128, 356, 54, 66]. Matching cost aggregation in cost volumes has also become a critical component in stereo matching [355, 128], since local, feature-based matching is often ambiguous due to occlusions, repetitive or homogeneous texture, reflections, noise *etc*. Based on the full-range cost volume, several cost aggregation approaches have been developed, such as geometry and context networks [128], and pyramid matching networks [32] that use 3D convolutions with a pyramidal encoder-decoder for cost volume learning, and guided aggregation networks [355] that use non-local, semi-global matching layers for non-local cost aggregations. Our Separable Flow motion representation makes it possible to use these effective local and non-local matching cost aggregation layers to learn a better cost volume for optical flow estimation.

5.3 Method

This section first describes the prototypical optical flow framework to which our Separable Flow module can be applied, then details the module itself, and finally presents the method used to train it.

5.3.1 Prototypical cost-volume-based optical flow

Cost volume based optical flow methods [268, 274] usually consist of the following stages: 1) image feature extraction, 2) cost volume computation and 3) motion refinement. Our work addresses stage two by introducing the separable cost volume and the cost aggregation modules. We briefly describe the common blocks in existing approaches, but refer the reader to prior works [268, 274] for the full details.

Image feature extraction. A convolutional network (*e.g*. ResNet [274]) is trained

to extract per-pixel, local features from an image, and produces a feature tensor, $F \in \mathbb{R}^{H \times W \times D}$, where $F(i, j)$ is the D-dimensional feature of the pixel at location i, j.

Cost volume computation. Given the feature tensors F_1 and F_2 of the two optical flow images, a cost volume, $\mathbf{C} \in \mathbb{R}^{H \times W \times |U| \times |V|}$ is computed, where $U = \{u_{min}, .., 0, .., u_{max}\}$ and $V = \{v_{min}, .., 0, .., v_{max}\}$ are the sets of discrete horizontal and vertical motions considered for each pixel. Each entry in the 4D volume is typically [268, 274] computed for pixel i, j and pixel motion u, v via a dot product of feature vectors, thus:

$$C(i, j, u, v) = F_1(i, j) \cdot F_2(i + u, j + v) \tag{5.1}$$

Using this approach, higher "costs" represent greater similarity. Our work proposes a new way to represent and compute this cost volume, as described in section 5.3.2.

Motion refinement. Motion is estimated through iterative updates, usually in a coarse-to-fine framework [268, 274, 295]. The update layers take as input the current motion estimate, the cost volume, and context features, and output an additive motion update. Motion is usually initialized to zero. This work uses regression (sec. 5.3.2) to better initialize motion.

5.3.2 Separable Flow

We propose to replace the purely correlation-based cost volume of previous optical flow methods with an efficient, separable cost volume. Our Separable Flow module consists of the following three stages, functionally described below: self-adaptive cost separation, non-local cost aggregation, and motion regression. Fig. 5.2 provides a high level schematic of the design, while parameter and layer settings of the whole architecture can be found in the supplementary material.

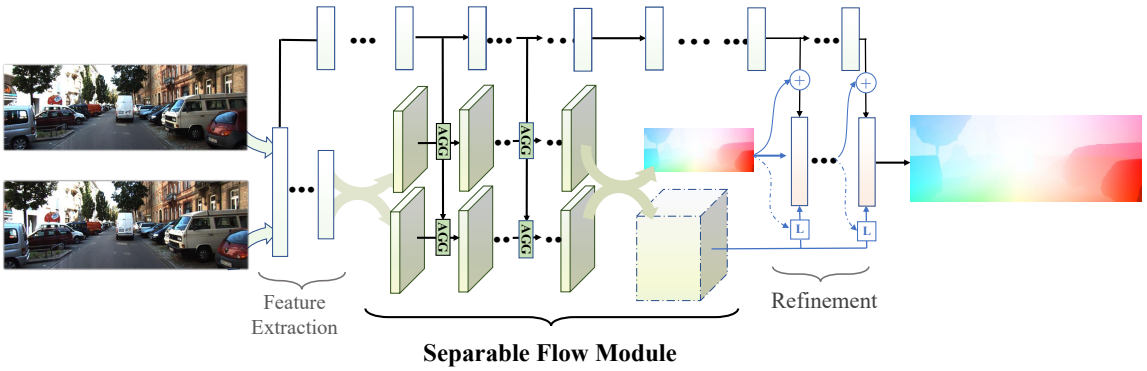


Figure 5.2: Architecture overview. Our model consists of three main parts: 1) feature extraction network, 2) our Separable Flow module of cost volume separation and aggregation networks, and 3) refinement modules. The top layers are a context network [274] that learns weights and context information for cost aggregations, refinement and upsampling (some models do without this). Our Separable Flow module separates the 4D motion cost volume generated from the features into two independent 3D displacement cost volumes. These volumes go through several non-local aggregation layers, as shown. The refined volumes, plus an initial flow estimate regressed from them, are input into the refinement network for further coarse-to-fine improvement and interpolation.

Self-adaptive cost separation In order to improve memory and computational efficiency, and enable non-local aggregation in a learned cost volume, we separate and compress the 4D cost volume, $\mathbf{C}$, into two 3D, K-dimensional feature tensors, $\mathbf{C}_u \in \mathbb{R}^{H\times W\times|U|\times K}$ and $\mathbf{C}_v \in \mathbb{R}^{H\times W\times|V|\times K}$, where $K << |U|, |V|$, representing horizontal and vertical motion respectively.

The first two channels (indexed by superscripts) of $\mathbf{C}_u$ are computed as:

$$C_u^1(i,j,u) = \frac{1}{|V|}\sum_{v\in V} C(i,j,u,v), \tag{5.2}$$

$$C_u^2(i,j,u) = \max_{v\in V} C(i,j,u,v). \tag{5.3}$$

Since mean and maximum select predetermined values of the cost volume, we propose to learn an adaptive selection for the remaining $K-2$ channels, with an attention module. Using the first two channels of the compressed $\mathbf{C}_v$ for efficiency,

this self-adaptive compression is realized by

$$\mathbf{A_u} = \phi_u(\mathbf{C}_v^{1:2}), \quad \in \mathbb{R}^{H \times W \times |V| \times K-2} \tag{5.4}$$

$$C_u^{k+2}(i,j,u) = \sigma\left(A_u^k(i,j)\right) \cdot C(i,j,u,:), \tag{5.5}$$

where ϕ_u is a single, 3D convolutional layer, and $\sigma(\cdot)$ represents the softmax operation. Note that $\mathbf{C}_u$ can be computed without storing the intermediate 4D cost volume $\mathbf{C}$. A similar approach is used to compute $\mathbf{C}_v$. Here we use $K = 4$.

This adaptive compression has several advantages over mean, maximum or convolutional compression. Convolutions, for example, require a fixed range of $|U|$ and $|V|$, while our method can handle variable search spaces. More importantly, convolutions are translationally invariant, but motion varies spatially. Our attention module outputs translationally varying weights, allowing it to adapt to different motions, learning better cost volume representations.

Learning cost aggregation Semi-global matching aggregates non-local information in traditional stereo [93], and more recently optical flow [324], methods. Similarly effective aggregation layers have since been applied to neural networks for stereo matching [52, 128, 355], to great effect, but have not yet been shown to be practical for optical flow networks. However, our separable framework enables us to apply these aggregation layers directly to separated 2D motion.

Our cost aggregation module uses an encoder-decoder architecture that consists of four non-local, semi-global aggregation (SGA) layers, proposed in GANet [355], and eight 3D convolutional layers, to refine $\mathbf{C}_u$ from a $H \times W \times |U| \times K$ feature tensor to a $H \times W \times |U|$ cost volume, $\mathbf{C}_u^A$. A similar network is trained to compute $\mathbf{C}_v^A$.

Motion regression Disparity regression has been used in stereo matching [128], where it is shown to be more robust than classification-based methods, and can generate sub-pixel accuracy. Furthermore, regression has been used to learn stereo cost volumes that are rich in geometry and contextual information [355, 128]. It is computed as the sum of each disparity, weighted by its probability, computed via a softmax over the cost volume.

We use a similar approach here to learn optical flow regression, $\mathbf{f}_0 = \{\hat{\mathbf{u}}, \hat{\mathbf{v}}\}$, as follows for each pixel i, j, prior to motion refinement:

$$\hat{u}(i,j) = U \cdot \sigma\left(C_u^A(i,j,:)\right), \tag{5.6}$$

$$\hat{v}(i,j) = V \cdot \sigma\left(C_v^A(i,j,:)\right). \tag{5.7}$$

Then, the initial flow prediction $\mathbf{f}_0$ and the learned cost volumes, C_u^A, C_v^A, are sent to the refinement module to compute a final motion prediction. Where motion refinement previously used correlation cost $C(i,j,u,v)$, it is instead fed concatenated, aggregated costs $[C_u^A(i,j,u), C_v^A(i,j,v)]$.

This motion regression learns a lower-resolution (*e.g.* 1/8, as used in RAFT [274]), but high-quality motion prediction that serves as a better input to the refinement module, considering that previous methods initialize with zero motion [268, 274]. As our ablation study shows (section 5.4.3), initializing motion with this regressed estimate is key to improving the prediction quality. It is worth noting that a standard (*i.e.* non-separated) 2D motion regression is naturally separable:

$$\mathbf{C}' = \sigma(C(i,j,:,:)), \tag{5.8}$$

$$\hat{u}(i,j) = \sum_u \sum_v U(u)\, C'(u,v), \tag{5.9}$$

$$= U \cdot \sum_v C'(:,v), \tag{5.10}$$

such that the $\sigma\left(C_u^A(i,j,:)\right)$ plays a similar role to $\sum_v C'(:,v)$. Given its efficacy in the stereo domain [128, 355], and the separable nature of 2D motion regression, this gives us some intuition into why motion regression can be used to effectively learn two separable 3D cost volumes that are also rich in prior contextual and geometry information.

5.3.3 Loss Function

Following RAFT [274], we use the L_1 loss between the predicted and ground truth flow for a sequence of N refinement predictions of optical flow, $\{\mathbf{f}_1, ..., \mathbf{f}_N\}$. However, in addition we also have the motion regressed flow, $\mathbf{f}_0$. Given ground truth flow $\mathbf{f}_{gt}$, our loss is thus defined as

$$\mathcal{L} = \sum_{i=0}^{N} \lambda^{N-i} \, ||\mathbf{f}_{gt} - \mathbf{f}_i||_1 \tag{5.11}$$

where $\lambda = 0.8$ in our experiments, weighting later refinement steps higher to ensure convergence.

5.4 Experiments

This section details the experiments and results that demonstrate our Separable Flow module is the new state of the art in accuracy for optical flow. It also demonstrates its improved cross-domain generalization, as well as the specific categories of error that our model fixes, with a discussion on why. An ablation study rounds off the evaluation.

Implementation Details: Our model is implemented in PyTorch [210] and we follow the training setting of RAFT [274]. Unless otherwise stated (*e.g.* sec. 5.4.3), we use the feature extraction and refinement modules from RAFT [274].

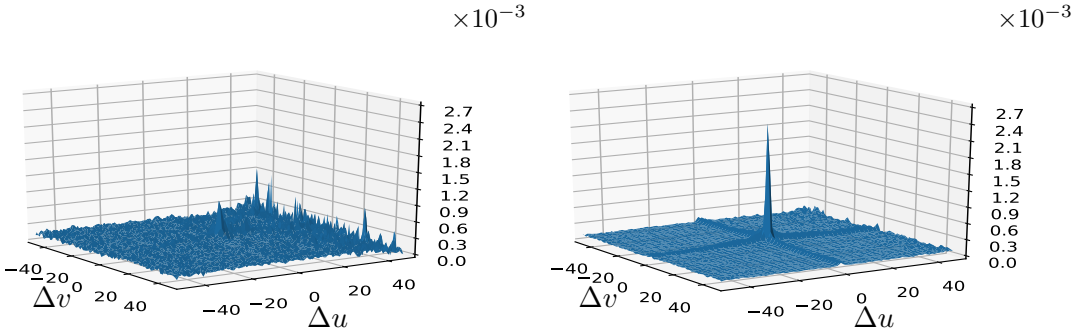
Following RAFT [274], we train our network on FlyingChairs [64] for 100k iterations (with batch size 12), then FlyingThings [187] for 100k iterations (batch size 6), and finally finetune on a combination of data from FlyingThings [187], Sintel [25], KITTI-2015 [191], and HD1K [136], for another 100k iterations (batch size 6). All other learning settings (including data augmentation) are the same as those in RAFT [274].

5.4.1 Quantitative Evaluation

We evaluate our Separable Flow model on the now standard, online, Sintel [25] and KITTI [191] benchmarks. We evaluate two models on each benchmark. The first is finetuned on the training set of the specific benchmark (*i.e*. Sintel or KITTI). The second is finetuned on the combined training set described above. Results are presented in the bottom two sections of Table 5.4 respectively. When compared with other methods trained on the same data, our method is leading in both the epe (end-point-error) and the Fl-all (threshold error rates) evaluations. On both benchmarks, best results for our method are achieved using the mixed training set. On Sintel, the average end point errors (EPE) of 1.50 (clean) and 2.67 (final) are both reductions of 7% over the previous best result, from RAFT [274]. On KITTI, the 4.64% error rate is a 9% reduction over the previous best result, also achieved by RAFT.

5.4.1.1 Cross-domain Generalization

Since collecting ground truth for real data is costly, generalization abilities are particularly important in real application scenarios. We test the cross-domain generalization performance of our model on Sintel (train) and KITTI (train) after training on synthetic FlyingChairs (C) and FlyingThings (T), with results shown in Table 5.4, second section. Our model again outperforms all existing published



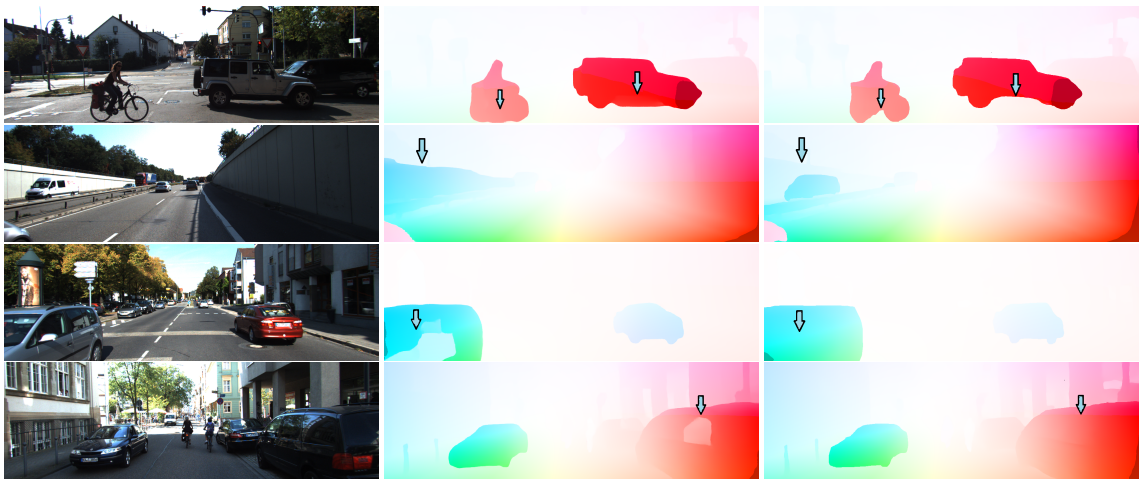
(a) RAFT (b) Ours

Figure 5.3: Comparisons of cost volumes. $(\Delta u, \Delta v)$ are the shifts from ground truth flow (origin of coordinates). The average normalized cost volumes over all pixels in occlusion regions across 100 image pairs are visualized. The center value (at the ground truth displacement) is 4.2× higher for our method vs. RAFT.

methods. Moreover, on real KITTI evaluations, our model achieves an error rate of 15.9%, which is far better than most existing models, and a 9% reduction over the previous best (once again, RAFT [274]).

In addition, we use extra synthetic driving scenes [26] to boost the generalization from synthetic scenes to a real driving dataset. By training only with these synthetic data (FlyingChairs, FlyingThings and Virtual KITTI2 [26]), our model achieves an error rate of 7.60% (Table 5.4, third section) on the real KITTI training set, and 7.92% on the KITTI test set. Several DNNs (*e.g.* PWC-Net [268], FlowNet2 [115] and LitleFlowNet [111]) perform worse than this, even when finetuned on the target KITTI training set.

We thus find that Separable Flow provides even greater performance gains when applied to cross-domain scenarios. We attribute these generalization abilities to our separable non-local aggregations, which capture more robust, non-local geometry and contextual information, instead of local, domain-sensitive features. *Visualized results and comparisons are shown in the supplementary materials.*



(a) Input views (b) RAFT [274] (c) Ours

Figure 5.4: Qualitative comparisons. (b) Results of the state-of-the-art RAFT [274]. (c) Results of our Separable Flow. Significant improvements are highlighted by arrows. The cost aggregations can effectively aggregate motion information to large, textureless regions (*e.g.* white wall behind the car), and reflective regions (*e.g.* car windows), give precise estimates. By learning contextual object information, it also preserves object boundaries very well (top row).

5.4.2 Qualitative Analysis

Separable Flow produces a clear quantitative improvement in accuracy. In this section we seek to explain qualitatively where these improvements arise, and why.

Fig. 5.3 visualizes the averaged & normalized Separable Flow cost volume (b) for a challenging occlusion regions and those of RAFT [274]. It can be seen that our cost volume offers a single, large peak at the ground truth motion, in contrast to the RAFT which has many noisy, false peaks in its cost volume. A similar effect can be seen in the reflection region (*available in the supplementary materials*). This demonstrates that our learned cost volume is able to overcome regional ambiguities, by exploiting global geometry and contextual information.

Fig. 5.4 compares optical flow outputs from our model with those of RAFT [274]. In challenging regions such as large textureless areas (*e.g.* the white wall behind the car), and reflection areas (*e.g.* the car windows), the matching information is

usually ambiguous, and thus leads to wrong matches in RAFT [274]. The non-local aggregations in our Separable Flow allow it to recognise and capture long-range contextual information, generating more accurate motion estimates in these regions. This rich contextual information also preserves object boundaries very well (top row).

5.4.3 Ablation Study

We perform a set of ablation experiments to validate the need for, and show the relative importance of, each of the components of the Separable Flow module that we propose. All ablation models are trained on FlyingChairs (C) + FlyingThings (T) and evaluated on the Sintel and KITTI training set.

Componentwise ablations: Results of componentwise ablations are shown in Table 5.1. In each section of the table, we test a specific component of our approach in isolation, with the settings used in our final model underlined.

Separation Channels: the attention layers of our self-adaptive cost separation provide a significant boost over just mean or max aggregation. *Aggregation layers* all improve performance, with SGA layers [355] providing the most benefit, highlighting the need for non-local aggregation. *Shared Agg. Weights*: cost aggregation networks for computing $\mathbf{C}_u^A$ and $\mathbf{C}_v^A$ can either share weights, or learn separate weights. The latter generates a reasonable advantage, due to the rotational variance of natural scenes. *Aggregation Blocks*: The hourglass blocks used in the [32, 355] are too resource heavy here. Instead, we tested using UNet and ResNet blocks, with the former providing better performance. *Motion Regression* substantially increases performance when used to initialize the motion refinement block, without increasing network bandwidth. This suggests it helps the network to learn *better*, rather than more.

These experiments validate the importance of each of the contributions of this

Experiment	Variations	Sintel (train)		KITTI-15 (train)		Parameters
		Clean	Final	Epe-all	Fl-all	
Baseline [274]	–	1.43	2.71	5.04	17.4	5.3M
Separation Channels	Mean	1.39	2.65	4.80	16.7	5.9M
	Max	1.38	2.65	4.74	16.5	5.9M
	Attention	1.32	2.62	4.72	16.2	6.0M
	All	1.30	2.59	4.60	15.9	6.0M
Aggregation Layers	2× 3D conv	1.39	2.68	4.91	16.8	5.9M
	8× 3D conv	1.33	2.63	4.75	16.4	6.2M
	2× SGA	1.34	2.64	4.71	16.2	6.0M
	4× SGA	1.30	2.59	4.60	15.9	6.0M
Shared Agg. Weights	No	1.30	2.59	4.60	15.9	6.0M
	Yes	1.34	2.65	4.72	16.3	5.7M
Aggregation Block	ResNet	1.33	2.63	4.74	16.1	6.0M
	UNet	1.30	2.59	4.60	15.9	6.0M
Motion Regression*	No	1.37	2.65	4.89	16.8	6.0M
	Yes	1.30	2.59	4.60	15.9	6.0M

Table 5.1: Ablation experiments. Settings used in our final model are underlined. See Sec. 5.4.3 for details.

work.

Different frameworks: Table 5.2 shows the performance gains using Separable Flow in different frameworks, and vice versa. We apply Separable Flow to two popular optical flow frameworks [268, 274], which differ in their refinement modules. Both frameworks are significantly improved, PWC-Net [268] even more so than RAFT [274], with reduction in errors ranging between 11-31% (compared to the latter's already reported 7%). Given our separated motion cost volumes, we are also able to use many different stereo matching backbones to process these volumes independently, and predict the motion directly. We test both PSMNet [32] and GANet [355]. Even without coarse-to-fine optical flow refinement modules, these models can still estimate motions more accurately than some popular optical flow models (*e.g.* PWC-Net [268]), demonstrating the flexibility of a separable motion cost volume representation.

Cost Agg. module	Refinement module	Sintel (train) final	Sintel (train) clean	KITTI (train) Fl-all (%)
–	PWC-Net [268]	2.55	3.93	33.7
Ours	PWC-Net [268]	1.89	3.51	23.1
–	RAFT [274]	1.43	2.71	17.4
Ours	RAFT [274]	1.30	2.59	15.9
Ours+PSMNet [32]	–	3.21	4.32	32.8
Ours+GANet [355]	–	2.49	3.81	28.1

Table 5.2: Performance using different refinement and aggregation modules. Models are trained on FlyingChairs and FlyingThings datasets and evaluated on Sintel and KITTI training sets.

5.4.4 Timing, Parameter and Accuracy

In Table 5.3, we compare the parameter counts, inference time, and training iterations for our method versus several recent cost-volume-based optical flow networks [332, 319, 274]. Separable Flow has a similar number of parameters and running speed as another cost-volume-based method [319], but achieves 24% lower error rates. Compared with state-of-the-art RAFT [274], our Separable Flow introduces about 0.7M new parameters, and is slightly slower. The main benefit of our method is therefore its improved accuracy.

Method	Param	Speed	Iterations	KITTI Fl-all (%)
FlowNet2 [115]	162.5M	0.1s	7100K	11.48
VCN [332]	6.2M	0.18s	300k	6.30
VCN+LCV [319]	6.3M	0.26s	–	6.25
RAFT [274]	5.3M	0.2s	350k	5.10
Ours	6.0M	0.25s	350k	4.64

Table 5.3: Comparisons of parameter counts, inference time, and training iterations *vs*. accuracy, of our model against recent cost-volume-based optical flow networks [319, 332, 274]. Speed measurements are from the KITTI2015 benchmark.

5.5 Conclusion

We have introduced the Separable Flow module, a cost-volume computation module for optical flow inference that is able to exploit non-local cost aggregation through the use of a separable cost volume representation, and motion regression. Our experimental results, which beat the previous state of the art in accuracy with a consistent 7% reduction in error, demonstrate that this module both resolves ambiguities in occluded, textureless and other such regions, through the use of non-local, contextual information and prior knowledge, and also improves cross-domain generalization when appyling a synthetically trained network to real data. Our ablation study validates the importance of each of the blocks that make up our Separable Flow module. We note that this module can benefit a broad class of optical flow methods, based on cost volumes.

Our model fails in just a few cases, where objects (*e.g.* cars) move in occluded regions.This is a common limitation of optical flow approaches: when a moving object is visible in only one image, networks predict the object to be stationary since this is the most plausible motion (for cars in KITTI, at least). To address this issue, multi-view or video inputs can be employed.

Acknowledgements This work was supported by Snap Inc., Turing AI Fellowship: EP/W002981/1, EPSRC/MURI grant EP/N019474/1. We would also like to thank the Royal Academy of Engineering and FiveAI.

Training Data	Method	Sintel (train)		KITTI-15 (train)		Sintel (test)		KITTI-15 (test)
		Clean	Final	Epe-all	Fl-all	Clean	Final	Fl-all
-	FlowFields [11]	-	-	-	-	3.75	5.81	15.31
-	FlowFields++ [246]	-	-	-	-	2.94	5.49	14.82
S	DCFlow [324]	-	-	-	-	3.54	5.12	14.86
S	MRFlow [316]	-	-	-	-	2.53	5.38	12.19
C + T	HD3 [349]	3.84	8.77	13.17	24.0	-	-	-
	PWC-Net [268]	2.55	3.93	10.35	33.7	-	-	-
	LiteFlowNet2 [112]	2.24	3.78	8.97	25.9	-	-	-
	VCN [332]	2.21	3.68	8.36	25.1	-	-	-
	MaskFlowNet [369]	2.25	3.61	-	23.1	-	-	-
	FlowNet2 [115]	2.02	3.54	10.08	30.0	3.96	6.02	-
	DICL-Flow [295]	1.94	3.77	8.70	23.6		-	-
	RAFT [274]	1.43	2.71	5.04	17.4	-	-	-
	Ours	**1.30**	**2.59**	**4.60**	**15.9**	-	-	-
C + T + V	RAFT [274]	1.45	2.75	3.20	9.13	-	-	-
	Ours	**1.32**	**2.61**	**2.60**	**7.74**	-	-	**7.92**
C+T+S/K	FlowNet2 [115]	(1.45)	(2.01)	(2.30)	(6.8)	4.16	5.74	11.48
	PWC-Net [268]	-	-	-	-	4.39	5.04	9.60
	LiteFlowNet [111]	(1.35)	(1.78)	(1.62)	(5.58)	4.54	5.38	9.38
	HD3 [349]	(1.87)	(1.17)	(1.31)	(4.1)	4.79	4.67	6.55
	ScopeFlow [15]	-	-	-	-	3.59	4.10	6.82
	DICL-Flow [295]	(1.11)	(1.60)	(1.02)	(3.60)	2.12	3.44	6.31
	VCN+LCV [319]	(1.62)	(2.22)	(1.13)	(3.80)	2.83	4.20	6.25
	RAFT+LCV [319]	(0.94)	(1.31)	(1.06)	(3.77)	2.75	3.55	6.26
	RAFT [274]	(0.77)	(1.20)	(0.64)	(1.5)	2.08	3.41	5.27
	Ours	(0.71)	(1.14)	(0.68)	(1.57)	**1.99**	**3.27**	**4.89**
C+T+S+K+H	PWC-Net+ [267]	(1.71)	(2.34)	(1.50)	(5.3)	3.45	4.60	7.72
	VCN [332]	(1.66)	(2.24)	(1.16)	(4.1)	2.81	4.40	6.30
	MaskFlowNet [369]	-	-	-	-	2.52	4.17	6.10
	RAFT (2-view)	(0.76)	(1.22)	(0.63)	(1.5)	1.94	3.18	5.10
	RAFT (warm-start)	(0.77)	(1.27)	-	-	1.61	2.86	-
	Ours	(0.69)	(1.10)	(0.69)	(1.60)	**1.50**	**2.67**	**4.64**

Table 5.4: Results on Sintel and KITTI datasets. Values in the brackets "$(*)$" are the results when evaluating on the training set (or subset). C+T: We test the generalization performance on KITTI (train) after training on FlyingChairs (C) and FlyingThings (T). C+T+V: We also provide extra synthetic driving scenes from Virtual KITTI (V) [26] to further boost the generalization on real driving scenes. Our method outperform existing methods for synthetic to real generalization. We also evaluate our model on public benchmarks after finetuning. C+T+S/K includes methods which finetune only on Sintel data when evaluating on Sintel, or only KITTI data when evaluating on KITTI. C+T+S+K+H includes methods that combine KITTI, HD1K, and Sintel data when finetuning. Separable Flow outperforms previous state-of-the-art approaches, ranking 1st among *all* published optical flow approaches on both Sintel (clean and final passes) and KITTI 2015 optical flow benchmarks.

Chapter 6

Looking Beyond Single Images for Contrastive Semantic Segmentation Learning

Feihu Zhang

University of Oxford

Philip H. S. Torr

University of Oxford

René Ranftl*

Intel Labs

Stephan R. Richter*

Intel Labs

* Equal advising

Abstract

We present an approach to contrastive representation learning for semantic segmentation. Our approach leverages the representational power of existing feature extractors to find corresponding regions across images. These cross-image correspondences are used as *auxiliary labels* to guide the pixel-level selection of positive and negative samples for more effective contrastive learning in semantic segmentation. We show that auxiliary labels can be generated from a variety of feature extractors, ranging from image classification networks that have been trained using unsupervised contrastive learning to segmentation models that have been trained on a small amount of labeled data. We additionally introduce a novel metric for rapidly judging the quality of a given auxiliary-labeling strategy, and empirically analyze various factors that influence the performance of contrastive learning for semantic segmentation. We demonstrate the effectiveness of our method both in the low-data as well as the high-data regime on various datasets. Our experiments show that contrastive learning with our auxiliary-labeling approach consistently boosts semantic segmentation accuracy when compared to standard ImageNet pre-training and outperforms existing approaches of contrastive and semi-supervised semantic segmentation.

6.1 Introduction

Training semantic segmentation models typically requires a large amount of densely annotated images. Producing such dense annotations for an adequate number of images at sufficient quality is an extremely laborious and costly task [58]. Self-supervised and weakly-supervised learning techniques offer the possibility to signi-

ficantly reduce the manual labeling effort that is required to train high-performance machine learning models. Contrastive learning [86, 41] has recently emerged as a promising technique for self-supervised representation learning that can leverage large amounts of unlabeled data for augmenting and extending existing labeled datasets without costly manual annotation. However, recent works on contrastive learning primarily focused on image/instance-level learning. Few works have studied contrastive learning in dense image labeling tasks, such as semantic segmentation [304, 301], as dense pixel-level segmentation poses unique challenges.

At its core, self-supervised contrastive learning is rooted in the availability of training samples that can be grouped automatically into *positive* pairs of similar concepts, and *negative* pairs of dissimilar concepts. The notion of similarity thereby depends strongly on the task at hand.

The appropriate selection of such pairs is of paramount importance for the performance of the resulting models. While it is often straightforward to generate negative pairs, automatically generating positive pairs can present a formidable challenge that requires intimate knowledge about the data and task at hand. Image augmentations are both simple and effective to generate positive pairs for image/instance-level learning [41, 42] but are too coarse-grained to learn representations at a pixel level.

Recent works [304, 322] adapt the idea of using image augmentations to generate positive pairs for more fine-grained pixel-level representation learning. While surprisingly effective, image augmentations have only a limited capability of covering the full visual spectrum of how two semantically related pixels or patches can look. Mid and high-level information such as context, texture, large visual shifts inside a semantic class (for example, the class "car" may comprise a wide range of different shapes and colors), and even projective distortions due to the imaging process, may be extremely relevant but cannot be simulated realistically

(a) Pseudo-labels for semi-supervised learning (b) Auxiliary labels for our contrastive learning

Figure 6.1: Pseudo-labels vs. auxiliary labels. (a) Semi-supervised semantic segmentation requires pseudo-labels to be compatible with the ground truth labels. (b) Our method employs auxiliary labels to find corresponding regions across images. These correspondences do not need to match the ground truth classes exactly as we leverage contrastive learning.

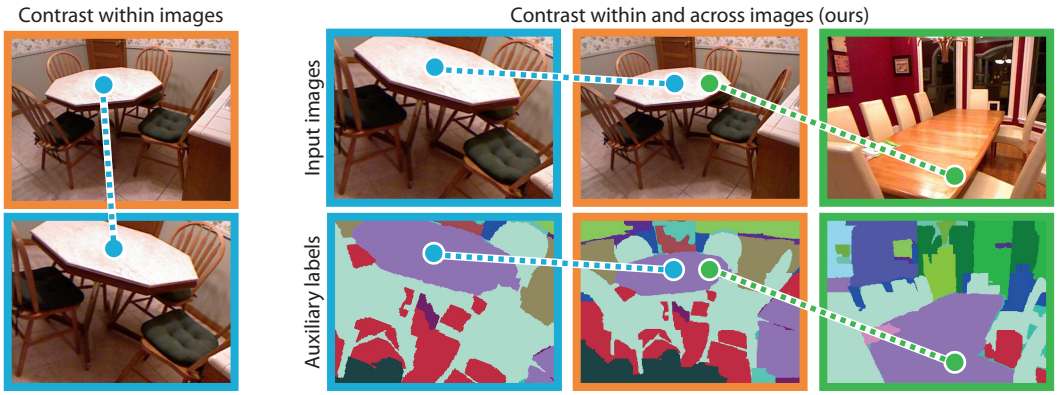


Figure 6.2: Comparison of augmentation-based contrastive learning (left) to our proposed cross-image contrastive learning with auxiliary labels (right). For an anchor image, prior work[304, 322] samples positive correspondences only within augmentations of the same image. Our method additionally establishes correspondences to other images based on matching auxiliary labels.

with augmentations alone. As a consequence, methods that exclusively rely on augmentations are limited in the contrastive information they can provide to downstream tasks. A different line of work thus exploits ground truth semantic labels to establish correspondences between images to generate more diverse and informative positive pairings [309, 301]. However, the need for labeled data impedes scaling these approaches to the massive amounts of data required to fully reap the benefits of contrastive learning [86, 41].

In this work, we explore the generation of *auxiliary labels* to establish fine-grained, pixel-level correspondences across images to select contrastive pairs. We distinguish

auxiliary labels from the more commonly employed pseudo-labels [385], as auxiliary labels need not represent the same classes as the ground truth labels. This concept is illustrated in Figure 6.1. To this end, we leverage feature embeddings that we generate with existing feature extractors. The feature extractors have either been trained using coarse image-level supervision, on a small set of pixel-level labeled data, or using image-level, self-supervised contrastive learning approaches[86, 44]. We cluster feature embeddings and assign auxiliary labels to individual pixels based on cluster membership. We also introduce a simple, yet effective, voting strategy to spatially regularize the auxiliary label maps. The auxiliary labels tend to cluster into semantically and perceptually meaningful groups which can be used to establish cross-image correspondences. This allows us to generate a diverse and realistic set of positive pairs that can be used for fine-grained contrastive visual representation learning. An example that illustrates our approach is shown in Figure 6.2.

We additionally introduce a proxy metric that allows us to quickly assess the quality of any auxiliary labeling without the need to train a model. Finally, we present a comprehensive experimental evaluation of various factors and hyper-parameters that influence the performance of contrastive training in semantic segmentation.

Our experiments across three datasets and two network architectures show that contrastive learning with auxiliary labels consistently boosts semantic segmentation performance when compared to standard ImageNet pre-training and outperforms existing approaches for contrastive and semi-supervised semantic segmentation.

6.2 Related Work

Approaches to self-supervised learning via a contrastive learning objective have recently demonstrated impressive performance compared to fully supervised methods in a wide variety of tasks [86, 44, 41, 42, 91]. In self-supervised learning, the contrastive objective is used as a pretext task to learn general visual representations,

which can be well transferred to downstream tasks via fine-tuning. The intuition of the contrastive objective is straightforward: similar visual concepts should be mapped to nearby representations in latent space, whereas dissimilar concepts should be mapped further apart. A crucial ingredient is the definition of similarity and hence the selection of similar and dissimilar samples during training. Seminal work adopted instance discrimination [86, 41]. That is, positive samples are generated from the same image via different transformations (views). All other images available for training represent potential negative samples. Recent work investigated different strategies to generate more informative positive and negative pairs, including mutual view selection[278, 277], hard mining[236, 126, 301], adversarial perturbations[132, 94, 100, 124], and using large mini-batches or a momentum memory bank [86, 194, 315].

Another line of work exploited ground truth labels to find positive and negative pairs for contrastive learning [130, 309, 301, 24, 298, 370]. While this approach may improve performance in fully supervised settings, it cannot be applied to unlabeled supplementary data.

Contrastive learning for segmentation. Recent works adapted contrastive learning to dense prediction tasks such as semantic segmentation [31, 304, 322, 383]. DenseCL applied the contrastive objective to the pixel level [304]. ReSim learned spatial features from sliding windows [318]. Xie *et al*. generated views by smoothing over nearby feature representations to aggregate local context [322]. Chen *et al*. exploited foreground/backward masks [35] while Van Gansbeke *et al*. leveraged object masks from pre-trained saliency detectors to group samples [289]. All of the above-mentioned approaches generate positive views from the same image. This limits the diversity of positive samples severely, as instances of the same category in different images cannot be paired. In contrast to existing works, our auxiliary-labeling strategy allows us to establish positive correspondences across

images and consequently to pair more diverse examples of the same concept.

Pseudo-labels in semi-supervised semantic segmentation. Pseudo-labels are a surrogate ground truth for unlabeled data. Many existing works[3, 380, 156, 36, 383, 380] train a teacher network on labeled data in order to produce surrogate annotations on the unlabeled data available. A student network is then trained on both the ground truth and the pseudo-labels. The surrogate annotations are usually far from perfect, forcing the student to learn from noisy (and sometimes just misleading) data, which may negatively affect its performance [271, 5]. An underlying problem in this setup is the need for compatibility between pseudo labels and ground truth labels. This forces a (possibly wrong) decision on the teacher network, while potentially more fine-grained knowledge present in the feature activations used to make the decision is completely ignored [197].

The auxiliary labels we construct from feature extractors exploit exactly this type of knowledge. By clustering feature representations, we obtain a more fine-grained set of labels that does not need to match the ground truth label set. Furthermore providing a more comprehensive perspective on the unlabeled data, auxiliary labels also allow us to leverage a wider range of feature extractors than just a (teacher) method built for the same task.

Unsupervised clustering for representations learning. Several works proposed unsupervised clustering objectives for deep representation learning, including approaches to image or instance-level representation learning [337, 16, 77, 105, 321, 328, 337, 381, 120] as well as image segmentation [133, 195, 120]. A line work clusters features and uses the cluster assignments as pseudo-labels for self-training [28, 29, 30, 8]. While we generate auxiliary labels also by clustering features, we leverage powerful existing feature extractors instead of a self-trained network. Furthermore, instead of learning image-level visual representations, we address the challenges of dense prediction tasks, such as semantic segmentation.

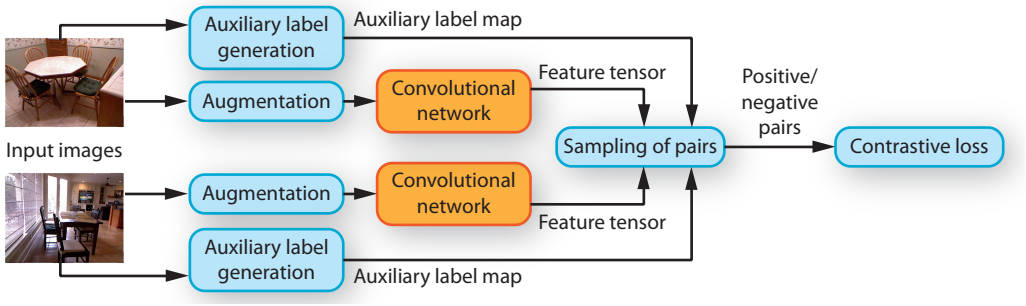


Figure 6.3: Method overview. Our method augments input images and feeds them to a convolutional network. We generate an auxiliary label map for each image and use it to guide the sampling of contrastive pairs. The network is trained using a confidence-weighted contrastive loss. Trainable components are shown in orange.

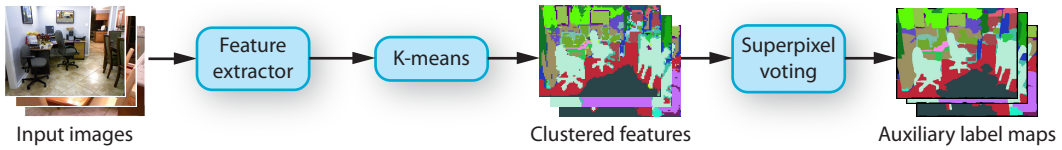


Figure 6.4: We generate auxiliary labels using existing feature extractors. We cluster features from the whole dataset with K-means. Pixels are assigned to clusters based on their distance to the cluster centers. A final smoothing step using superpixel voting aligns the auxiliary labels to image edges and removes outliers.

6.3 Method

The key to contrastive learning is the definition of positive and negative pairs, *i.e.* which samples should share a representation, and which samples should be mapped to different representations. For annotated datasets, the decision on how to construct positive and negative pairs is clear: samples with matching labels define the positives while samples with different labels define the negatives [130, 309, 301]. However, the need for annotations severely limits the data available for learning.

Missing any information about correspondences in the dataset, unsupervised approaches take a conservative stance and implicitly assume all randomly paired samples to mismatch. Matching samples are synthesized by sampling a single image and transforming it in different ways. The resulting views represent positive correspondences for training [86, 44, 41, 42, 91]. An obvious problem with this

approach is that correct pairs (*e.g.* the same car in different images) will be labeled as negative. At the same time, invariances are only learned w.r.t. the simple transformations applied.

To mitigate this issue, we generate auxiliary labels that allow establishing positive pairs across the whole training set akin to supervised methods. However, our auxiliary labels are generated using existing feature extractors, which alleviates the need for dense annotations and unlocks training on large-scale unlabeled data. Figure 6.3 provides an overview of our method.

Auxiliary label generation. Our approach for generating auxiliary labels consists of three steps: From a set of unlabeled images, we obtain visual features with an existing feature extractor. We cluster the features across the whole dataset and assign individual pixels in the input images to their corresponding cluster center. In a final step, we refine the resulting label maps through a super-pixel voting algorithm. A schematic overview of our approach is shown in Figure 6.4.

In principle, any feature extraction method can be used for generating auxiliary labels and we study several choices. First, we employ a ResNet-50 [89], trained through self-supervised contrastive learning [86]. It represents a feature extractor that has seen no manual annotations. Second, we use the same ResNet-50 architecture but trained for image classification on the ImageNet dataset to study the influence of coarse, image-level labels for training the feature extractor. Finally, we explore several instances of a DeepLabV3 [38] semantic segmentation network that has been trained for semantic segmentation on various datasets.

When using ResNet, we extract features before the final adaptive average pooling layer. This yields a representation at 1/32 of the input resolution. We extract multi-scale features by rescaling the input image to various resolutions and passing it to the feature extractor. We concatenate the feature maps from multiple scales, upsample the representation to 1/8 of the input image size through bilinear interpolation, and

normalize the resulting feature vectors using L_2 normalization. For DeepLabV3 feature extractors, we extract features before the penultimate layer (counting the final linear layer as the last layer) and follow the same multi-scale feature extraction procedure as before.

We apply our feature extraction approach to all images in an unlabeled dataset and generate a set of auxiliary classes using K-means clustering [212]. Since the total number of features in an unlabeled dataset can be extremely large (on the order of tens or hundreds of millions of feature vectors), we perform the clustering step on features from $5\,000$ images that we sample uniformly at random. We annotate the remaining images in the dataset by assigning pixels to the closest auxiliary class in terms of the L_2 distance between the pixel feature vector and the cluster center. To remove outliers and false auxiliary labels, we drop the 5% pixels with the largest distance to the nearest cluster center. An example result of our labeling strategy is shown in Figure 6.2.

The resolution and spatial consistency of auxiliary labels depends on the feature extractor employed. Some may produce only noisy or low-resolution feature maps. We thus refine the auxiliary label maps with a simple superpixel voting strategy. We compute a superpixel over-segmentation [2] and assign the most common label within a superpixel to all pixels of the superpixel. This aligns label map boundaries to image edges, removes outliers, and alleviates errors from low-resolution features.

Contrastive training. We use the auxiliary labels to generate positive and negative pairs both within an image as well as across images, and leverage them for pre-training semantic segmentation models. We train two standard segmentation architectures in our experiments [367, 38]. The final classification layer of both models is replaced with a linear layer that outputs a 128-dimensional feature map at $1/8$ of the input resolution. For each training iteration, we randomly sample $10\,000$ feature vectors as anchor vectors $\boldsymbol{v} \in \mathbb{R}^{128}$ and a different set of $40\,000$ feature

vectors as contrastive feature candidates ($\boldsymbol{v}^+ \in \mathbb{R}^{128}$ or $\boldsymbol{v}^- \in \mathbb{R}^{128}$). All anchors are selected from the current mini-batch, whereas we select $10\,000$ contrastive features from the current mini-batch and the remaining $30\,000$ contrastive features from a memory bank [86]. For every anchor $\boldsymbol{v}$, we construct a set of associated positive pairs $\mathcal{P}(\boldsymbol{v}) = \{(\boldsymbol{v}, \boldsymbol{v}^+)\}$ from all contrastive candidates where the auxiliary label of a candidate matches the anchor. We similarly create the set of negative pairs $\mathcal{N}(\boldsymbol{v}) = \{(\boldsymbol{v}, \boldsymbol{v}^-)\}$ from all contrast feature vectors where the auxiliary labels differ.

Our training loss is inspired by the supervised contrastive loss of Khosla *et al*. [130]. We define the contrastive distance for a single pair as

$$\mathcal{L}(\boldsymbol{v}, \boldsymbol{v}^+) \quad = \quad -\log \frac{\exp(\boldsymbol{v} \cdot \boldsymbol{v}^+/\tau)}{\exp(\boldsymbol{v} \cdot \boldsymbol{v}^+/\tau) + \sum_{(\boldsymbol{v},\boldsymbol{v}^-)\in\mathcal{N}(\boldsymbol{v})} \exp(\boldsymbol{v} \cdot \boldsymbol{v}^-/\tau)}, \tag{6.1}$$

where we set the temperature $\tau = 0.07$ in our experiments. To compute the loss over all pairs, we introduce a confidence weight that is based on the expected reliability of a positive pair. To this end, we partition the set of positive pairs into three subsets: the set $\mathcal{P}_0$ consists of pairs for which both the anchor and the contrast sample belong to the same superpixel. We assign particularly high confidence to these pairs due to the inherent high spatial regularity of superpixels and feature extractors. We assign medium weight to the set $\mathcal{P}_1$ that consists of pairs sampled from the same image. Finally, the set $\mathcal{P}_2$ consists of samples where the anchor and contrastive features are from different images. Compared with samples in $\mathcal{P}_0$ and $\mathcal{P}_1$, these cross-image samples are more likely to be wrongly labeled as positive pairs. We thus assign the lowest weight to these samples. Our final loss is given by

$$\mathcal{L} \quad = \quad \frac{\sum_{i=0}^{2} \lambda_i \sum_{(\boldsymbol{v},\boldsymbol{v}^+)\in\mathcal{P}_i} \mathcal{L}(\boldsymbol{v}, \boldsymbol{v}^+)}{\sum_{i=0}^{2} \lambda_i |\mathcal{P}_i|}, \tag{6.2}$$

where $\lambda_{0,1,2} = \{10, 4, 1\}$ for all our experiments.

Auxiliary label quality. Our label generation strategy supports various technical

choices that affect the efficacy of the auxiliary label maps. For example, different feature extractors or a different number of auxiliary classes in the clustering step will produce auxiliary labelings that will lead to different performance for a given downstream task. We introduce a novel metric that can be quickly computed and can serve as a guiding metric to choose a specific auxiliary-labeling strategy when a representative set of labeled images is available.

Let $S = \{S_j\}_{j=1}^{K}$ be the set of auxiliary segments, where S_j is defined as the set of pixels with the auxiliary label j (computed over the complete validation set). Similarly, let $G = \{G_i\}_{i=1}^{M}$ be the set of true segments, where G_i is the set of pixels that correspond to the ground truth class i. We define the *Proxy mean Intersection-over-Union (PmIoU)* as

$$\mathrm{PmIoU}(S, G) = \frac{1}{\max\{K, M\}} \sum_{S_j} \max_{G_i} \left\{ \frac{|S_j \cap G_i|}{|S_j \cup G_i|} \right\}. \tag{6.3}$$

The metric assigns each auxiliary segment S_j to the ground truth segment G_i that yields the highest Intersection-over-Union (IoU) among all possible assignments, as we do not know the correct correspondence between auxiliary labels and ground truth labels (additionally, in most cases a one-to-one correspondence will not exist). Intuitively, PmIoU reflects the highest mIoU that is achievable by optimally matching the auxiliary labels with the ground truth classes. In the ideal case, where the auxiliary labels exactly coincide with the ground truth labels, the PmIoU equals one. On the other hand, if the auxiliary labels represent a strong over-segmentation or are crossing spatial boundaries between different ground truth segments, the PmIoU will be small.

Figure 6.5 shows PmIoU together with the mIoU after fine-tuning a model that was pre-trained with the corresponding auxiliary labels. We observe that PmIoU correlates well with mIoU across auxiliary labeling strategies. This indicates that

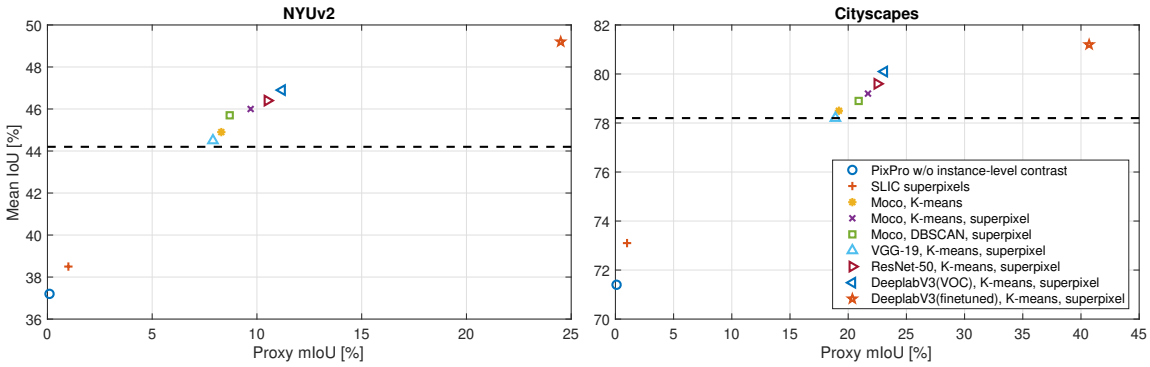


Figure 6.5: Auxiliary label quality and final segmentation performance for conditions in our controlled experiments. The proposed proxy mIoU correlates strongly with the mIoU of segmentations after fine-tuning. Dashed lines mark the performance of a DeeplabV3 baseline with ImageNet pre-training.

PmIoU is an excellent and, most importantly, computationally cheap metric to evaluate different auxiliary-labeling strategies.

Practical considerations. In practice, we find that a large amount of positive pairs improves convergence speed, while a sufficiently large amount of negative pairs are required to train models that perform well on the downstream task. We thus implement a memory-efficient version of our loss that doesn't explicitly store the inner products between all pairs. This allows us to scale the total number of pairs per mini-batch substantially. With $10\,000$ anchor feature vectors and $40\,000$ contrast feature vectors, we can, depending on the composition of the mini-batch and auxiliary labels, see on average 10 million positive pairs in a single mini-batch. We find that with this very large sample size, 20 epochs suffice to train strong representations for segmentation models. For comparison, MoCo [86] uses only 256 positive pairs per mini-batch and consequently requires more than 200 training epochs.

Supervised semantic segmentation models commonly employ an auxiliary loss on mid-level features to improve performance [367, 38]. We find that an auxiliary contrastive loss on mid-level features can likewise improve the representational power of our contrastively trained models. State-of-the-art semantic segmentation

models output features at a lower resolution than the input image (usually 1/8 of the input size). We train with the contrastive loss both at the low output resolution that is defined by the model, as well as at the original resolution of the input image (by upsampling the feature maps). We give equal weight to both losses during training.

In addition to large mini-batches, momentum contrast [86] can provide candidate pairs with high diversity and is commonly employed for effective contrastive learning. We also adopt this technique.

6.4 Evaluation

We evaluate our approach on multiple challenging datasets: NYUv2 [254] contains about $500\,000$ unlabeled and 1449 labeled images of indoor scenes. The labeled set is annotated with 40 classes and split into 795 train and 654 test images. Cityscapes [58] contains more than $100\,000$ unlabeled video frames of driving scenes, 2975 labeled images for training and another 500 annotated images for validation. ADE20K[376] consists of $20\,000$ training and 2000 validation images, all of which are fully annotated. Due to the lack of unlabeled images, we supplement ADE20K with MS-COCO [166] (without using MS-COCO's annotations).

For generating auxiliary labels from multi-scale feature maps, we resize input images to the ResNet feature extractor by factors 1, 2, 4 for NYUv2 & ADE20K, and by 0.5, 1, 2 for Cityscapes. With a DeepLabV3 feature extractor, we scale input images for all datasets by factors 0.5, 0.75, 1, 1.5, 2. We generate $K = 1.5M$ auxiliary labels, where M is the number of groundtruth classes in the target dataset. We used our proposed PmIoU proxy metric to determine this number.

Our approach can be used to train any semantic segmentation architectures. We

choose two popular architectures, DeepLabV3[38] and PSPNet[367]*, both with a ResNet-50 backbone for our experiments. All experiments were conducted on 8 Nvidia Quadro 6000 GPUs. We use batches of 128 input images and set the crop size to 321×361. The memory bank for implementing momentum contrast holds the features of 384 frames. All models are trained using SGD with momentum[239]. We set weight decay to $1e{-}4$ and the momentum term to 0.9. We perform warmup with a learning rate of 0.01 for two epochs and then train for another 18 epochs with a learning rate of 0.1. The learning rate is reduced by a factor of 10 after epochs 10, 15, and 18, respectively. We use the augmentations defined in SimCLR[41], which include random scaling, rotation, cropping, and color transformations, together with CutOut[62], where we fill the cut-out region with the mean value of the cut-out. A single training run on NYUv2 takes approximately 50 hours.

For fine-tuning, we use the same hyper-parameters as defined in the original works [367, 38].

*Results for PSPNet are available in supplementary materials.

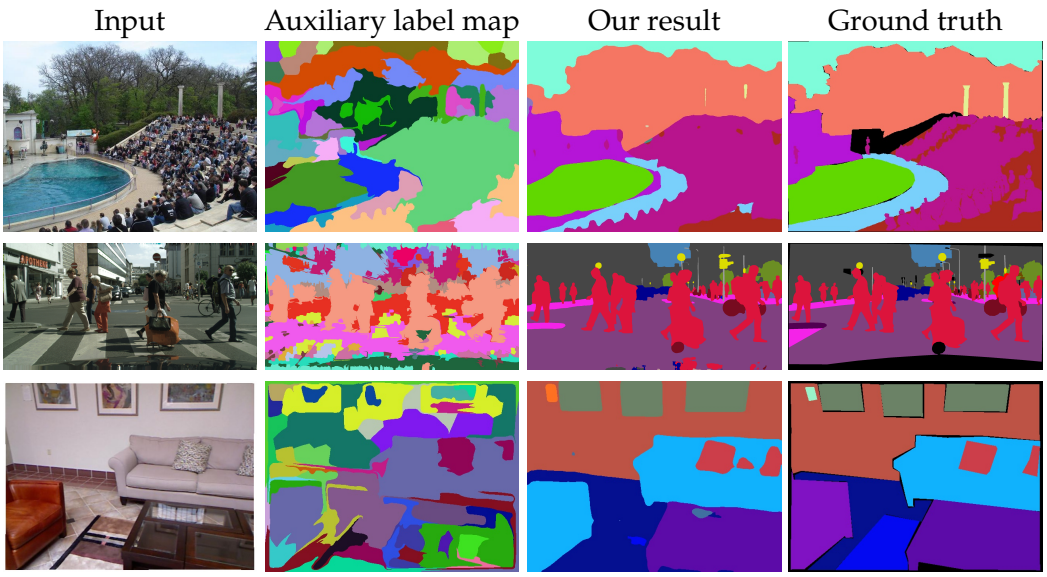


Figure 6.6: Comparison of auxiliary labels (unsupervised ResNet-50 (MoCo) as the feature extractor) and results after fine-tuning for samples from ADE20K, Cityscapes, and NYUv2 (top to bottom).

	Method	NYUv2 10%	NYUv2 20%	NYUv2 100%	Cityscapes 10%	Cityscapes 20%	Cityscapes 100%	ADE20K 10%	ADE20K 20%	ADE20K 100%
	Random initialization	-	-	25.3	-	-	68.2	-	-	35.1
	ImageNet	20.4	30.3	44.2	42.3	57.2	78.2	20.2	26.7	43.0
Baselines	SWAV [30]	-	-	41.2	-	-	77.3	-	-	43.1
Baselines	PCL [154]	-	-	43.9	-	-	77.6	-	-	42.9
Baselines	MoCo-v2 [44]	19.1	29.5	43.8	41.5	56.3	77.9	19.7	26.3	42.8
Baselines	DenseCL (NYUv2) [304]	17.2	26.1	39.8	-	-	-	-	-	-
Baselines	DenseCL (ImageNet) [304]	-	-	44.8	-	-	78.6	-	-	43.4
Baselines	PixPro [322]	22.1	31.2	45.0	43.5	58.1	78.4	21.4	27.2	43.2
Baselines	PseudoSeg [385]	30.8	35.9	47.1	48.1	59.7	79.9	-	-	43.6
Ours	ResNet-50 (MoCo)	29.8	35.1	46.0	48.2	60.1	79.2	24.7	29.1	43.7
Ours	ResNet-50 (ImageNet)	30.4	35.7	46.4	48.4	60.3	79.6	25.2	29.7	43.9
Ours	DeepLabV3 (Pascal VOC)	31.7	36.3	46.9	49.1	60.0	80.1	25.7	30.0	44.0
Ours	DeepLabV3 (VIPER)	-	-	-	48.1	60.2	79.8	-	-	-
Ours	DeepLabV3 (*)	36.1	39.4	49.2	52.1	61.9	81.2	27.1	31.2	44.5

Table 6.1: Comparison to prior work. We report mIoU after fine-tuning on various fractions (10%, 20%, 100%) of the target dataset. We compare to several contrastive learning methods[304, 44, 322] and to a state-of-the-art semi-supervised semantic segmentation model[385]. To generate auxiliary labels we employ feature extractors trained for image classification using self-supervised learning (MoCo) and labeled data (ImageNet), or semantic segmentation networks that were trained on another dataset (Pascal VOC [69]), on synthetic data (VIPER [233]), or on the target dataset (*).

Comparison to prior work. We first compare to prior work on unsupervised and semi-supervised representation learning. We use a DeepLabV3 network that has been trained on the target dataset as feature extractor and report results for different fractions of labeled data from the target dataset used for training. We compare to different state-of-the-art contrastive learning methods, including image-level contrastive learning [86] and pixel-to-pixel dense contrastive learning methods [322, 304]. Our results with different feature extractors are shown in Table 6.1. Existing contrastive learning methods (sometimes implicitly) assume that only a single object is present in a training image. We thus find that they struggle to

Method	Pixel-level contrast	Cross-image contrast	Epochs	mIoU
MoCo-v1[86]			80	42.1
MoCo-v1[86]			200	47.0
Hierarchical Grouping [362]	✓		80	46.5
Ours (MoCo)	✓	✓	20	49.1
Ours (*)	✓	✓	20	**54.3**

Table 6.2: Comparison to prior work [362] on the Pascal VOC validation set. All methods were pre-trained on a mix of the Pascal VOC [69] and MS-COCO [166] training sets (without ImageNet pre-training). Our approach outperforms the baselines both with an unsupervised feature extractor (MoCo) and with a semi-supervised feature extractor (*) with only a quarter of training epochs.

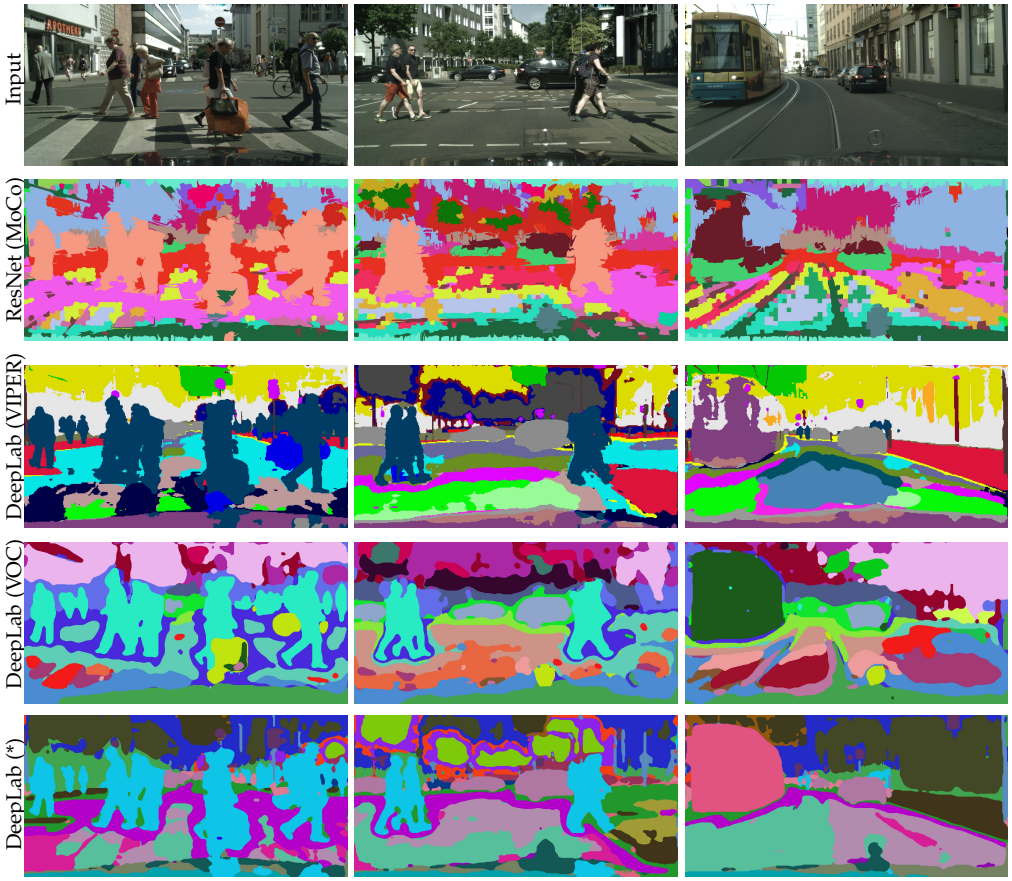


Figure 6.7: Qualitative comparison of auxiliary labels from different feature extractors. Corresponding colors indicate corresponding auxiliary labels across images.

learn competitive representations when trained on the complex images that are typically found in semantic segmentation datasets. For example, when training DenseContrast [304] on the NYUv2 dataset, we observe that the results trail ImageNet

pre-training by more than 4%. We see much stronger performance of this approach when training on ImageNet data, where the results even slightly surpass training with labeled data. We consequently train the remaining contrastive baselines, MoCo and PixPro, only on ImageNet as they are subject to the same considerations.

Our method is specifically designed for semantic segmentation. It can thus be directly trained on complex semantic scenes and can consequently reap the benefits of seeing this more complex data. It consistently outperforms ImageNet pre-training and existing contrastive approaches across datasets, independent of the amount of total labeled samples that are given. We additionally compare our approach to PseudoSeg [385], a state-of-the-art semi-supervised segmentation model that leverages pseudo-labels and has been jointly trained on the same labeled and unlabeled data as our approach. Using all available labeled data, our method outperforms PseudoSeg by 2.1% mIoU on NYUv2, by 1.3% mIoU on Cityscapes, and by 0.9% mIoU on ADE20K, respectively. When only a fraction of labeled data is available, our approach provides even further gains over the state of the art. Figure 6.6 shows example auxiliary labels together with our results after fine-tuning on these datasets.

We additionally compare our approach to a closely related work that leverages hierarchical grouping to sample pixel-level positive pairs within a single image [362] in Table 6.2. We follow the protocol of [362] and use a ResNet-50 backbone. All approaches (and feature extractors used in our method) were pre-trained on a mix of the Pascal VOC [69] and MS-COCO [166] training sets before fine-tuning on the Pascal VOC training set. Cross-image positive samples strongly improve performance even after only training for a quarter of epochs. The performance of [362] is comparable to MoCo-v1[86], whereas our method outperforms the more advanced MoCo-v2[44] baseline on other datasets.

Feature extractors. We assess how different feature extractors affect the segmentation

performance after fine-tuning. To that end,We evaluate two ResNet-50 feature extractors, one of which was trained using unsupervised image-level contrastive learning on ImageNet data (without annotations), and one of which was trained on ImageNet labels. We further evaluate a DeepLabV3 feature extractor that has either been trained on Pascal VOC [69], the synthetic VIPER dataset [233], or on the target dataset itself. These evaluatedfeature extractors represent a spectrum of different labeling efforts that are required to construct them, ranging from no effort (MoCo) to potentially high effort when labeled data on the target dataset (*) is required. Table 6.1 (bottom) summarizes our results. While all variants yield strong results, we see that feature extractors that were provided with more informative labels and data that corresponds closely to the target task yield the best performance. Surprisingly, even the MoCo feature extractor, which was trained without any labeled data, yields results competitive with feature extractors that have been trained on ImageNet or even segmentation datasets (Pascal VOC and Synthetic VIPER). We observe the best performance when training the feature extractor on the target dataset. Note that the results are consistent even when the amount of labeled data is reduced. Figure 6.7 shows example auxiliary labelings with different feature extractors.

Controlled experiments. We study the influence of different technical choices for generating auxiliary labels and report results in Figure 6.5 (corresponding numerical results are provided in supplementary materials). We find that superpixel refinement improves the auxiliary label quality and consequently leads to an improvement of the mIoU by 0.7-1.1%. We evaluate DBSCAN [68] as an alternative clustering algorithm but find that it performs slightly worse than K-means clustering while having higher memory and computational requirements.

We finally provide a comprehensive evaluation of the influence of various hyper-parameters. We show results on the NYUv2 dataset for these experiments and use a ResNet-50 trained with MoCo[44] on the unlabeled portion of the dataset to generate

auxiliary labels. Evaluations are conducted on the validation set. We study the effects of (1) the number of positive pairs and negative pairs in a training iteration, (2) batch size and the momentum-based memory bank[86], (3) multi-scale contrastive training, (4) the influence of the auxiliary contrastive loss, and (5) the influence of the unlabeled training set size and the number of epochs. Table 6.3 provides an overview of our results. We find that selecting more positive and negative pairs (top section) in a training step improves mIoU and reduces the required number of epochs to get well-performing representations. CutOut augmentation has a positive influence and using more unlabeled data and training for more epochs further improves results (middle section). Increasing the crop size reduces performance, as it necessitates a reduction of the batch size and thus reduces the diversity of input samples due to memory constraints. The last section shows that multi-scale training slightly improves accuracy by 0.2% mIoU, while using an auxiliary contrastive loss on the mid-level features contributes another 0.4% improvement. Finally, using the proposed confidence weighting of the loss function improves accuracy by another 0.4%.

6.5 Discussion

In this paper, we study the use of auxiliary labels as guidance to build cross-image correspondences for contrastive learning in semantic segmentation. Our method leverages existing feature extractors to generate auxiliary labels. The requirements on the feature extractors are loose, as we demonstrate by using a broad set, ranging from networks trained for image classification to semantic segmentation approaches trained on synthetic data. For all feature extractors, our method consistently outperforms pre-training on ImageNet. Our exploration was guided by a new quality metric for auxiliary labels (Proxy mIoU), which can provide an estimate for

+&- selection		Hyperparameters						Settings of contrastive loss			mIoU
Anchors	Contrast	CutOut	Batch size	Crop size	MoCo	Data	Epochs	Multi-scale	Aux. loss	Confidence	
100	400	-	32	321×361	-	20k	10	-	-	-	39.2
1k	4k	-	32	321×361	-	20k	10	-	-	-	42.1
10k	40k	-	32	321×361	-	20k	10	-	-	-	42.7
10k	40k		128	321×361	-	20k	10	-	-	-	43.2
10k	40k	√	128	321×361	-	20k	10	-	-	-	43.5
10k	40k	√	64	473×473	-	20k	10	-	-	-	43.3
10k	40k	√	128	321×361	384	20k	10	-	-	-	43.7
10k	40k	√	128	321×361	384	20k	20	-	-	-	43.9
10k	40k	√	128	321×361	384	128k	20	-	-	-	45.0
10k	40k	√	128	321×361	384	128k	20	√	-	-	45.2
10k	40k	√	128	321×361	384	128k	20	√	√		45.6
10k	40k	√	128	321×361	384	128k	20	√	√	√	46.0
10k	40k	√	128	321×361	384	480k	40	√	√	√	47.1

Table 6.3: Controlled experiments. "Anchors" and "Contrast" are the numbers of pixels that are selected as anchor features and contrast candidate features per iteration. "Multi-scale" denotes evaluating the contrastive loss on both 1/8 resolution and full resolution of the input image. "Aux. loss" adds a contrastive loss to an auxiliary middle layer, similar to the auxiliary loss used for fine-tuning DeeplabV3. "MoCo" denotes wether we use momentum contrast training [86].

the final segmentation performance after fine-tuning without the need for training a network. Finally, we propose several strategies for effective contrastive learning in semantic segmentation. We believe that our approach opens up new ways of integrating unlabeled training data for semantic segmentation models.

Limitations. Our method is demanding in its hardware requirements and its performance scales with available GPU memory. It shares this limitation with other state-of-the-art methods for contrastive learning [41, 86]. However, the computational cost of exploring new auxiliary-labeling strategies can be reduced substantially by applying our proposed proxy quality metric.

Potential risks. Every method that learns from data carries the risk of introducing biases. Additionally, the evaluation on specific benchmarks may suffer from dataset or concept bias. Work that bases itself on our method should carefully consider the consequences of any potential underlying biases.

Chapter 7

Unsupervised Contrastive Domain Adaptation for Semantic Segmentation

Feihu Zhang

University of Oxford

Vladlen Koltun

Intel Labs

Philip H. S. Torr

University of Oxford

René Ranftl*

Intel Labs

Stephan R. Richter*

Intel Labs

* Equal advising

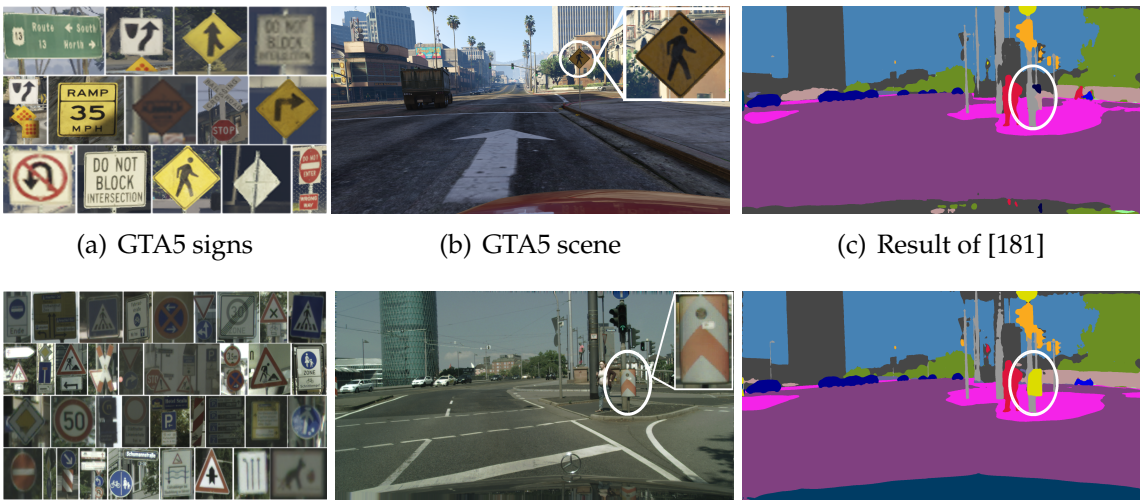
Abstract

Semantic segmentation models struggle to generalize in the presence of domain shift. In this paper, we introduce contrastive learning for feature alignment in cross-domain adaptation. We assemble both in-domain contrastive pairs and cross-domain contrastive pairs to learn discriminative features that align across domains. Based on the resulting well-aligned feature representations we introduce a label expansion approach that is able to discover samples from hard classes during the adaptation process to further boost performance. The proposed approach consistently outperforms state-of-the-art methods for domain adaptation. It achieves 60.2% mIoU on the Cityscapes dataset when training on the synthetic GTA5 dataset together with unlabeled Cityscapes images.

7.1 Introduction

Semantic segmentation is the task of dense per-pixel classification of the content of an input image and is one of the fundamental problems of computer vision. Deep networks have achieved tremendous advances towards solving this problem when a large amount of labeled data is available. In the supervised training paradigm it is crucial that the labeled training data is sufficiently similar to the data that will be encountered in the target application. When this fundamental assumption is violated, heavy performance drops are frequently observed, rendering such models impractical for deployment in real-world applications. Unsupervised domain adaptation aims at mitigating these performance drops by augmenting existing labeled training data from a source domain with unlabeled images from the target domain.

The underlying issues resulting in a performance loss when switching domains



(a) GTA5 signs (b) GTA5 scene (c) Result of [181]

(d) Cityscapes signs (e) Cityscapes scene (f) Our result

Figure 7.1: Problem illustration and example result. (a) Traffic signs in the GTA5[234] dataset. (d) Traffic signs in the Cityscapes dataset [58]. Strong domain shift within a semantic class can lead to poor performance when transferring a model. (b) and (e) Domain shift can also include low-level style and overall scene layout. (c) Results of a state-of-the-art domain adaptation method[181]. (f) Result of our approach. Our method learned to recognize a traffic sign that didn't appear in source dataset.

are loosely summarized by the term *domain shift*. It includes apparent low-level difference between images that might be due to different camera setups or weather conditions. Several existing works apply style transfer networks with adversarial training [313, 308, 160, 108] to align the low-level image statistics of the source and target domain in order to improve transfer. A second, more subtle, source of domain shift is the distribution of semantic content in the images. This issue includes, but is not limited to, a mismatch between the actual objects depicted in the source and the target domain. While it is apparent that a model cannot learn to segment an object if it was never encountered in the labeled source data, more subtle issues can lead to poor performance in the target domain. An example is Figs. 7.1 (a) and (b), where we show various traffic signs in two different domains. It is apparent that there is a strong class-specific domain shift that can not be mitigated by low-level style transfer alone.

An underlying assumption of many domain adaptation approaches is that training on different datasets (domains) will lead to different visual representations, even if the set of classification labels is the same. Thus, many works explicitly target the alignment of feature representations between domains. What makes this particularly challenging is the lack of labeled data for the target domain, and hence no directly available supervision for learning a representation in the target domain to begin with.

In this work, we introduce contrastive learning for unsupervised adaptation of semantic segmentation models across domains. We build both in-domain contrastive pairs (in the source and the target domain) as well as cross-domain contrastive pairs (from source to target domain) to improve feature alignment between domains while simultaneously ensuring a highly discriminative representation. We propose a student-teacher approach that iteratively updates a set of pseudo labels in the target domain, which allows us to draw corresponding samples from both domains. Contrastive learning allows us to naturally balance frequent easy classes and infrequent hard classes by controlling the sampling of contrastive pairs. We further propose a feature-based pseudo-label expansion strategy to discover more pixels that correspond to hard classes.

Our approach consistently outperforms the state of the art for domain adaptation across multiple datasets. It achieves 60.2% mIoU on the Cityscapes validation set when trained on the synthetic GTA5 dataset [234] together with unlabeled Cityscapes training images. We observe a state-of-the-art 56.5% mIoU when leveraging the synthetic SYNTHIA dataset [238] in the same setting. Our experiments show that our approach significantly improves performance particularly on hard classes that have few labeled pixels in the source data or that potentially undergo a strong domain shift between source and target domain.

7.2 Related Work

Adversarial training. In an adversarial setup, a discriminator is trained to distinguish images [50, 55, 95, 108, 379], intermediate representations [48, 50, 308, 294], or predicted labels [284, 285] from different domains. The discriminator then provides a supervisory signal for aligning the distributions of its inputs and thus the different domains. For more fine-grained supervision, one line of work trained the discriminator to additionally distinguish individual classes [48, 294]. To leverage local spatial invariances, ROAD aligns fixed image regions [49], and Tsai *et al*.'s work predicts patches mined from the source domain [285]. SIM aligns the feature representation from stuff and things of the target domain to specific samples in the source domain [308]. ADVENT finds that the entropy of predictions is higher on the target domain and introduces an entropy minimization loss [291]. This can lead to exploding gradients for confident predictions, which can be addressed via a maximum square loss [39]. CrDoCo trains separate networks in the task domains and leverages a cross-domain consistency [50].

Kim *et al*. [198] apply random styles using style transfer to learn a texture-invariant representation [131]. Yang *et al*. propose to reconstruct images from inferred label maps [334]. FDA transfers low-frequency image components [344] and PCEDA introduces a phase-consistency loss to retain semantic content [341]. DISE allows for domain-specific, non-transferable features [33] and CSCL learns non-transferable content via reinforcement learning [63]. Zhang *et al*. propose several domain-invariant regularizers to improve the consistency of predictions [364]. Yang *et al*. introduce an attention mechanism to learn transferable contextual relations across domains [335].

Self-training. CBST introduced self-training for domain adaptation [270] to semantic segmentation [384]. State-of-the-art self-training relies on pseudo labels for the

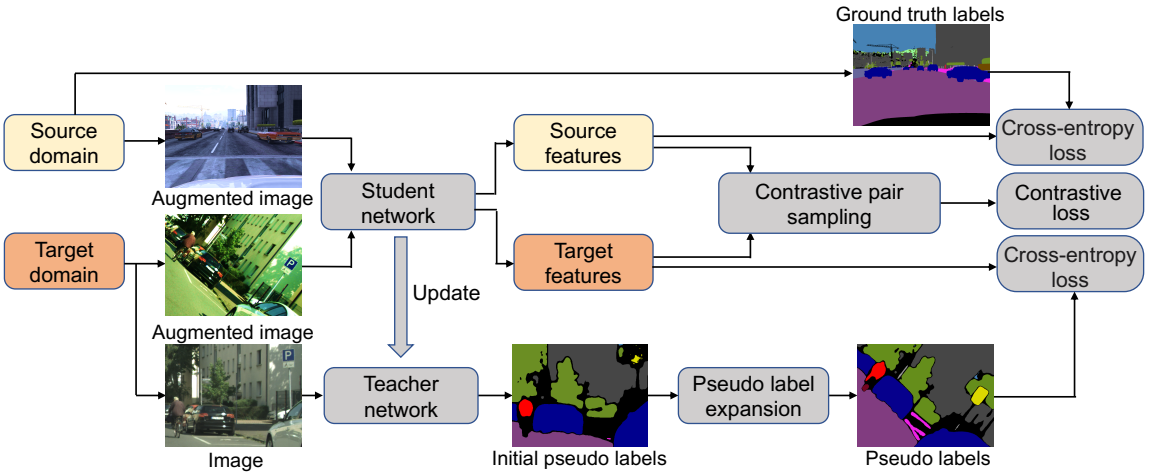


Figure 7.2: Overview of self-training stage. We train a semantic segmentation network (student) with augmented images from source and target domains. It is supervised via ground truth labels from the source domain and pseudo labels generated by a teacher network from unmodified target domain images. We improve the pseudo labels via a novel expansion mechanism. Further details in the text.

target domain, which are generated by earlier versions (a *teacher*) of the network (the *student*) being trained. To balance the selection of samples to harder cases in pseudo labels, adaptive thresholds are commonly used for classes [4, 265] or even instances [189]. The predicted label distributions are frequently sharpened during training [18]. Subhani and Ali propose to leverage the expected consistency of predictions across scales to generate pseudo labels [265]. SAC extends this idea to a broader range of image transformations [4], and TGCF-DA as well as BiSIDA to different stylizations [55, 296]. Another line of work leverages spatially related samples [127, 180]. IAST smooths pseudo labels in high-confidence regions and sharpens them for low-confidence regions [189]. Zheng and Yang exploit auxiliary classifiers to obtain a confidence measure for the pseudo labels [373]. ProDA weights pseudo labels by their distance to prototypical features to reduce the influence of outliers [359]. Ma *et al*. propose a triplet loss to align average feature representations for different categories [181]. While we also aim to align feature representations, we formulate the alignment as a more powerful contrastive objective.

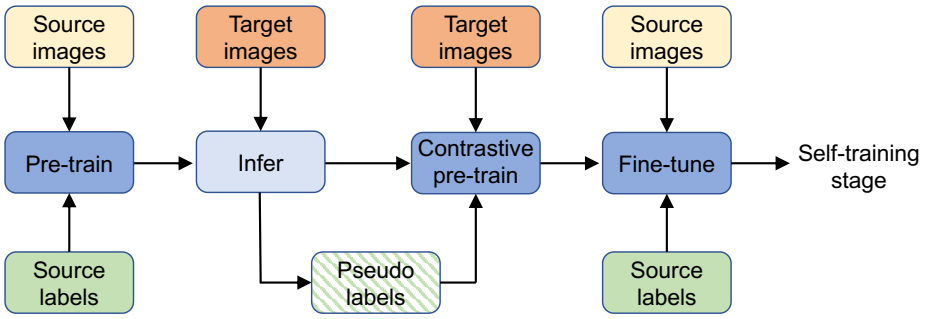


Figure 7.3: Pre-training stage. We start by training on the source domain with direct supervision and obtain pseudo labels for the target domain to guide contrastive training. Finally, we fine-tune again on the source domain before progressing to self-training.

Contrastive training. After impressive performance demonstrations for image classification [86, 44, 41, 42], contrastive representation learning has found its way to semantic segmentation as an alternative to the ubiquitous pre-training on ImageNet [362, 370] as well as a supporting training objective [309]. A number of recent works have adapted the contrastive learning framework to semantic segmentation, but restricted the sampling of positive correspondences to the same image [31, 34, 304, 322, 318, 383, 289].

Wang *et al*. exploited ground truth labels to construct contrastive pairs for fully supervised semantic segmentation [301]. SSC retains high-quality features to build contrastive pairs in a semi-supervised setting [3]. For domain adaptation, Liu *et al*. [173] identify patches to pair via simplified spatial pyramid matching. ASSUDA uses a contrastive loss to minimize the distance of predictions to adversarial samples [336]. Masden *et al*. construct contrastive pairs from mean feature representations in the source and target domain to features averaged over target images [184]. Our work employs three complementary strategies for sampling contrastive pairs, which exploit supervision in the source domain, compress feature representations, and align them across domains.

7.3 Overview

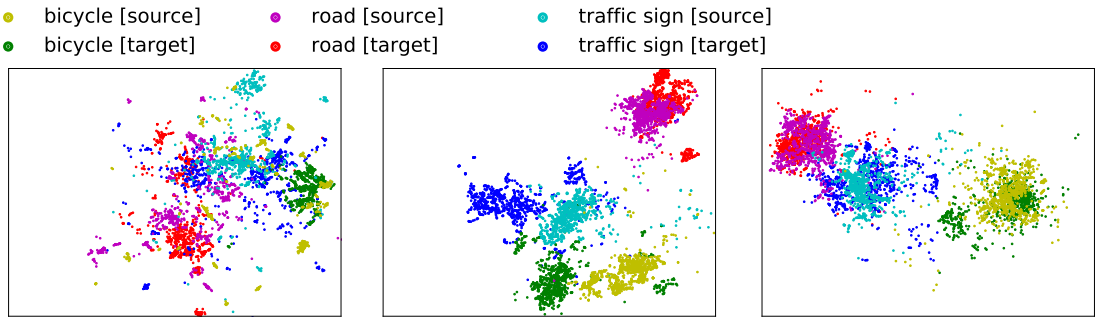
Our approach consists of a pre-training stage (as illustrated in Fig. 7.3) and a self-training stage (Fig. 7.2).

We pre-train our segmentation network in the pre-training stage (Sec. 7.4), and the pre-trained network is then used to initialize the teacher and student networks in the self-training stage (Sec. 7.5). Here, the teacher network generates pseudo labels as supervision for the student network. The student network is trained via a standard cross-entropy loss for direct supervision and a novel contrastive training objective (Sec. 7.5.2). Furthermore, we propose a label expansion mechanism (Sec. 7.5.3) to extend the set of samples for which pseudo labels are generated.

7.4 Pre-training

Before training, we first translate images from the source domain to the style of the target domain via an off-the-shelf image-to-image translation network [108]. This translation step helps bootstrapping our method as it reduces the visual difference between domains. Note that we only apply the translation during pre-training and later operate directly on unmodified images from the source domain, which we augment for robustness. Figure 7.3 illustrates the pre-training phase. We start by training a semantic segmentation network via direct supervision on the translated images and source domain labels. Training the network on the source domain with direct supervision serves as a strong initialization. Since the images have been translated to the target domain, the initially learned visual representations are already partially aligned with the target domain. Next, we train the network solely on the target domain with a contrastive objective. The same objective is used for both pre-training and self-training. We defer a detailed discussion to Section 7.5.2.

After adapting the visual representations, we fine-tune the network again on



a) w/o contrastive learning b) only in-domain c) in-domain & cross-domain

Figure 7.4: Visualization of representation compression and alignment. We visualize the learned feature space for a subset of classes from the source (GTA5) and target (Cityscapes) domains with UMAP [188]. a) Without any contrastive objective, representations for individual classes overlap. b) Applying a contrastive objective within domains leads to more discriminative representations for individual classes, but representations are misaligned between domains. c) Adding a cross-domain contrastive objective aligns the classes from different domains while keeping the representation discriminative.

translated images of the source domain and ground truth labels. For this, we fix the backbone network and train only the segmentation head with a reduced learning rate of 0.001 for 5 epochs.

7.5 Self-training

Following the self-training paradigm [271], we employ a teacher network for generating pseudo labels. The teacher network is initialized with the same weights as the student network. While we update the weights of the student network in every training iteration, the weights of the teacher network are only updated every 200 iterations by copying from the student network.

7.5.1 Pseudo Label Generation

To generate pseudo labels for images from the target domain, we randomly sample 713×713 crops (without augmentation) from these images and pass them through

the teacher network. The same crop is augmented before it is ingested by the student network (see Sec. 7.5.4). As typical for semantic segmentation networks, our teacher network predicts class probabilities. As in prior work, we take the most confident prediction per pixel as pseudo label and employ the output probability as confidence measure [359, 181]. We follow Ma *et al*. [181] and ignore predictions below an adaptively computed, class-dependent confidence threshold (detailed in the supplement). To increase the robustness of the generated pseudo labels, we apply the teacher network at multiple scales (0.5,0.75,1.0,1.25,1.5,1.75) and average predictions over scales. The student network is supervised with the pseudo labels directly via a cross-entropy loss. Furthermore, we use them to formulate a contrastive objective.

7.5.2 Contrastive Objective

Contrastive learning aims to learn a representation space in which similar entities cluster closely together, while simultaneously pushing apart dissimilar entities [41, 86, 44]. Importantly, the notion of what constitutes similarity between entities is entirely defined by data and can be steered by an appropriate selection of similar (positive) pairs of samples together with dissimilar (negative) pairs.

We follow SimCLR [41] and add a 2-layer MLP head to our student network to learn a 128-dimensional latent feature representation f per pixel. We leverage the contrastive loss proposed by SupContrast [130]:

$$\mathcal{L}_{ctr}(\boldsymbol{v}, \boldsymbol{v}^+) = -\log \frac{\exp(\frac{\boldsymbol{v}\cdot\boldsymbol{v}^+}{\tau})}{\exp(\frac{\boldsymbol{v}\cdot\boldsymbol{v}^+}{\tau}) + \sum_{\boldsymbol{v}^-} \exp(\frac{\boldsymbol{v}\cdot\boldsymbol{v}^-}{\tau})}, \tag{7.1}$$

which maximizes similarity between positive feature pairs ($\boldsymbol{v}/\boldsymbol{v}^+$, *i.e*. of the same class), and minimizes it between negative pairs ($\boldsymbol{v}/\boldsymbol{v}^-$, *i.e*. of different classes). We set the temperature $\tau = 0.07$ in our experiments.

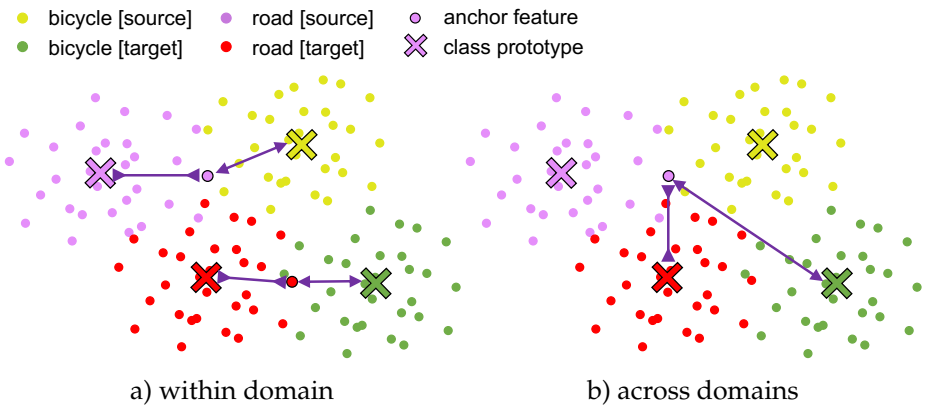


Figure 7.5: We employ different sampling strategies for obtaining corresponding contrastive pairs. Starting from a sampled anchor feature, we match class prototypes within domain to compress representations (a), and across domains to align them (b). Figure 7.4 illustrates the effect on the representations learned.

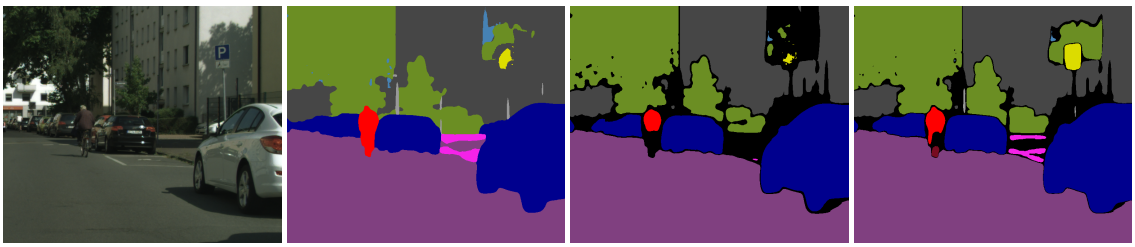


Figure 7.6: Pseudo label expansion. For a target domain image (a), our teacher network predicts an initial pseudo label map (b), from which we retain only high-confidence predictions (c). For low-confidence predictions, we assign the label of the closest pseudo label prototype and obtain our final pseudo label map (d). The expansion commonly corrects the labels of "hard classes" such as *bicycle* & *traffic sign* here.

We employ three different strategies to assemble contrastive pairs. First, as ground truth labels are available in the source domain, we exploit this knowledge to sample positive and negative pairs in the source domain, as proposed for supervised segmentation [301]. Second, we construct class-dependent feature prototypes $\overline{f}_c$ for each domain to compress the learned representations. Third, we leverage the prototypes to further align source and target domain.

A prototype for a class c is assembled as a weighted average of features:

$$\overline{f_c} = \frac{1}{\sum_{f \in F_c} \omega(f, c)} \sum_{f \in F_c} \omega(f, c) \cdot f, \tag{7.2}$$

where F_c is a set of admissible features f. We weight the contribution of individual features by $\omega(f, c)$, which is the softmax probability of the respective pixel as provided by the teacher network. For computing prototypes on the images of the source domain, we do not need to resort to the teacher network as ground truth is available, and we simply set $\omega(f, c) \coloneqq 1$. The set F_c contains all feature vectors for which weights exceed a class-dependent threshold τ_c:

$$F_c = \{f \mid c = \arg\max_c \ \omega(f, c) \wedge \omega(f, c) > \tau_c\}. \tag{7.3}$$

We build the prototypes for each mini-batch during training.

To compress feature representations with the prototypes in our contrastive objective, we sample anchor features $\boldsymbol{v}$ and pair them with respective class prototypes from the same domain – we create positive pairs with the class prototype $\overline{f_c}$ of the corresponding class and negative pairs with other prototypes. This is visualized in Figure 7.5 (a), and the effect on the learned feature representations can be seen in Figure 7.4 (b). Our class-dependent prototypes resemble the category-centers in the regularization approach of Ma *et al*. [181]. Different from their work, however, we incorporate the prototypes into a contrastive framework, where we use them to align learned representations across domains explicitly.

To enforce alignment of feature representations across domains, we further pair anchors from the source domain with the corresponding class prototypes from the target domain, see Figure 7.5 (b). This has the effect of pulling the representations learned for specific classes on images from different features together and can be observed in the learned representations in Figure 7.4 (c).

7.5.3 Pseudo Label Expansion

A common issue in self-training is the generation of reliable pseudo labels for a diverse set of samples. As the teacher network represents an earlier or ensembled version of the student network, it inherits the student network's classification capabilities. It typically classifies easy examples with high confidence, but hard examples, which would be informative for training, only with low confidence. To extend the set of pseudo-labeled examples we could simply lower the confidence threshold, but this inevitably increases the number of mislabeled training examples, which in turn reduces accuracy.

A possible cause for hard examples is a missing clear correspondence between source and target domain [33, 63]. We show an example for adapting traffic signs from GTA 5 to Cityscapes in Figure 7.1. While traffic signs are present in both domains, they typically belong to a set of hard classes, which are misclassified to a large extent. Prior work attributed this to a smaller pixel footprint in the dataset [39], but we empirically found that even after adjusting for this, traffic signs keep getting misclassified. We hypothesize that the underlying reason is a limited diversity in the source domain paired with a high diversity in the target domain.

To mitigate this discrepancy, we propose a pseudo label expansion mechanism. Intuitively, we want to find more samples of hard-to-label classes to which we can assign a label reliably. For this, we take inspiration from the class-dependent prototypes from Sec. 7.5.2 (from here on termed *contrastive* prototypes to avoid confusion), and compute prototypes from features of the teacher network here, which we term *pseudo-label prototypes*.

As with the contrastive prototypes, we assemble pseudo-label prototypes from features extracted at the penultimate network layer. To increase robustness, we accumulate the features from which we compute the prototypes over 200 training iterations, instead of a single mini-batch as with the contrastive ones. Starting from

an input image of the target domain (see Fig. 7.6 (a)), we obtain an initial pseudo label map (Fig. 7.6 (b)). We treat the softmax probabilities of the network as a measure of confidence and retain only labels for which they exceeds a class-dependent threshold (Fig. 7.6 (c)). For simplicity, we use the same adaptive mechanism [181] as for the contrastive prototypes. We assign pseudo labels to remaining low-confidence predictions. To that end, we match each low-confidence feature with the closest pseudo-label prototype and assign the respective label. If there exists no suitable pseudo-label prototype within a maximal distance, we do not assign a label. For details on the distance computation we refer to the supplementary material.

As shown in Figure 7.6 (d), our pseudo label expansion is able to correct mispredictions of "hard samples", typically belonging to classes with few training samples and a considerable difference in diversity between source and target domain. This is further confirmed in the quantitative evaluation we conduct in Section 7.6.

7.5.4 Training & Implementation Details

Augmentations. To learn robust features and improve generalization, we augment the images fed to the student network. Specifically, we apply random color transformations, random scaling and rotations, as well as CutOut [62]. Geometric transformations are accordingly applied to the pseudo label maps to ensure correspondence between image content and labels. For more details we refer to the supplemental material.

Contrastive pairs. For each mini-batch, we use ground truth and pseudo labels to sample K features per class in each domain. These features represent anchors $\boldsymbol{v}$. To build contrastive correspondences $\boldsymbol{v}^{+/-}$ for these anchors, we employ a momentum strategy [86] and store the K most recent feature embeddings per class at the pixel and prototype level. We use $K = 1000$ in our experiments. Finally, $(\boldsymbol{v}, \boldsymbol{v}^{+})$ from the same category are used as the positive pairs. All samples $\boldsymbol{v}^{-}$ from other

categories are used to build the negative pairs $(\boldsymbol{v}, \boldsymbol{v}^-)$. Sampling the same number of contrastive pairs per class mitigates the effect of an unbalanced distribution of samples per class. Note that even if no samples of a particular class are available as anchors in a mini-batch, samples of that class may still participate in the contrastive loss through the momentum strategy applied to the pixel-level and prototype-level embeddings.

Loss. We train our method with a cross-entropy loss and a contrastive loss. The cross-entropy is applied for direct supervision of the network per pixel:

$$\mathcal{L}_{cls} = \mathcal{L}_{CE}(s, y_s) + \lambda \mathcal{L}_{CE}(t, \hat{y}_t), \tag{7.4}$$

where y is a ground truth label from the source domain, $\hat{y}$ a pseudo label, and s/t are the source or target domain predictions by the student network. A weight $\lambda = 0.1$, applied to samples from the target domain, accounts for imperfect supervision from pseudo labels.

The in-domain and cross-domain contrastive objectives are based on the contrastive loss of Eq. (7.1). The in-domain contrastive losses are given by:

$$\mathcal{L}_{in} = \mathcal{L}_{ctr}(s, s^+) + \mathcal{L}_{ctr}(t, t^+), \tag{7.5}$$

where (s, s^+) are contrastive pairs from the source domain and (t, t^+) are the contrastive pairs from the target domain. Note that s^+ can be either a pixel representation or class prototype. t^+ represents the class prototypes in the target domain.

The cross-domain contrastive loss is given by

$$\mathcal{L}_{cross} = \mathcal{L}_{ctr}(s, t^+), \tag{7.6}$$

where s are the pixels from the source domain and t^+ are the class prototypes computed on the target domain.

The final training loss is given by

$$\mathcal{L} = \mathcal{L}_{cls} + \alpha(\mathcal{L}_{in} + \mathcal{L}_{cross}). \tag{7.7}$$

We set $\alpha = 0.1$ in all experiments.

Training details. All experiments were conducted on 8 24GB GPUs with 2 samples per GPU. We use momentum of 0.9 and weight decay of 1e-4 for training. All other training settings are the same as [367] (detailed in the supplement). We start with an initial learning rate of 0.01 for pre-training on the source domain. In the self-training phase, we train for 25K iterations and the learning rate starts at 1e-3.

7.6 Experiments

We follow the evaluation protocol that was proposed in [360, 181] and evaluate our method with source domain datasets GTA5 [234] and SYNTHIA[238], and with target domain datasets Cityscapes [58] and Mapillary Vistas [201] (Vistas in the supplement). The GTA5 dataset shares 19 common classes with Cityscapes and we ignore classes that are not shared during training. SYNTHIA shares 16 classes with Cityscapes. Some existing works only train and test on a 13-class subset of SYNTHIA, or train two separate models on the subset and on the full set. Here we follow the practice introduced in [204, 294] and train a model on the full label set but test it on both settings (13 & 16 classes). We use the training set (ignoring the labels) from the respective target domain to perform unsupervised adaptation for all experiments. We report results in terms of per-class Intersection over Union (IoU) as well as the mean IoU (mIoU) over all classes on the respective validation sets.

Table 7.1: Comparison to prior work on GTA5→Cityscapes. All methods use a DeepLabV2 (ResNet101) backbone, except Coarse-to-fine [181] and RPT [364], which use DeepLabV3 (ResNet-101) and FCN (ResNet-101) respectively.

	road	sidewalk	building	wall	fence	pole	light	sign	veget.	terrain	sky	person	rider	car	truck	bus	train	m.cycle	bicycle	mIoU
BDL [160]	91.0	44.7	84.2	34.6	27.6	30.2	36.0	36.0	85.0	43.6	83.0	58.6	31.6	83.3	35.3	49.7	3.3	28.8	35.6	48.5
IDA [204]	90.6	36.1	82.6	29.5	21.3	27.6	31.4	23.1	85.2	39.3	80.2	59.3	29.4	86.4	33.6	53.9	0.0	32.7	37.6	46.3
BiMaL [283]	91.2	39.6	82.7	29.4	25.2	29.6	34.3	25.5	85.4	44.0	80.8	59.7	30.4	86.6	38.5	47.6	1.2	34.0	36.8	47.3
DTST [308]	90.6	44.7	84.8	34.3	28.7	31.6	35.0	37.6	84.7	43.3	85.3	57.0	31.5	83.8	42.6	48.5	1.9	30.4	39.0	49.2
FGGAN [294]	91.0	50.6	86.0	43.4	29.8	36.8	43.4	25.0	86.8	38.3	87.4	64.0	38.0	85.2	31.6	46.1	6.5	25.4	37.1	50.1
FDA [344]	92.5	53.3	82.3	26.5	27.6	36.4	40.5	38.8	82.2	39.8	78.0	62.6	34.4	84.9	34.1	53.1	16.8	27.7	46.4	50.4
CAG [360]	90.4	51.6	83.8	34.2	27.8	38.4	25.3	48.4	85.4	38.2	78.1	58.6	34.6	84.7	21.9	42.7	41.1	29.3	37.2	50.2
Uncertainty [306]	90.5	38.7	86.5	41.1	32.9	40.5	48.2	42.1	86.5	36.8	84.2	64.5	38.1	87.2	34.8	50.4	0.2	41.8	54.6	52.6
SAC [4]	90.4	53.9	86.6	42.4	27.3	45.1	48.5	42.7	87.4	40.1	86.1	67.5	29.7	88.5	49.1	54.6	9.8	26.6	45.3	53.8
RPT [364]* (FCN-101)	89.7	44.8	86.4	44.2	30.6	41.4	51.7	33.0	87.8	39.4	86.3	65.6	24.5	89.0	36.2	46.8	17.6	39.1	58.3	53.2
coarse-to-fine [181] *	92.5	58.3	86.5	27.4	28.8	38.1	46.7	42.5	85.4	38.4	91.8	66.4	37.0	87.8	40.7	52.4	44.6	41.7	59.0	56.1
BAPA-Net [174]	94.4	61.0	88.0	26.8	39.9	38.3	46.1	55.3	87.8	46.1	89.4	68.8	40.0	90.2	60.4	59.0	0.00	45.1	54.2	57.4
ProDA [359]	87.8	56.0	79.7	46.3	44.8	45.6	53.5	53.5	88.6	45.2	82.1	70.7	39.2	88.8	45.5	59.4	1.0	48.9	56.4	57.5
Ours	92.6	59.1	**88.5**	45.8	40.5	**52.9**	**53.6**	54.1	88.0	41.9	86.0	**73.5**	**44.1**	89.7	39.3	53.2	26.8	**51.6**	**61.8**	**60.2**

Table 7.2: Comparison to prior work on adapting SYNTHIA→Cityscapes (mIoU: 16-class; mIoU*: 13-class).

	road	sidewalk	building	wall	fence	pole	light	sign	veget.	sky	person	rider	car	bus	m.cycle	bicycle	mIoU	mIoU*
BDL [160]	86.0	46.7	80.3	-	-	-	14.1	11.6	79.2	81.3	54.1	27.9	73.7	42.2	25.7	45.3	-	51.4
IDA [204]	84.3	37.7	79.5	5.3	0.4	24.9	9.2	8.4	80.0	84.1	57.2	23.0	78.0	38.1	20.3	36.5	41.7	48.9
DTST [308]	83.0	44.0	80.3	-	-	-	17.1	15.8	80.5	81.8	59.9	33.1	70.2	37.3	28.5	45.8	-	52.1
FGGAN [294]	84.5	40.1	83.1	4.8	0.0	34.3	20.1	27.2	84.8	84.0	53.5	22.6	85.4	43.7	26.8	27.8	45.2	52.5
FDA [344]	79.3	35.0	73.2	-	-	-	19.9	24.0	61.7	82.6	61.4	31.1	83.9	40.8	38.4	51.1	-	52.5
CAG (13 classes) [360]	84.8	41.7	85.5	-	-	-	13.7	23.0	86.5	78.1	66.3	28.1	81.8	21.8	22.9	49.0	-	52.6
CAG (16 classes) [360]	84.7	40.8	81.7	7.8	0.0	35.1	13.3	22.7	84.5	77.6	64.2	27.8	80.9	19.7	22.7	48.3	44.5	-
BiMaL [283]	92.8	51.5	81.5	10.2	1.0	30.4	17.6	15.9	82.4	84.6	55.9	22.3	85.7	44.5	24.6	38.8	46.2	53.7
Uncertainty[306]	79.4	34.6	83.5	19.3	2.8	35.3	32.1	26.9	78.8	79.6	66.6	30.3	86.1	36.6	19.5	56.9	48.0	54.6
coarse-to-fine[181]	75.7	30.0	81.9	11.5	2.5	35.3	18.0	32.7	86.2	90.1	65.1	33.2	83.3	36.5	35.3	54.3	48.2	55.5
BAPA-Net[174]	91.7	53.8	83.9	22.4	0.8	34.9	30.5	42.8	86.6	88.2	66.0	34.1	86.6	51.3	29.4	50.5	53.3	61.2
ProDA[359]	87.8	45.7	84.6	37.1	0.6	44.0	54.6	37.0	88.1	84.4	74.2	24.3	88.2	51.1	40.5	45.6	55.5	62.0
Ours	85.2	46.5	83.3	**39.2**	**6.1**	37.3	50.1	40.1	87.9	88.2	70.1	29.8	85.4	45.4	**59.1**	49.8	**56.5**	**63.1**

Comparisons to existing work. Tab. 7.1 and Fig. 7.7 show the results when adapting from GTA5 to Cityscapes. Our approach outperforms existing work by a large margin in this experiment. It shows a mIoU of 60.2% and achieves the highest IoU on 7 out of 19 classes. 6 of these classes are what can be considered "hard classes" (*pole*, *traffic light*, *person*, *rider*, *motorcycle*, and *bicycle*) due to their small footprint in the dataset or because of large per-class domain shift. The performance improvement of our method mostly comes from these challenging classes. Table 7.2 shows a similar experiment where we adapt from a different source domain (SYNTHIA) to Cityscapes. Our method again achieves state-of-the-art results with

an mIoU of 56.5% and 63.1% for 16 and 13 classes, respectively. We again see a strong performance increase on hard classes, such as *wall*, *fence*, and *motorcycle*. Finally, we show a comparison to recent state-of-the-art methods on adaptation to a different target domain (GTA5 to Mapillary Vistas) in the supplement. Again, our method consistently outperforms prior work.

Ablation. We show the influence of different components of our approach on the GTA5 to Cityscapes adaptation task in Table 7.3. For the pre-training step, both style transfer (*transfer*) and contrastive learning (*pre-training*) provide large performance improvements. These components alone improve the baseline from 37.6% to 50.1%. Contrastive learning further improves the mIoU from 50.1% to 57.9%. Our approach without label expansion already outperforms all existing works. Enabling label expansion leads to an additional performance boost of more than 2.3%. Our ablation also shows that the label expansion step strongly benefits from contrastive learning, as contrastive learning yields more high confidence predictions particularly for hard classes (Table 7.4). When we use label expansion without the contrastive feature alignment, we see a moderate improvement of 1.8% compared to the relatively weak baseline that shows 50.1% mIoU. When using label expansion together with contrastive feature alignment, we see an improvement from 57.9% to 60.2%. That is, we see a larger improvement over a significantly stronger baseline. We also evaluate our method with different network backbones and provide additional ablations of various hyper-parameters in the supplementary material.

Effect of contrastive learning. The effect of contrastive learning on feature alignment across domains is shown in Figure 7.4. We visualize the learned feature space of a subset of classes from the GTA5 and Cityscape domains using UMAP [188]. The in-domain contrastive learning concentrates samples close to their category centers while pushing the category centers apart. This leads to a well separated feature space for every class inside the respective domain. However, when applying the

Table 7.3: Ablation study on GTA5→Cityscapes. We find that all of our contributions improve model performance and our full model (#7) performs best. Interestingly, label expansion is boosted through a contrastive objective (compare #3 → #4 vs. #6 → #7).

condition	transfer	pre-training	contrast	label expansion	mIoU
1					37.6
2	√				45.7
3	√	√			50.1
4	√	√		√	51.9
5		√	√	√	57.5
6	√	√	√		57.9
7	√	√	√	√	**60.2**

Table 7.4: Improvements over "hard classes" after implementing the pseudo label expansion.

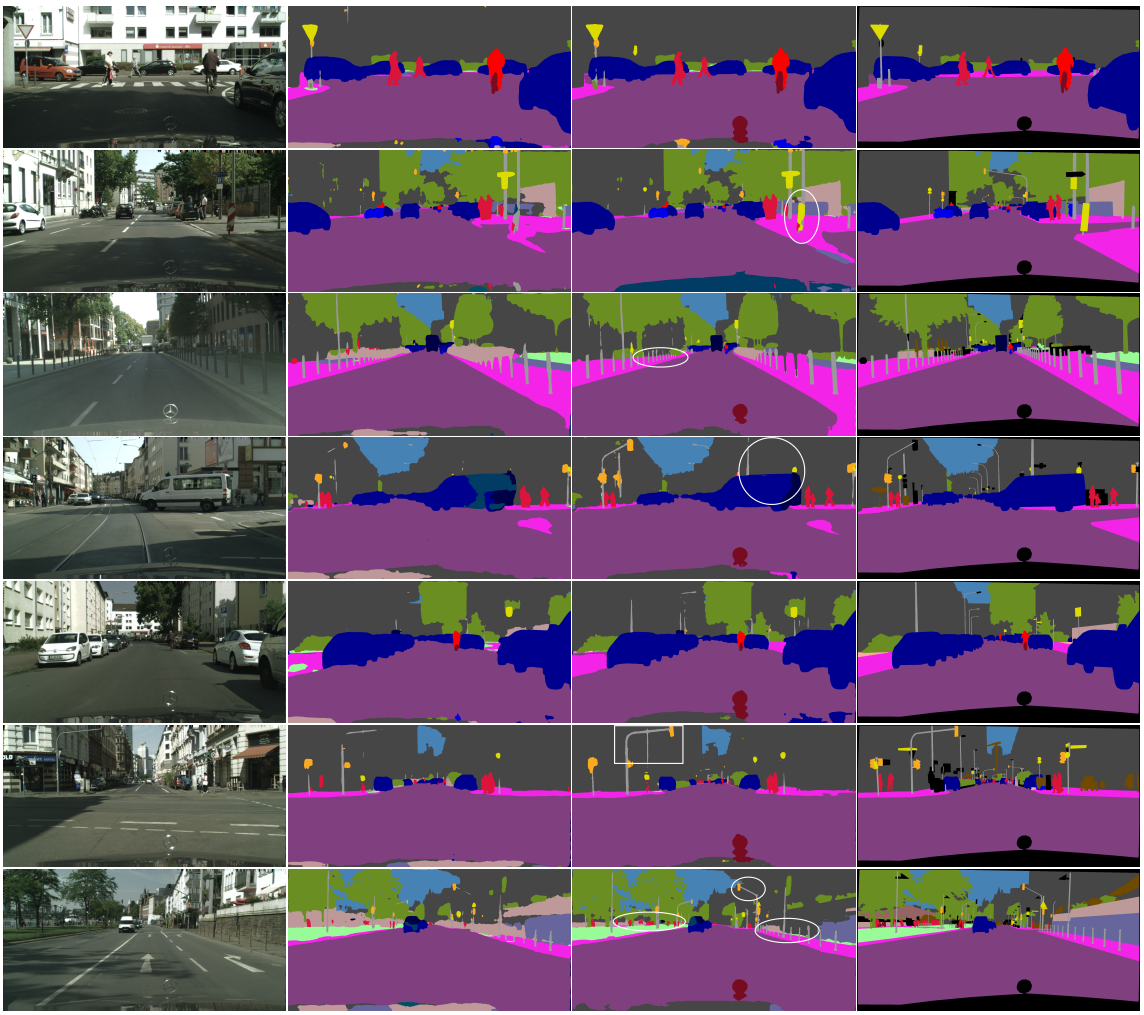
	pole	light	sign	person	rider	train	m.cycle	bicycle	mIoU
w/o expansion	49.2	48.5	45.8	70.2	41.4	21.5	40.1	53.4	47.5
w/ expansion	52.9	53.6	54.1	73.5	44.1	26.8	51.6	61.8	53.1
improvements (%)	7.5	10.5	18.1	4.7	6.5	24.7	28.7	15.7	11.8

same feature extractor to a different domain, we observe that this separation is lost. By adding cross-domain contrastive learning the differences between domains can be mitigated, by aligning the features that correspond to individual classes across domains.

Effect of label expansion. As illustrated qualitatively in Figure 7.6, label expansion increases the number of pixels with valid pseudo-labels from hard classes (see the traffic sign and the bicycle highlighted in Figure 7.6 (d)). This additional data positively affects learning and boosts final mIoU from 57.9% to 60.2% in our GTA5→Cityscapes experiment (see Table 7.3). It particularly improves results on hard classes. As shown in Table 7.4, *traffic light*, *sign*, and *bicycle* are improved by more than 10% whereas *motorcycle* and *train* are improved by 20∼30% when compared to the baseline without label expansion.

7.7 Conclusion

We introduced contrastive learning for unsupervised domain adaptation in semantic segmentation. We leverage both in-domain contrastive samples as well as cross-domain contrastive samples that bridge the source and target domains. Our approach achieves robust class-based feature alignment to facilitate domain adaptation. Our framework is based on a student-teacher architecture that generates pseudo-labels, which guide the selection of contrastive pairs. We introduced a label expansion strategy to discover reliable pixels from particularly hard classes. Our method achieves state-of-the-art domain adaptation results for a variety of source and target domains. It significantly improves results for hard classes where only a few pixels are available in the source domain or ones that are affected by strong class-specific domain shift between the domains.



(a) Inputs (b) ProDA (c) Ours (d) Ground Truth

Figure 7.7: Comparison to prior work. (a) Input images from Cityscapes. (b) ProDA [359]. (c) Results of our method. (d) Ground truth. As highlighted by the white circles and rectangles, the improvements are mainly in the "hard classes" (pole, traffic signs, traffic light and person *etc.*)

Chapter 8

Deep FusionNet for Point Cloud Semantic Segmentation

Feihu Zhang

University of Oxford

Jin Fang

Baidu Research, Baidu Inc.

Benjamin Wah

The Chinese University of Hong Kong

Philip H. S. Torr

University of Oxford

Abstract

Many point cloud segmentation methods rely on transferring irregular points into a voxel-based regular representation. Although voxel-based convolutions are useful for feature aggregation, they produce ambiguous or wrong predictions if a voxel contains points from different classes. Other approaches (such as PointNets and point-wise convolutions) can take irregular points for feature learning. But their high memory and computational costs (such as for neighborhood search and ball-querying) limit their ability and accuracy for large-scale point cloud processing. To address these issues, we propose a deep fusion network architecture (FusionNet) with a unique voxel-based "mini-PointNet" point cloud representation and a new feature aggregation module (fusion module) for large-scale 3D semantic segmentation. Our FusionNet can learn more accurate point-wise predictions when compared to voxel-based convolutional networks. It can realize more effective feature aggregations with lower memory and computational complexity for large-scale point cloud segmentation when compared to the popular point-wise convolutions. Our experimental results show that FusionNet can take more than one million points on one GPU for training to achieve state-of-the-art accuracy on large-scale Semantic KITTI benchmark.

8.1 Introduction

Semantic segmentation of 3D Point cloud is widely applicable in many scenarios, including remote sensing, AR/VR, robotics, and self-driving cars. Many deep neural network models have been proposed for this important task.

Approaches typically rely on a voxel-based regular representation that converts

unordered points to regular 3D voxel/grids before using 3D/2D convolutions for feature learning [314, 220, 264, 314, 185, 220, 147, 99]. Inspired by the success in image-based segmentation, these volumetric networks can be efficiently designed and trained for semantic segmentation of 3D point clouds, using regular convolutions which have been shown useful for coarse-grain feature aggregation/learning [80, 56, 354]. However, they can only give voxel-level predictions. Moreover, their voxelization process transfers many raw points to one single voxel that will produce ambiguous or wrong predictions at object borders when voxels consist of points from different classes (as illustrated in Fig. 8.1(c)).

When compared to regular convolution operations, PointNets [219, 221] can take irregular points for feature learning. However, the Multi-Layer-Perceptron (MLP) [219] in PointNets keeps only the most significant activation and may lose some useful detailed information for segmentation. Also, these models are limited for training deep and robust networks for large-scale point clouds (Fig. 8.1(a)) because they have high memory and computational complexity in their neighborhood search, sampling, and ball-querying operations [219, 221, 299, 331].

There is also plenty of work on point-based convolutions that explores the idea of convolutions directly on irregular points [175, 326, 155, 276, 81, 290]. They learn to approximate a weight function or interpolate convolutional weights [326, 155, 312, 299, 205, 92, 183]. Compared to volumetric convolutions that can directly index a kernel with fixed relative positions of the neighborhoods, the positions of neighbors become unpredictable in these point-wise convolutions as the points are scattered irregularly. Hence, the kernel for neighboring points must be calculated on the fly. The additional memory costs and matrix multiplications will limit the training of effective networks for large-scale point clouds and may produce wrong predictions in some objects due to insufficient feature aggregations (Fig. 8.1(b)).

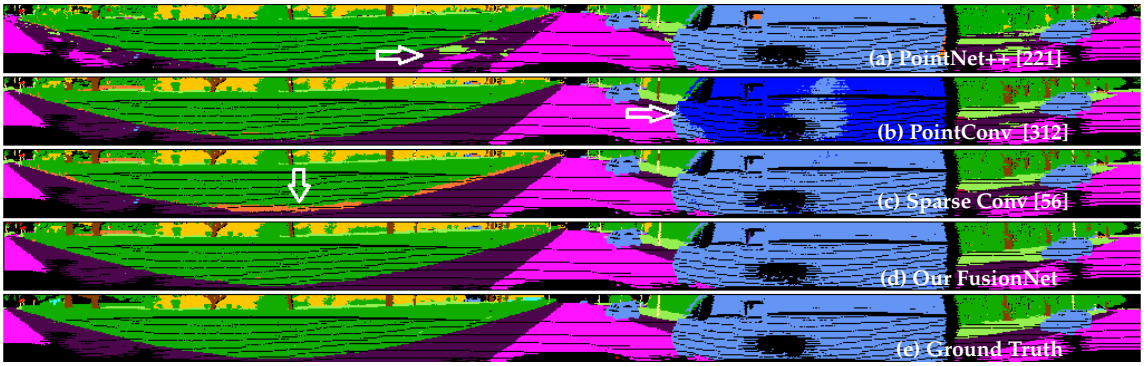


Figure 8.1: Problem illustrations with large-scale point clouds (Semantic KITTI [17]). The LiDAR frame contains more than 120K points in a large 3D space of $160m \times 160m \times 20m$. Results are projected into 2D cylindrical images for visual comparisons. *Top rows (a) and (b):* State-of-the-art PointNet++ [221] and point-wise convolutions [312] fail in large objects/areas (*e.g.* sidewalk, car) due to insufficient feature propagations/aggregations for large-scale point clouds. *Third row (c):* State-of-the-art sparse convolutions [56] predict wrong labels at the border between different classes due to their ambiguous and coarse voxel-level learning. *Fourth row (d):* Our FusionNet gives accurate segmentation labels for the large-scale point cloud. *Last row (e):* ground truth.

To address the challenges of designing effective network architectures for accurate point-wise segmentation of large-scale point clouds, we develop a deep fusion network architecture with a unique voxel-based "mini-PointNet" structure for point cloud representation and a novel fusion module (Fig. 8.2) for feature aggregation. The proposed FusionNet realizes both the voxel aggregation and the fine-grain point-wise feature learning. It inherits all the advantages (in terms of effectiveness and efficiency) of volumetric convolutional networks (*e.g.*, 3D UNet [237]) while being able to learn point-wise features for accurate label predictions.

FusionNet possesses many advantages over existing models in large-scale point cloud segmentation:

1) When compared to existing voxel networks [80, 56, 354], FusionNet can predict point-wise labels and avoid those ambiguous/wrong predictions when a voxel has points from different classes.

2) When compared to the popular PointNets [219, 221, 299, 331] and point-based convolutions [312, 155], FusionNet has more effective feature aggregation operations (including the efficient neighborhood-voxel aggregations and the fine-grain inner-voxel point-level aggregations). These operations help produce better accuracy for large-scale point cloud segmentation.

3) FusionNet takes full advantage of the sparsity property to reduce its memory footprint. For instance, it can take more than one million points in training and use only one GPU to achieve state-of-the-art accuracy.

With an effective feature aggregation module and lower memory and computation costs, we can train our FusionNet to learn more effective deep features for accurate large-scale point cloud segmentation (As illustrated in Fig. 8.1d). In our experiments, FusionNet achieves state-of-the-art performance on large-scale point cloud datasets. Especially, it outperforms state-of-the-art PointNets [221] (by 40%), point-wise convolutions [312] (by 10%) and sparse CNN [56] (by 7%) in mean IoU evaluations on the challenging Semantic KITTI benchmark.

8.2 Related Work

Existing approaches for 3D point cloud segmentation can be roughly categorized into two types: regular voxel-based networks and irregular point-based networks.

8.2.1 Voxel-based Networks

There is substantial previous work on 3D CNN to convert point clouds to 2D or 3D volumetric grids/voxels (or similar slices/lattices) [314, 185, 220, 354]. For example, 3D point clouds or shapes have been projected into several 2D image-like grids with 2D convolutions for shape classification and retrieval tasks [264, 220]. Other studies

[314, 185, 220, 147, 99] voxelize point clouds into 3D volumetric voxels and apply 3D convolution networks for point cloud processing and understanding.

Among these studies, VV-Net [190] uses a kernel-based interpolated variational auto encoder (VAE) on regular voxel grids; instance segmentation models have been proposed on dense 3D voxels/2D grids [140, 354]; panoptic labels can be predicted using a spatially hashed volumetric map [200]; high-resolution RGB inputs have been leveraged by associating 2D images with the volumetric grid [99]; efficient feature aggregations have been explored in PVCNN [176] through a voxel-based regular CNN in low resolution to avoid random memory accesses.

To extend voxel-based networks to high-resolution scene segmentation tasks [60, 17], a set of unbalanced octrees have been used to improve the resolution [235]. Sparse Submanifold CNN [56, 80] computes the convolutions only at activated points to design high-resolution volumetric inputs. However, higher resolution inputs mean more sparse inputs, making convolutional layers less efficient for feature aggregation. Also, it will increase memory and computation costs.

8.2.2 Point-based Networks

Recent work takes raw points as input for feature learning and label predictions.

PointNets: These methods aggregate features from different points by shared multi-layer perceptrons (MLP) [219]. PointNet++ improves its network by adding a hierarchical structure [221]. Wang *et al*. associate instances and semantics segmentation [297] with feature encoder of stacked PointNet layers. He *et al*. learn deep geodesic-aware representations for PointNet/PointNet++ [90]. The issue with the MLP module is that they keep only the most significant activation on features, leading to the loss of some useful detailed information for segmentation.

Point-wise Convolutions: Other recent work explores the extension of convolutions on irregular and unordered point-cloud inputs [175, 326, 155, 276, 81, 290].

For unordered points, PointCNN performs convolutions on transformed points [155]. Zhang *et al*. use statistics from concentric spherical shells to resolve point-order ambiguities [365]. FeaStNet generalizes conventional convolution layers by adding a soft-assignment matrix [290]. KPConv learns flexible and deformable point convolutions [276]. Point-wise Conv [104, 312], GeoCNN [143] and annular convolution [135] query the nearest neighbors for convolutions with different kernels.

Some work explores contiguous convolutions [205, 312, 299] for point clouds. For instance, PointConv [312] treats convolution kernels as nonlinear functions of the coordinates and uses (MLP) modules to learn continuous weight functions for point-wise convolutions. Hermosilla *et al*. develop Monte Carlo convolutions for learning non-uniformly sampled point clouds [92]. Similarly, Mao *et al*. propose to interpolate discrete convolutional weights for convolutions on points [183].

Other work explores the ideas of convolutions on unique surfaces. Rao *et al*. map 3D points onto a discretized sphere for convolution definition [225]. Huang *et al*. utilize a 4-rotational symmetric field to define a domain for convolution on a surface [106]. Tangent convolution projects local surface geometry on a tangent plane around every point, producing planar convolution-able tangent images [272].

In the original regular 2D/3D convolutions, the relative positions of the neighborhoods are fixed, and the convolutional kernel can be directly indexed. However, for point-based convolutions, the points are irregularly scattered over a 3D space, making the relative positions of neighbors unpredictable. Hence, the kernel for each neighboring point must be calculated on the fly using additional matrix multiplications.

There are also graph or tree-based convolutions for point clouds, including local spectral graph convolutions [292], graph attention convolutions [297], regularized graph CNN (RGCNN) [273], and octree guided CNN with spherical kernels [149].

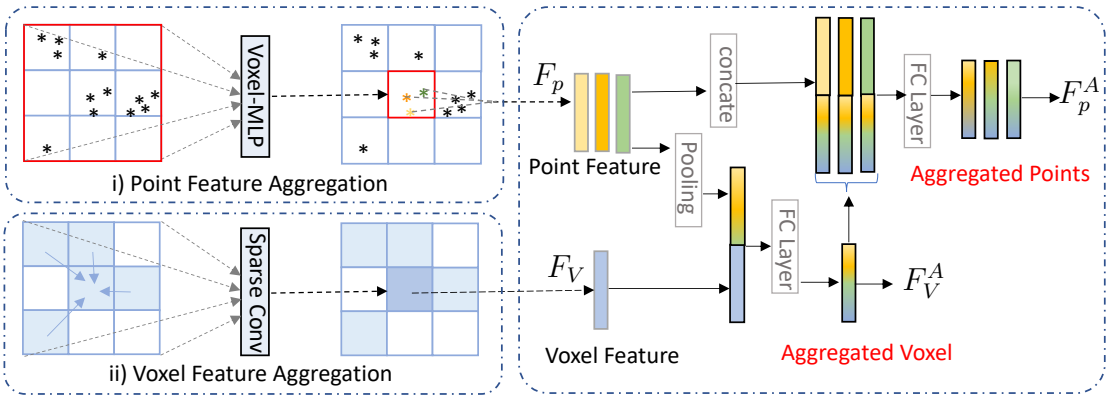
The unique feature of them is that they often rely on a graph construction and a learned kernel function for convolutions on unordered points. These are realized by additional modules/layers with extra computations and memory overhead which limit their abilities for designing effective deep architectures in large-scale point cloud processing.

Other Point Networks: There are also special networks developed for various applications to directly take raw points as input for sampling [338, 348], semantic [303, 366, 123, 346, 368, 231], and instance segmentation [213, 331]. Some employ new and special data structures for point cloud processing, including 3D point capsule encoder and decoder networks [371], over segmentation on graph structure [144], basis point sets [217], multi-resolution tree-structured networks [72], bilateral convolutions on a sparse lattice of the point cloud [263], recurrent neural networks (RNN) on slices of a point cloud [107], and the superpoint graph-based semantic segmentation [145].

8.3 FusionNet

Many previous models are either limited by the ambiguous voxel-level predictions [80, 185, 354] or have insufficient feature aggregations that use high memory or computational costs [155, 176, 354, 221] in large-scale point-cloud segmentation. This section describes our deep FusionNet model that can learn accurate point-wise features and realize more effective and memory-efficient feature aggregations in large-scale processing. It utilizes a unique voxel-based "mini-PointNet" for sparse representation and the novel FusionNet modules for feature aggregations.

Fig. 8.2 illustrates the FusionNet module that includes both **neighborhood voxel aggregation** (Fig. 8.2(a)) for voxel-level learning and **inner-voxel aggregation** for fine-grain point-level learning (Fig. 8.2(b)). The rest of this section is organized as



(a) Neighborhood Aggregations (b) Inner-Voxel Aggregation

Figure 8.2: Illustartion of the FusionNet module. (a) Neighborhood aggregation, including i) point-level feature aggregation by the proposed Voxel-MLP and the ii) voxel-level feature aggregation with sparse convolutions. (b) Inner-voxel aggregation of each voxel in a feedback fashion. Features of all the points in the "mini-PointNet" are fused into voxel-level features by average-pooling and concatenation. The voxel-level features are then fed back to each point (with concatenation and refinement).

follows. Sec. 8.3.1 describes the unique point cloud representation. Sec. 8.3.2 and 8.3.3 present the design of the FusionNet module (Fig. 8.2). Sec. 8.3.4 shows the up- and down-sampling layers. Finally, Sec. 8.3.5 describes the architecture and its sparse implementation.

8.3.1 Point Cloud Representation

Our point cloud representation is based on a unique voxel-based "mini-PointNet" that has two steps: voxelization and "mini-PointNet" construction.

Voxelization: Given point cloud P in a range of $L_x \times L_y \times L_z$ as input, we transfer the irregular points into regular 3D voxels with a resolution of $H \times W \times D$. The resolution is controlled by the voxel parameter $s = (s_x, s_y, s_z)$ (length/width/height of the each voxel). Here, $(H, W, D) = (L_x/s_x, L_y/s_y, L_z/s_z)$.

Mini-PointNet: Each point p can be represented as $p = \{(p_x, p_y, p_z), F_p\}$ with (p_x, p_y, p_z) as its 3D space location and F_p as its point features (*e.g.* color, intensity). Since many voxels may have more than one point, voxel V with m points can be represented as a "mini-PointNet": $V = \{(V_x, V_y, V_z), F_V, \{p_1, p_2, ...p_m\}\}$, where (V_x, V_y, V_z) is the coordinate of the voxel that contains m points $\{p_1, p_2...p_m\}$, and F_V is the voxel features that can be learned from those points in each voxel.

The voxel-based "mini-PointNets" are stored in sparse data structures to reduce their memory requirement. Empty or invalid voxels are not be stored or processed during feature aggregations.

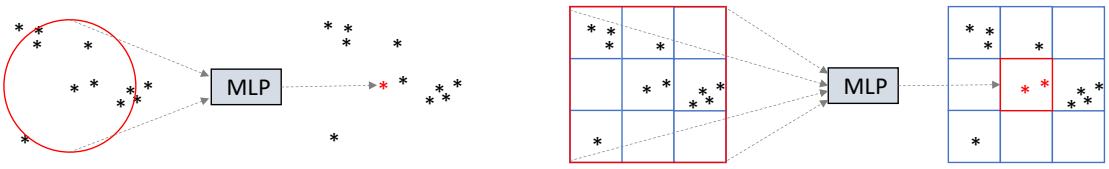
After transferring the point cloud into the voxel-based "mini-PointNet" representation, our FusionNets use the proposed feature aggregation modules to learn deep feature representation for segmentation.

8.3.2 Neighborhood Aggregations

As illustrated in Fig. 8.2(a), there are two types of feature aggregations can propagate information from neighborhood voxels to the current voxel: voxel feature aggregation and point feature aggregation.

Voxel Feature Aggregation: We utilize regular convolutions for voxel feature aggregation, as the convolutional layer has been proven to be successful in 2D-image semantic segmentation. The knowledge can be easily transferred to the design and training of deep network models for 3D point clouds. Regular voxel-based convolutions are also more efficient than many existing point-wise convolution models [326, 155, 312, 299, 205, 92, 183]. These models requires extra memory and computations to learn or interpolate the weight kernels of the point convolutions, as well as to locate the neighborhood points from irregular point cloud by KNN search or ball querying.

To reduce the memory footprint and to develop deep neural networks for



(a) MLP in PointNet (b) Our Voxel-MLP

Figure 8.3: (a) The original MLP module widely used in PointNets [219, 221] and point-wise convolutions [312, 104]. It relies on neighborhood search/ball querying for each pixel with high time complexity ($O(n^2)$ or $O(n\log(n))$ with k-d tree). (b) Our voxel-level MLP (*e.g.* kernel size = 3) directly aggregates features from all the points in the neighborhood voxels to points in the target voxel, allowing neighborhoods to be easily visited in regular voxels. The Voxel-MLP has a lower $O(n)$ time complexity.

the segmentation of large-scale point clouds, we use the Sparse Submanifold Convolution layer [80, 56] for voxel-based aggregation to compute the convolution only at activated voxels (as illustrated in step ii) of Fig. 8.2(a)). This approach helps minimize the memory needed.

Point Feature Aggregation: Fig. 8.3(b) presents a more efficient voxel-based multi-layer perceptron (Voxel-MLP) for neighborhood feature aggregation.

For the m points $\{p_1, p_2, ...p_m\}$ in the current voxel V, their point feature can be aggregated from all the points in the neighborhood voxels $N(V)$ (*e.g.* 27 neighborhood voxels if the kernel size is 3) by our Voxel-MLP as follows:

$$\text{mlp}(p_1, p_2, ...p_m) = \gamma\left(\max_{p_k \in N(V)} \{h(p_k)\}, \{p_1, p_2, ...p_m\}\right). \tag{8.1}$$

The implementation details are presented in Algorithm 1. The response of h can be interpreted as the spatial encoding [219] of a point. The feature aggregation γ can be realized by concatenations and fully connected layers.

Our Voxel-MLP (Fig. 8.3(b)) is similar to the original MLP module (Fig. 8.3(a)) widely used in PointNets [219, 221] and point-wise convolutions [312, 104] for segmentation. However, the ball querying, sampling, or KNN neighborhood search in these existing models need $O(n^2)$ time complexity for points selection in

Algorithm 1: Implementation of the Voxel-MLP

for *each voxel* V*:* **do**
  for *all points* p *(*$p \in N(V)$*) in neighboor voxels* $N(V)$ *of* V **do**
    Spatial encoding:
      i) spatial coordinate shifts $\Delta p \leftarrow (p_x, p_y, p_z) - (V_x, V_y, V_z)$;
      ii) concatenation and FC layer: $\mathbf{F'}_p \leftarrow \text{fc}(\text{cat}\{\mathbf{F}_p, \Delta p\})$;
    Point feature max-pooling: $\mathbf{F}_{max} \leftarrow \max\{\mathbf{F'}_p, p \in N(V)\}$;
  end
  for *each point* p *in current voxel* V **do**
    Concatenate $\mathbf{F'}_p$ and $\mathbf{F}_{max}$: $\mathbf{F''}_p \leftarrow \text{cat}(\mathbf{F'}_p, \mathbf{F}_{max})$;
    Refinement with point-wise fully-connected layer: $\mathbf{F}_p^A \leftarrow \text{fc}(\mathbf{F''}_p)$;
  end
end
Result: Aggregated point feature $\mathbf{F}_p^A$ ($p \in V$)

irregular point clouds. Although k-d trees can be used to accelerate the search with $O(n \log(n))$ complexity, building a k-d tree for each layer will also cost extra memory and computations. In contrast, in our Voxel-MLP, all the neighborhood points can be directly visited more efficiently from neighborhood voxels (by hash mapping with $O(1)$ time complexity for each point–in total $O(n)$).

The receptive field of the Voxel-MLP is controlled by the kernel size that is the same as our convolutional aggregation. For example, when setting the kernel size to 3, neighborhood points will be selected from 27 neighborhood voxels.

8.3.3 Inner-voxel Aggregation

Voxel-level aggregations only learn coarse-grain voxel features that usually produce ambiguous/wrong predictions at the object borders when voxels consist of points from different classes. To address this issue, we design the inner-voxel aggregation for fine-grain point-level feature aggregations and label predictions.

Fig. 8.2(b) illustrates the inner-voxel aggregation realized by a feedback design of the feature fusion. The point-wise feature F_p and voxel feature F_V are from the outputs of the neighborhood aggregation block (Fig. 8.2(a)). They are sent to

the inner-voxel block for fusion and aggregation. We first use point-wise average pooling to achieve the pooled feature vector from m points in the "mini-PointNet." The feature vector is then fused into the voxel feature F_V by concatenation. After refinement by one 1×1 convolution/FC layer, the aggregated voxel-level features F_V^A are then fed back to each point in the "mini-PointNet" by concatenations. Finally, a point-wise fully connected layer is employed to get the aggregated point feature F_p^A.

Namely, for each valid voxel V with m points ($\{p_1, p_k ... p_m\} \in V$) in the "mini-PointNet", the inner-voxel aggregation can be represented as:

$$\begin{aligned} F_V^A &= \gamma_1 \left(\underset{p \in V}{\text{avg}} \left\{ h(F_p, p) \right\}, F_V \right) \\ F_p^A &= \gamma_2 \left(h(F_p, p), F_V^A \right), \ p \in V. \end{aligned} \tag{8.2}$$

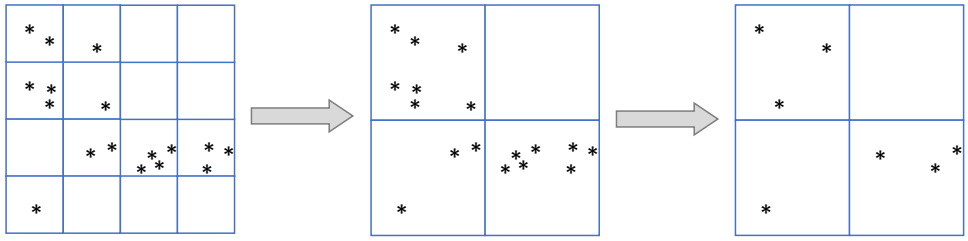
Where the response of h can be interpreted as the spatial encoding of the point p. It can be learned by fully-connected layers with the point feature F_p and its spatial shifts to the voxel center $(p_x - V_x, p_y - V_y, p_z - V_z)$ as inputs (which is similar to [219] and the Voxel-MLP). γ_1 and γ_2 are the feature fusion functions which are realized by concatenations and fully-connected layer.

After the circulation of the inner-voxel aggregation, the voxel-level features and the point features are deeply fused. They are then sent to the next FusionNet module/layer for further aggregations.

8.3.4 Down/Up-sampling

Down- and up-sampling are essential in neural networks for segmentation since they help capture pyramidal features in different resolutions and receptive fields.

For regular voxels, we use convolutions and transposed convolutions with strides (*e.g.* 2) for down- and up-sampling. These are similar to existing image segmentation models (*e.g.* UNet [237]) that have been found to be effective.



(a) input voxels (b) voxel down-sample (c) point down-sample

Figure 8.4: Illustration of feature down-sampling. (a) Voxels and "mini-PointNets" inputs. (b) After down-sampling with stirde of 2, the size of the voxels is doubled to get a lower resolution. (c) The number of points in each voxel is also reduced to realize the "mini-PointNet" down-sampling.

Fig. 8.4 illustrates our point feature down-sampling that strictly follows voxel-level sampling. After voxel-level down-sampling, the size of a voxel is expanded to get a lower-resolution representation. To realize point-level down-sampling, we reduce the number of points in each voxel by re-sampling to get the new "mini-PointNet". For example, we reduce the number of points of each voxel to half while ensuring at least one valid point to avoid new empty voxels (Fig.8.4(c)).

When compared to the slow iterative farthest-sampling strategy [221] (which sometimes takes 1–2 seconds for large-scale point cloud), the point sampling in FusionNet is realized in each voxel independently and can be computed in parallel by GPU in super fast speed (10–100 times faster than the farthest-sampling strategy). Moreover, voxel-based sampling guarantees each valid voxel to have at least one point and avoids the appearance of new empty voxels. This is especially important for sparse point cloud or regions (such as LiDAR point cloud where points are very sparse in the distance). Our voxel-based down-sampling can keep the consistency between point-level and voxel-level networks that cannot be realized by random sampling and the farthest-sampling strategy.

For point up-sampling, we use point interpolations proposed in PointNet [219] to recover the high-resolution point representations.

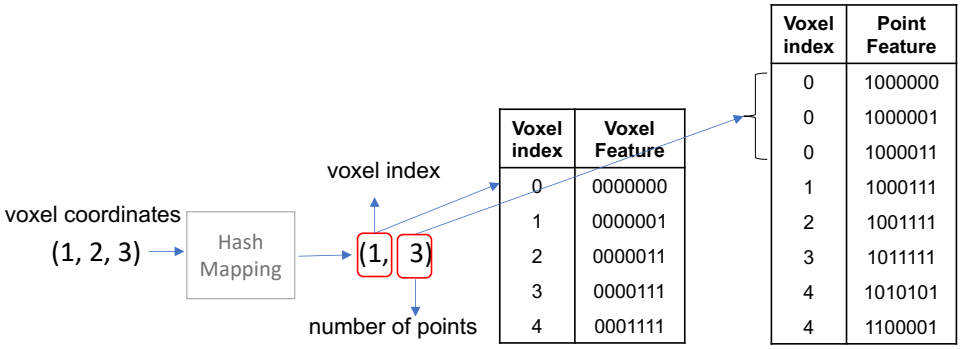


Figure 8.5: Illustration of the sparse data structure and the voxel/point visiting by hash mapping.

8.3.5 Architecture and Sparse Implementation

Architecture: *We use the 3D UNet backbone which is the same as that in [80, 56]. At each pyramid level, one FusionNet module is employed to replace the convolutional layer. Both point-level and voxel-level features are densely connected to the previous layers by concatenations.

Sparse Implementation: Fig. 8.5 illustrates the storage of the points and voxels in a sparse data structure. Each voxel and its points can be visited quickly by hash mapping. The coordinates (including the batch ID) of the voxels are used as the keys to the hash map, and a querying coordinate directly points to the location of each voxel and the "mini-PointNet".

When compared to dense regular convolution nets [354, 276], FusionNet takes full advantage of sparsity to minimize its memory requirement. It differs from PointNets [219, 221] and point-wise convolutional nets [104, 312] that need high computational complexity for neighborhood search. As FusionNet utilizes a regular-voxel representation and hash mapping to realize the $O(1)$ point/voxel indexing, it is more suitable for large-scale point cloud segmentation tasks.

*Architecture details are in the supplementary materials: `www.feihuzhang.com`

8.4 Experiments

In our experiments, we train our model with conventional cross-entropy loss for 40k iterations on one GPU.We use Momentum SGD with the Poly scheduler to train networks from learning rate 1e-1 and apply data augmentation including rotation around the gravity axis, scaling, spatial translation and spatial elastic distortion. For evaluations, we use the standard mean Intersection over Union (mIoU) and mean Accuracy (mAcc) as metrics following the previous works.

8.4.1 Datasets

We use three point cloud datasets for evaluations. Two of them are collected from the indoor scenes, and the rest one is from the large-scale driving scenes.

S3DIS: Stanford 3D Indoor Spaces (S3DIS) dataset [7] contains scans of six floors of three buildings. We use the Fold #1 split following many previous works.

ScanNet: The ScanNet [60] 3D semantic segmentation benchmark consists of 3D reconstructions of real rooms which has 1.5k rooms and 20 classes for semantic segmentation. Each point cloud scan contains 7–550k points.

Semantic KITTI: The Semantic KITTI dataset is a new large-scale LiDAR point cloud dataset in driving scenes. It has 22 sequences with 19 valid classes, and each scan contains 10–13k points in a large space of $160m \times 160m \times 20m$. We use Sequences 0–7 and 9–10 as the training set (in total 19k frames), Sequence 8 (4k frames) as the validation set, and the remaining 11 sequences (20k frames) as the test set. Different from other point cloud datasets, LiDAR points are distributed irregularly in a large 3D space. There are many small objects with only a few points and the points are very sparse in the distance. All these make it challenging for semantic segmentation of the large-scale LiDAR point clouds.

Table 8.1: Ablation study on large-scale Semantic KITTI dataset.

Neighborhood-Voxel Agg		Inner-voxel Agg	Mean Acc (%)	Mean IoU
Voxel Agg	Point Agg			
√			89.6	54.8
	√	√	86.2	41.6
√	√		90.7	58.9
√	√	√	91.8	63.7

8.4.2 Ablation Study

We test the performance of the models with different feature aggregation settings: 1) only using the regular convolutional aggregation like that in [80, 56]; 2) with both the convolutional voxel-feature aggregation and the neighborhood-voxel point feature aggregation; and 3) with all three types of feature aggregations (our FusionNet).

Table 8.1 shows that only with the regular voxel-based convolutional aggregation, it will become a sparse convolutional nets [80, 56] and achieves 54.8% in the mean IoU evaluation. By introducing the neighborhood point feature aggregation, the results can be improved by 4%. On the contrary, the FusionNet will degenerate to pointnets [221] if we only use the point feature aggregation. Finally, the full settings of the FusionNet get the best mean IoU of 63.7%.

8.4.3 Benchmark Evaluations

Semantic KITTI: We use a voxel size of 5cm for building our voxel-based "mini-PointNet" representation. Each valid "mini-PointNet" contains 1–60 points. The model is trained on the training set, and the results are submitted to the online benchmark for evaluation using the 20k test set.

The results are presented in Table 8.2 and visualized in Fig. 8.6 for comparisons. Our FusionNet can achieve a new state-of-the-art performance in both the mean IoU and the accuracy evaluations. It outperforms the PointNet++ [221] (by 40%), state-of-the-art point-wise convolutions [312] (by 10%) and the sparse convolution

Table 8.2: Single scan results (19 classes) on Semantic KITTI benchmarks. All models are trained on the training set and evaluated on the online benchmark.

Approach	**mIoU**	**mAcc**	road	sidewalk	parking	other-ground	building	car	truck	bicycle	motorcycle	other-vehicle	vegetation	trunk	terrain	person	bicyclist	motorcyclist	fence	pole	traffic sign
PointNet [219]	14.6	-	61.6	35.7	15.8	1.4	41.4	46.3	0.1	1.3	0.3	0.8	31.0	4.6	17.6	0.2	0.2	0.0	12.9	2.4	3.7
SPGraph [145]	17.4	-	45.0	28.5	0.6	0.6	64.3	49.3	0.1	0.2	0.2	0.8	48.9	27.2	24.6	0.3	2.7	0.1	20.8	15.9	0.8
SPLATNet [263]	18.4	-	64.6	39.1	0.4	0.0	58.3	58.2	0.0	0.0	0.0	0.0	71.1	9.9	19.3	0.0	0.0	0.0	23.1	5.6	0.0
PointNet++ [221]	20.1	-	72.0	41.8	18.7	5.6	62.3	53.7	0.9	1.9	0.2	0.2	46.5	13.8	30.0	0.9	1.0	0.0	16.9	6.0	8.9
SqzeSegV2[311]	39.7	-	88.6	67.6	45.8	17.7	73.7	81.8	13.4	18.5	17.9	14.0	71.8	35.8	60.2	20.1	25.1	3.9	41.1	20.2	36.3
TanConv[272]	40.9	-	83.9	63.9	33.4	15.4	83.4	90.8	15.2	2.7	16.5	12.1	79.5	49.3	58.1	23.0	28.4	8.1	49.0	35.8	28.5
PointASNL[329]	46.8	-	87.4	74.3	24.3	1.8	83.1	87.9	39.0	0.0	25.1	29.2	84.1	52.2	**70.6**	34.2	**57.6**	0.0	43.9	57.8	36.9
DarkNet53[60]	49.9	87,8	**91.8**	74.6	64.8	27.9	84.1	86.4	25.5	24.5	32.7	22.6	78.3	50.1	64.0	36.2	33.6	4.7	55.0	38.9	52.2
RandLANet[101]	50.3	85.9	88.0	67.9	56.9	15.5	81.1	94.0	**42.7**	19.8	21.4	**38.7**	78.3	60.3	59.0	47.5	48.8	4.6	49.7	44.2	38.1
PointConv[312]	51.2	87.1	88.9	68.4	58.9	19.7	84.6	93.1	37.8	20.7	22.9	38.1	79.9	62.8	60.7	46.2	39.1	5.5	52.0	48.1	44.7
MinkNet42[56]	54.3	89.5	91.1	69.7	63.8	29.3	**92.7**	94.3	26.1	23.1	26.2	36.7	83.7	68.4	64.7	43.1	36.4	7.9	57.1	57.3	60.1
FusionNet	**61.3**	**91.2**	**91.8**	**77.1**	**68.8**	**30.8**	92.5	**95.3**	41.8	**47.5**	**37.7**	34.5	**84.5**	**69.8**	68.5	**59.5**	56.8	**11.9**	**69.4**	**60.4**	**66.5**

Table 8.3: Stanford Area 5 Test (Fold #1) (S3DIS) [7]

Methods	mIoU	mAcc
PointNet [219]	41.09	48.98
SparseUNet [79]	41.72	64.62
PVConv [176]	52.3	–
TangentConv [272]	52.80	60.70
3D RNN [346]	53.40	71.30
PointCNN[155]	57.26	63.86
SuperpointGraph[145]	58.04	66.50
MinkNet32 [56]	65.35	71.71
KPConv[276]	67.1	**72.8**
Our FusionNet	**67.2**	72.3

Table 8.4: 3D Semantic Label Benchmark on ScanNet[56].

Methods	mIoU
ScanNet[60]	30.6
PointNet++ [221]	33.9
TangetConv [272]	43.8
SurfaceConv [205]	44.2
MVPNet [119]	64.1
PointConv[312]	66.6
PointASNL[329]	66.6
MinkNet42 (5cm)[56]	67.9
KPConv[276]	68.4
FusionNet (5cm)	**68.8**

[56] (by 7%) in mean IoU evaluations. Moreover, It achieves the best IoUs in 14 out of the 19 classes. For many small objects (*e.g.* person, bicycle, motorcycle), it outperforms other existing models by at least 10–25%.

FusionNet has many advantages for the large-scale LiDAR point cloud segmentation. Compared to state-of-the-art voxel-based networks [80, 56], FusionNet can predict point-wise labels and avoid those ambiguous/wrong predictions at object boundaries when a voxel has points from different classes. When compared to state-of-the-art point-wise convolutions (*e.g.* [299]), our FusionNet gets much better segmentation accuracy in the large-scale LiDAR dataset. This is because our

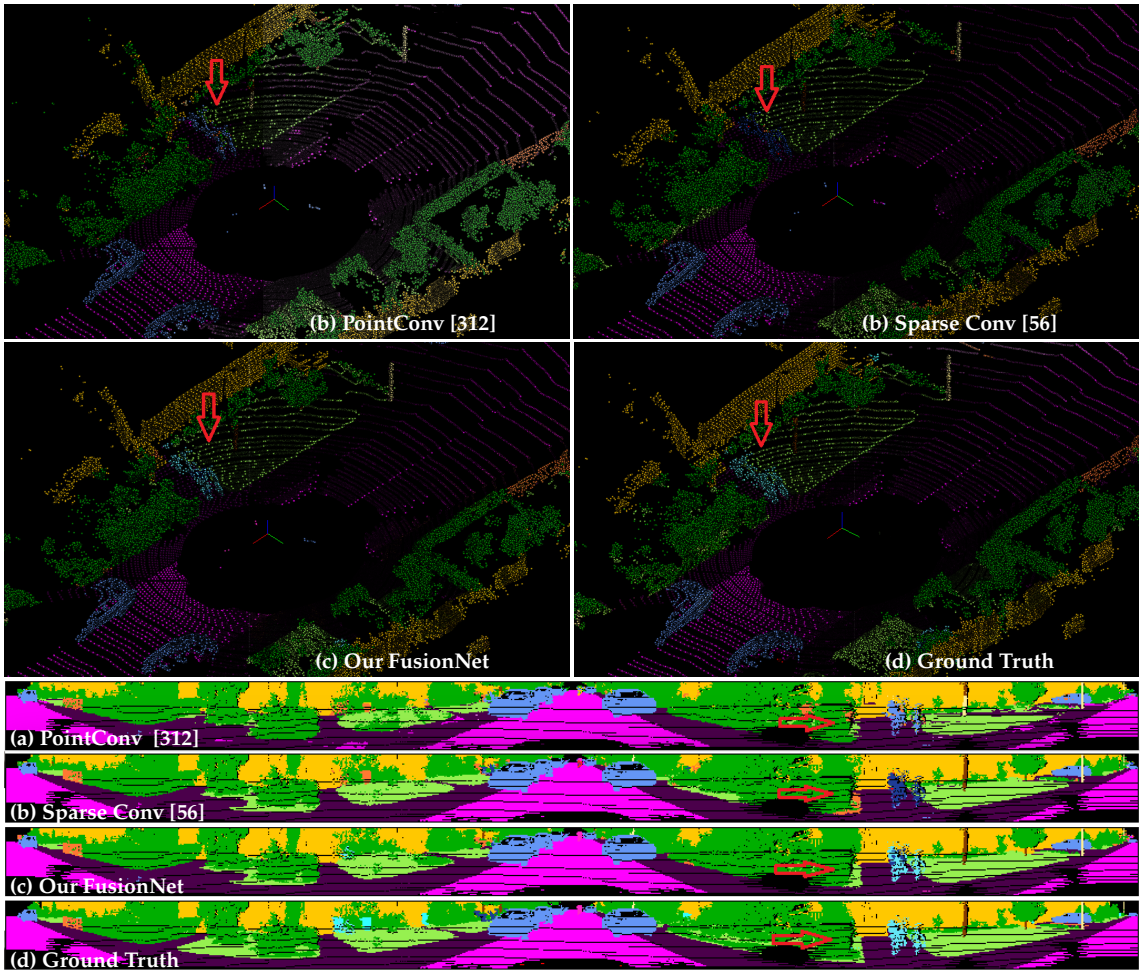
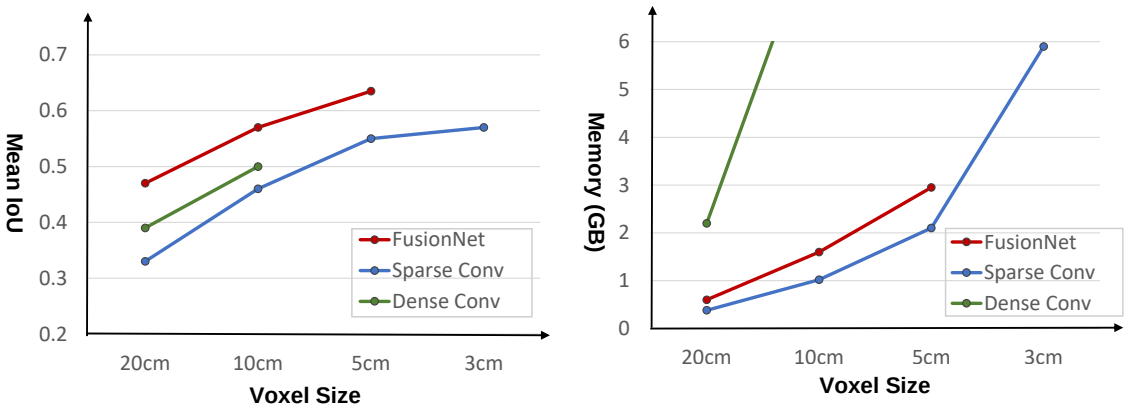


Figure 8.6: Visualization of the results of LiDAR point clouds. Top (a-d) are the raw point cloud results. Bottom (a-d) are the visual comparisons by projecting the point clouds to cylindrical images. (a) State-of-the-art point-wise convolutions [312], (b) state-of-the-art sparse convolutions [56], (c) our FusionNet, (d) ground truths. The differences are as illustrated by red arrows (where bicycles are predicted as other objects by [312, 56]).

FusionNet is realized with more effective feature aggregation operations (including the effective voxel-level neighborhood aggregations and the fine-grain inner-voxel point-level aggregations).

3DSIS and ScanNet: We also evaluate our FusionNet on two indoor-scene datasets (3DSIS [7] and ScanNet [60]). The results are presented in Table 8.3 and Table 8.4. The voxel size is set to 5cm for our voxel-based "mini-PointNet" representation. When compared with the sparse convolutional nets [56] which use the same



(a) IoU Changing Trends (b) Memory Changing Trends

Figure 8.7: Performance illustration with different voxel sizes. (a) The mean IoU changes along with the voxel sizes. PVCNN [176] and sparse convolutions [56]) are used for comparisons. Higher resolution can help get a better mean IoU. (b) The memory consumption changes along with the voxel sizes (training phase using LiDAR frame [17]). The memory consumption significantly increases as the resolution goes up. Our FusionNet with a lower resolution (5cm) can achieve a better mean IoU when compared to the sparse convolutions [56] (with a high resolution of 3cm).

resolution as input, our FusionNet model can learn fine-grain point-wise features and give point-wise predictions. Therefore, it can get better mean IoUs (68.8% on ScanNet benchmark and 67.2% on S3DIS test set). Also, our FusionNet outperforms state-of-the-art point-wise convolutions (*e.g.* PointCNN [155], PointConv [312]) by 2–10% in the mean IoU evaluations. This is because our FusionNet is realized with a more powerful feature aggregation module for learning more effective features in the segmentation.

8.4.4 Analysis of the Voxel-based "Mini-PointNet"

Our FusionNet is realized with the unique voxel-based "mini-PointNet" representation. The voxel size is the key parameter that decides the resolution of our "mini-PointNet" representation and will influence the accuracy of the segmentation results. We evaluate the performance by using different voxel sizes for building the

networks. Similar networks (PVCNN [176] and sparse convolutions networks [56]) are used for comparisons. They use either the dense voxel representation [176] or the sparse voxel representation [56]. Models are tested on the Semantic KITTI validation set. We fixed the number of points to 125k for the memory test (in the training phase).

As shown in Fig. 8.7, higher resolution can help get better accuracy. But, the memory and computational costs significantly increase (almost cubically) which is a big limitation for learning on large-scale point clouds. State-of-the-art voxel-based convolutional nets [80, 56] require an extremely high resolution to achieve state-of-the-art accuracy. By using the same resolution setting, our FusionNet can get far better accuracy. The FusionNet with a lower resolution (5cm) outperforms the high-resolution (3cm) sparse convolutional network [56]. This is benefited from our unique voxel-based "mini-PointNet" representation that can help learn point-wise predictions with more effective feature aggregations.

8.4.5 Efficiency for Large-scale Point Cloud Processing

In this section, we systematically evaluate the efficiency and abilities of our FusionNet on processing real-world large-scale point clouds. The Semantic KITTI dataset [17] is used for evaluations. We compare our FusionNet with the recent work that can also produce point-wise predictions. For fair comparisons, we fixed the number of points (to 120k) for each LiDAR scan during the speed and memory test. In addition, we also measure the maximum number of 3D points each network can take in a single pass to infer per-point semantics. All experiments are conducted on the same device (RTX TITAN GPU).

Due to the high memory and computational costs and the limitation of the architecture design, many state-of-the-art point-wise segmentation models [219, 221, 312, 101] require re-sampling to reduce and fix the number of points in each

Table 8.5: Efficiency comparisons for large-scale point cloud processing.

Approach	Memory consumption(GB)	Elapsed time (per scan)	Max inference points (million)
PointNet [219]	3.1	0.2	0.8
PointNet++ (MSG) [221]	4.2	2.0	0.6
PointCNN [155]	13.4	3.0	0.2
PointConv [312]	3.0	3.0	0.8
Our FusionNet	1.0	0.9	2.3

point cloud frame for training and testing. This need to split the point cloud. But, in LiDAR point clouds, points are very sparse in the distance, down-sampling will further increase the sparsity and lead to worse segmentation accuracy due to the insufficient feature aggregation from neighborhoods.

As a comparison, our FusionNet can directly take the entire point cloud as input for training and testing that is more flexible in real applications. As shown in Table 8.5, it can take up to 2.5 million points in a single pass to infer the point-wise segmentation which is four times as many as that of PointNet++.

8.5 Conclusions and Future Work

In this paper, we propose a deep fusion network architecture (FusionNet) with a new feature aggregation module for large-scale 3D semantic segmentation. The proposed FusionNet utilizes a unique voxel-based "mini-PointNet" for point cloud representation and learning. It can realize both the neighborhood voxel-level feature aggregation and the fine-grain point-wise feature learning. Therefore, FusionNet can produce more accurate point-wise predictions when compared to voxel-based convolutional networks. Also, when compared to popular point-wise convolutions, it realizes more effective feature aggregations with lower memory and computational costs.

Currently, part of our system (the hash mapping) is implemented on the CPU

that limits the running speed. We will try the GPU hash mapping in the future that will significantly accelerate our FusionNet for real-time applications.

Acknowledgement

Research is supported by Baidu, the ERC grant ERC-2012-AdG 321162-HELIOS, EPSRC grant Seebibyte EP/M013774/1 and EPSRC/MURI grant EP/N019474/1. We would also like to acknowledge the Royal Academy of Engineering and FiveAI.

Chapter 9

Instance Segmentation of LiDAR Point Clouds

Feihu Zhang
University of Oxford

Chenye Guan
Baidu Research, Baidu Inc.

Jin Fang
Baidu Research, Baidu Inc.

Song Bai
University of Oxford

Ruigang Yang
Baidu Research, Baidu Inc.

Philip H. S. Torr
University of Oxford

Victor Prisacariu
University of Oxford

Abstract

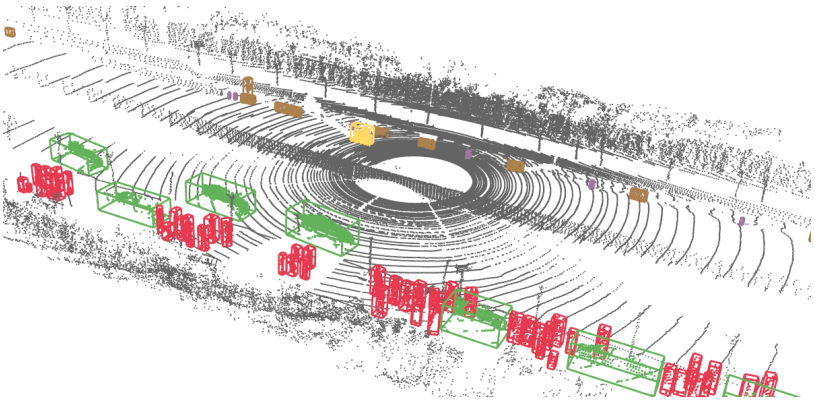
We propose a robust baseline method for instance segmentation which are specially designed for large-scale outdoor LiDAR point clouds. Our method includes a novel dense feature encoding technique, allowing the localization and segmentation of small, far-away objects, a simple but effective solution for single-shot instance prediction and effective strategies for handling severe class imbalances. Since there is no public dataset for the study of LiDAR instance segmentation, we also build a new publicly available LiDAR point cloud dataset to include both precise 3D bounding box and point-wise labels for instance segmentation, while still being about 3∼20 times as large as other existing LiDAR datasets.

9.1 Introduction

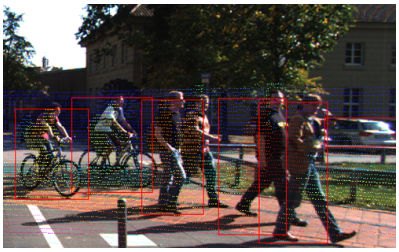
Accurately capturing the location, velocity, type, shape, pose, size *etc*. of objects (*e.g*. cars, pedestrians, cyclists *etc*.) in the outdoor scenes is a vital task for many vision and robotic applications. The LiDAR system can extract 3D information from the surrounding environment with high accuracy in various lighting conditions and has become a key component for *e.g*. autonomous driving and robotic systems.

LiDAR detection has seen much recent work, with the introduction of methods such as [378, 47, 330, 251], which estimate the locations of *e.g*. the cars, as either 3D or 2D bounding boxes. But, for detection, many outliers (*e.g*. points from the road or neighborhoods) will be mixed into the bounding boxes. Moreover, any imperfection (errors of orientation, size, center *etc*.) of the detected bounding boxes will heavily reduce the precision (illustrated in Fig. 9.1(b)).

A far less explored avenue of research is instance segmentation in LiDAR data. Unlike the coarse estimation of object detection, instance segmentation aims



(a) Results of Instance Segmentation



(b) Detection Results

(c) Segmentation Results

Figure 9.1: Problem and performance illustrations. (a) Results of instance segmentation produced by our method. (b) Results of the state-of-the-art detection methods [251] (projection on 2D image). (c) Our method (on 2D image). It can produce more accurate instance segments and help get better sensor fusions by merging image and point cloud segments/masks (zoom in to see sensor fusion results).

to accurately pick up every reflected point from each foreground object. It can significantly reduce the interference of outlier points for each object, better represent the irregular shapes and contribute to more accurate sensor fusions (illustrated in Fig.9.1(c)), motion planning and 3D HD map construction. Moreover, segmentation can better handle the occlusions or neighborhood objects where 3D bounding boxes are hard to estimate and usually ambiguous with large overlaps (as illustrated in Fig.9.1(b-c)).

However, LiDAR instance segmentation is more challenging and different from *2D* image or RGB-D based instance segmentation (*e.g.* [87, 61]). As illustrated in Fig. 9.1(a):

1) Data representation: LiDAR points are sparse and distributed irregularly in a large

3D space. Sparsity also increases considerably with distance;

2) Large-scale scenes: One LiDAR frame can capture more than 120k points in a space of larger than $140m \times 100m \times 5m$;

3) Small objects: There are many small but important objects (*e.g*. pedestrian, bicyclist *etc*.). These objects are relatively tiny with few points reflected and are especially hard to be discovered in large-scale scenes. For example, one pedestrian is less than $0.5m^3$ in the $140m \times 100m \times 5m$ point cloud.

4) Class imbalances: The class imbalance problem is severe in real-world road scenes. For example, the imbalance ratio between the major class (*e.g*. car) and the minor class (*e.g*. pedestrian) can be larger than a factor of 20.

5) Dataset: There is no publicly available dataset with point-wise instance labels.

In this paper, we focus on the multi-class and multi-object instance segmentation of the LiDAR point clouds in large-scale outdoor scenes. We propose a robust baseline model which includes a new dense feature encoding scheme for point cloud representations to achieve better localization and segmentation of the far and small objects, a powerful densely connected stacked backbone and a simple but effective approach for instance prediction.

Moreover, we contribute a large dataset for the study of LiDAR point cloud based instance segmentation. Existing datasets (*e.g*. KITTI [74] and Nuscenes [27]) only label 3D bounding boxes. Our new dataset has both 3D bounding box and point-wise labels, which allows robust instance segmentation models to be trained. It has a total of 130k point cloud frames, with more than 3 millions foreground objects, so is $3 \sim 20\times$ larger than existing LiDAR datasets.

9.2 Related Work

In this section, we review related work on point-cloud-based detection and instance segmentation. We also explore the datasets available for training point-cloud methods.

9.2.1 3D Detection

Several methods [378, 47, 330, 251, 67, 293, 251, 255] propose point-cloud-based detectors, that estimate object locations and produce 2D or 3D bounding boxes. Image information is also leveraged, in approaches like [138, 218, 345, 46, 45, 382, 317]. However, the accuracy of image-based 3D detection approaches is heavily dependent on the quality of the image data, so they would not be reliable in all lighting conditions (*e.g.* in the dark). Multi-sensor fusion approaches [161, 323] have also been proposed to increase the reliability of the detectors.

These detection methods rely on the generation of accurate 3D bounding box proposals. As the variances of objects' sizes and shapes increase, especially for many small objects, it becomes much more difficult to produce accurate bounding box proposals.

9.2.2 3D Instance Segmentation

Instance segmentation is the problem of simultaneous locating and delineating each distinct object of interest appearing in a scene. Based on recent advances in object detection [230, 78, 164, 227, 228, 229, 172, 165], instance segmentation [61, 215, 214] has achieved good results on 2D images. Many of the latest instance segmentation models are based on segment or mask proposals [214, 61, 87]. These methods, however, can not immediately be extended to 3D point clouds due to irregularity and sparsity of the large 3D space and the difficulty of generating proposals.

Recent approaches have shown promising performance for point clouds in smaller indoor environments. Some of them (*e.g.* [300], [167]) learn a similarity or affinity matrix for grouping the points and generating instance-level segments. Volumetric approaches, like 3D-SIS [99], uses both 3D geometry and 2D images as input for anchor based detection and mask prediction. Direct bounding box regression is used in [347], with a shape generation/reconstruction stage.

However, there are currently limited work for large-scale, outdoor point cloud instance segmentation. Existing approaches are designed for indoor scenes, which (i) are considerably smaller, (ii) have denser point clouds and (iii) require a slammer variability of object shape and size. Some also assume associated RGB data and are not applicable to the challenging outdoor scenes or produce poor performance.

9.2.3 Point Cloud Datasets

There are several popular point cloud datasets for semantic segmentation and detection. Most of them provide RGB-D data, filmed by ToF cameras. Examples are (i) [252, 163, 260, 259, 84], built for 2D object detection; (ii) LabelMe [241] and SUN RGB-D [258], providing polygons in 2D and bounding boxes in 3D; (iii) NYU v2 [253] consisting of 464 short sequences with semantic labels; (iv) Armeni *et al*. [7, 6] providing an indoor dataset with 3D meshes and semantic annotations for 265 rooms and (v) ScanNet [60], a RGB-D video dataset containing with 2.5M views in 1513 scenes for 3D semantic and instance labels.

All these datasets target (relatively) small-scale indoor scenes. There are also a few point cloud datasets for large-scale outdoor scenes. Semantic3D [85] is built for large-scale semantic segmentation. KITTI [74], Apollo [10], H3D [211] and Nuscenes [27] are all LiDAR datasets developed for object detection with only 3D bounding boxes labels.

To the best of our knowledge, currently, there is no large-scale, publicly available

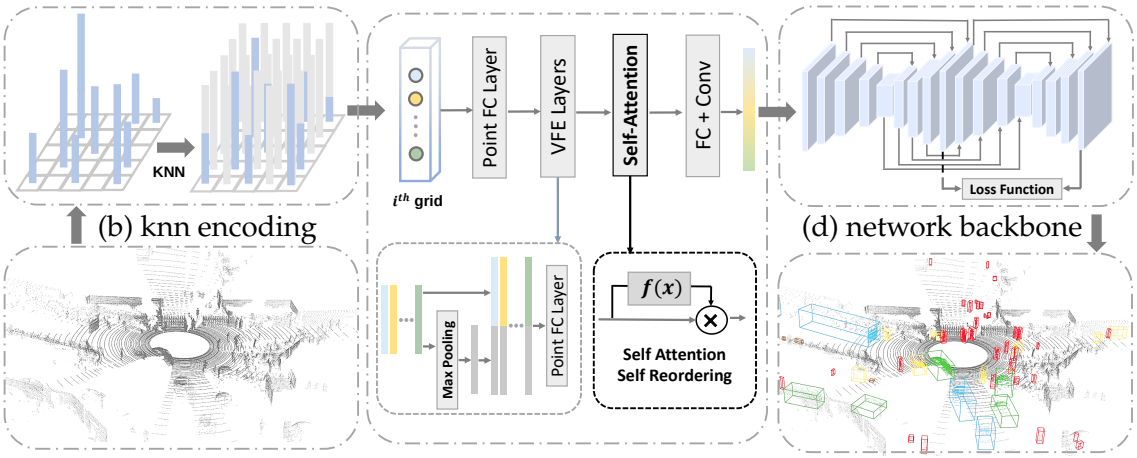


Figure 9.2: Framework of the proposed method. (a) Point cloud frame as input, (b) knn to get dense representation, (c) feature learning with a self-attention block to re-organize the unordered points, (d) backbone network, (e) instance segmentation result with bounding boxes.

LiDAR dataset with both precise point-wise semantic and instance labels. We present in section 9.4 our LiDAR segmentation dataset which has both 3D bounding boxes and point-wise labels.

9.3 Proposed Model

Fig. 9.2 outlines the components of our approach for large-scale instance segmentation of LiDAR point clouds, detailed in the following sections as follows: §9.3.1 for the feature representation, §9.3.2 for the network backbone, §9.3.3 for our final instance prediction and §9.3.4 for our loss.

9.3.1 Dense Feature Representation

The feature representation is the initial but crucial step for point cloud based recognition tasks. In this section, we develop our dense feature encoding technique for the instance segmentation.

1) Challenges for Feature Representation

A popular type of irregular feature representation, used for point-level semantic segmentation, is the approach employed by PointNet [219] and PointNet++ [221], where local and global features are learned for each point. The input points are unordered, so such methods are unstable to use powerful convolutional layers to learn spatial or geometric information (*e.g*. bounding box regression). Then, Li *et al*. propose the transformed CNN for feature extraction [155], which is also for classification and semantic segmentation.

For large-scale LiDAR point clouds, regular representations, like voxels or grids, enable richer geometric information (*e.g*., height, center, distance *etc*.) to be captured efficiently by convolutions. Some approaches then use hand-crafted features [185, 47, 330]. These yield satisfactory results when rich and detailed 3D shape information is available. However, they do not adapt well to sparse scenes and produce poor accuracies for small objects. Other approaches, *e.g*. VoxelNet [378], learn sparse point-wise features for voxels and use 3D convolutions for feature propagation, which shows good results in 3D object detection.

These sparse regular feature representations are popular in object detectors. But, they have limited performance for segmentation models. Instance segmentation requires high-resolution feature maps for finding tiny objects (*e.g*. pedestrians and cyclists *etc*.) in large-scale point clouds. This will lead to large amount of empty/invalid voxels/grids, especially for far locations. The invalid/empty locations will make the convolutional layers unstable and discount the efficacy of feature propagation, localization and regression. This is because traditional (dense) convolutions are not specialized for the sparse data structure and the propagation can be easily stopped or affected by void elements.

2) KNN Encoding as Input

9. Instance Segmentation of LiDAR Point Clouds

It is difficult to extract enough information only from the valid points in each grid/voxel when targeting object-wise inference. We design a novel dense high-resolution bird's-eye view representation for instance segmentation. Our point cloud encoding is based on the fact that every point contains rich information not only for its own grid/location/voxel but also for its surroundings. This is especially meaningful for the locations with few (*e.g.* in the distance) or no points. The distances to valid points and the surrounding points' heights all provide rich geometric information for feature encoding.

For efficiency, we directly encode the point cloud into a 2D high-resolution bird's-eye view representation. The segmentation and localization is then based on this regular representation. Compared with more complicated 3D voxels, our approach is much faster and memory efficient, since no 3D convolutions will be involved.

Our feature encoding and learning strategies are depicted in Fig. 9.2(b)∼(c). All points are projected into high-resolution ($H \times W$) 2D grids. We use KNN to augment empty or poor grid locations with their K near neighborhoods.

The KNN algorithm runs based on the horizontal (x, y) distance and can be efficiently parallelized for computation on the GPU or accelerated using K-D Tree. There are totally K points in each 2D grid. We represent each point as $P_i = \{(g_x, g_y), (\Delta x_i, \Delta y_i), (z_i, r_i)\}$, where (i) (g_x, g_y) are the coordinates of the center of the grid; (ii) $(\Delta x_i, \Delta y_i) = (x_i, y_i) - (g_x, g_y)$ is the shifts between the point coordinates and the grid center; (iii) r_i is the received reflectance and (iv) z_i is the point's height. Finally, the input becomes $\{P_0, P_1, ...P_{K-1}\}$ for each grid location.

3) Self-attention Block

Since the input of K points in each grid are unordered, we cannot use convolutional or fully connected layers to learn grid-level feature directly from these K

points. We instead, utilize a novel self-attention and re-ordering block to learn grid-level feature representation from raw points.

The attention block starts with two voxel feature encoding (VFE) layers, as proposed in [378], to learn point-wise features and attention matrix. For each grid location, we assume the K points are represented as $\mathbf{F_p}$, which is a $K \times n$ matrix (K is the number of the points and n is the length of the point-wise feature). We learn an adaptive weight matrix $\mathbf{A}$, which plays two key roles. Firstly, it helps re-organize the points into a regular and ordered representation by adapting the weights to the points, without being influenced by the order and locations of the points. Secondly, it learns a self-attention mask for the K points, to decide automatically which point should play key role and have more of our attentions.

The grid-level feature $\mathbf{F_g}$ can be learned with:

$$\mathbf{F_g} = \mathbf{A}^T \cdot \mathbf{F}_\mathrm{p} \text{ with } \mathbf{A} = \boldsymbol{f}(\mathbf{F_p}) \tag{9.1}$$

where, $\boldsymbol{f}$ is a mapping function to get the self-attention and self-reordering matrix $\mathbf{A}$. The output is in $K \times n$. Specifically, each (i-th) row of $\mathbf{F_g}$ is learned as:

$$\mathbf{F}_\mathbf{g}^i = a_i^0 \cdot \mathbf{F}_\mathbf{p}^0 ... + a_i^j \cdot \mathbf{F}_\mathbf{p}^j ... + a_i^{k-1} \cdot \mathbf{F}_\mathbf{p}^{k-1} \tag{9.2}$$

where $[\mathbf{F}_\mathbf{p}^0, ...\mathbf{F}_\mathbf{p}^j, ...\mathbf{F}_\mathbf{p}^{k-1}]$ are point-wise features for K points. The weight a_i^j is the j-th row and i-th column in matrix $\mathbf{A}$. Since a_i^j is adaptive and learned from $\mathbf{F}_\mathbf{p}^j$, the sequence of the K points will not influence the output $\mathbf{F}_\mathbf{g}^i$. Moreover, it helps re-weight the contributions from each point and automatically decide which point need more attentions.

After the self-attention block, the grid-level feature can be directly consumed by standard convolutional neural layers. We further use one fully connected layer to refine and reshape the $n \times n$ feature into a $48\times$ feature vector. For the

Table 9.1: Parameters of the backbone network.

No.	Layer Description	Output Tensor
input	feature encoding	H×W×48
1	3×3 conv (bn, relu)	H×W×24
2	3×3 conv (stride 2, bn, relu)	12H×12W×48
3	3×3 conv (bn, relu)	12H×12W×48
4-5	repeat 2-3	14H×14W×64
6-7	repeat 2-3	18H×18W×96
8-9	repeat 2-3	116H×116W×128
10-11	repeat 2-3	132H×132W×256
12	3×3 deconv (stride 2, bn, relu)	116H×116W×128
13	3×3 conv (bn, relu)	116H×116W×128
14-15	repeat 12-13	18H×18W×96
16-17	repeat 12-13	14H×14W×64
18-19	repeat 12-13	12H×12W×48
20-21	repeat 12-13	H×W×24
Output	1×1 conv (no bn or relu)	H×W×$(5+C)$
Loss	Eq. (9.5), loss weight: 0.6	———————-
22-41	repeat 2-21	H×W×24
Output	3×3 conv (no bn or relu)	H×W×$(5+C)$
Loss	Eq. (9.5), loss weight: 1.0	———————-
Concatenate	(1,20), (3,18), (5,16), (7,14), (9,12), (13,28), (15,26) (17,24), (19,22), (21,40), (23,38), (25,36), (27,34), (29,32)	

mapping function $\boldsymbol{f}$, we simply use a point-wise fully connected layer along with a column-normalization to learn the self-attention and self-reordering weight matrix.

Finally, the grid-level features construct a regular dense bird's-eye view representation, which is fed to the backbone network for localization, segmentation and classification.

9.3.2 Network Backbone

We use a revised stacked hourglass block as the backbone for instance segmentation. The architecture is illustrated in Fig. 9.2 and detailed in Table 9.1. The pyramid features from different layers are densely connected by concatenations.

Our approach is single shot, *i.e.* no region proposal network as those in [378, 47, 251, 67, 293, 255] is used. The outputs of our network are of size $H \times W \times (5 + C)$, and are used for (i) the foreground classification, (ii) regression of objects' center for

clustering (with 2 channels), (iii) predicting objects' height limits (both upper and lower constraints for removing the false positives) and (iv) classifying objects into C classes (with the remaining C channels).

9.3.3 Instance Segmentation

Popular methods [87, 99] use anchors to predict bounding boxes and achieve the instance IDs. However, in LiDAR point clouds the sizes and poses of the objects have large diversity (from large trucks to small children), making the anchor-based bounding box regression difficult. Others, like SGPN [300], learn a similarity score for each pair of points and then merge them into groups. However, a LiDAR point cloud frame usually contains more than 100k points, making the learning of the similarity matrix not easily tractable (with a time complexity of N^2). Also, the similarity based clustering is likely to be ambiguous and might miss a lot of small objects for large-scale outdoor scenes.

We use a simple while effective way to predict the instance objects. We predict the horizontal center and the height limits of the object. We use them as constraints to group points into each candidate object and further remove outliers. Namely, for each foreground grid, we predict an object center and the height limits. If the predicted centers of different grids/points are close enough ($< 0.3m$), we merge them into a single object. This is very effective for small objects since points are very close to each other. Also, the learning target is simple and easy to learn.

9.3.4 Loss Function

We use the class-balanced focal loss [165] to find foreground points and for classification:

$$\begin{aligned} \mathrm{L}_f(x,y) &= -\sum_{i=1}^{C} (1-p_i)^{\gamma}\ \log(p_i) \\ p_i &= \begin{cases} \frac{1}{1+\exp(-x_i)}, & \text{if } i = y. \\ \frac{1}{1+\exp(x_i)}, & \text{otherwise.} \end{cases} \end{aligned} \tag{9.3}$$

where x is a $C-$channel output for C binary classification tasks. p_i is the probability after $sigmoid$ function. We use $\gamma = 2$ which is the same as that in [165].

For the instance prediction, the learning target is set to $(\Delta x, \Delta y, h)$, where Δx and Δy are the shifts from the current location to the object's center and h is the upper and lower height limits of the object. The center of object is set as the center of the bounding box. We employ the smooth L_1 loss for regression. It is defined as:

$$\mathrm{L}_s(x) = \begin{cases} 0.5x^2, & \text{if } |x| < 1. \\ |x| - 0.5, & \text{otherwise.} \end{cases} \tag{9.4}$$

The final multi-task loss for classification and regression is:

$$\begin{aligned} \mathrm{L} \ &= \ \sum_{x \in I} \mathrm{L}_f(x_{seg}, y_{seg}) + \sum_{x \in I'} \alpha_1 \cdot \mathrm{L}_f(x_{cls}, y_{cls}) \\ &+ \ \sum_{x \in I'} \alpha_2 \cdot \mathrm{L}_s(x_{ct} - y_{ct}) + \alpha_3 \cdot \mathrm{L}_s(x_h - y_h). \end{aligned} \tag{9.5}$$

where, x_{seg} and y_{seg} are the predictions and labels for categorizing foreground grids, x_{ct} and y_{ct} are the $2-$channel predictions and ground truths of the object centers, x_h and y_h are the predictions and ground truths of the objects' height limits, and x_{cls} and y_{cls} are the predictions and labels for classification. The weights of $(\alpha_1, \alpha_2, \alpha_3)$ for balancing different tasks are set to $(0.1, 20, 0.1)$.

The loss for classification and clustering only takes effects on the foreground regions I'. We use the "one point one vote" strategy to predict the objects' categories.

Table 9.2: Comparisons of different LiDAR point cloud datasets.

Dataset	training data (sim + real)	test data (real)	Object number	classes	labeled region	label types 3D bbox	label types point label
Kitti [74]	7,481	7,518	80,256	4	45°	√	×
Apollo [10]	10k	10k	475k	4	360°	√	×
H3D [211]	27k	-	1.1m	2	360°	√	×
Nuscenes [27]	40k	-	1.4m	23	360°	√	×
Ours	100k+20k	10k	3.2m	7	360°	√	√

This is to reduce the influence of the extreme values of unexpected noises.

9.4 Dataset

A big limitation in developing a robust instance segmentation algorithm for LiDAR data is the lack of a publicly available dataset. We build a new large LiDAR point cloud dataset, which contains both 3D bounding boxes and point-wise labels for instance segmentation.

Table 9.2 shows comparisons between our dataset and the existing LiDAR datasets. Our new dataset contains 130k LiDAR frames with around three millions foreground objects, which is $3 \sim 20$ times larger than existing datasets. Most importantly, existing datasets are designed only for detection and only provide 3D bounding box labels. Our dataset also contains point-wise labels for instance segmentation, this is not available in any of the existing datasets.

Our dataset consists of two parts, real data and simulation data, both are larger than either KITTI or Apollo. It is much more expensive and difficult to label instance segments than 3D bounding boxes, To guarantee the diversity of the dataset, we augment the 30k real dataset with 100k simulation samples. We build our LiDAR simulator as in [70] which takes advantages of the real background and the diversity of hundreds of foreground 3D object models to generate high-quality simulation LiDAR samples. The domain gaps between our simulation and the real data are very small. In our experiments, simulation data can improve the AP by 2∼3.5% in training.

Table 9.3: Evaluations of models with different settings. $AP^{0.50}$ and AP (%) are used for evaluations.

Dense Feature	Backbone hourglass	Backbone dense	Backbone concat	Loss Function focal loss	Classification data resampling	Performance $AP^{0.50}$	Performance AP
						63.1	56.5
√						65.8	59.9
√	√					66.4	60.3
√	√	√				67.1	60.9
√	√	√		√		68.0	61.7
√	√	√		√	√	**68.9**	**62.4**

Table 9.4: Evaluations and comparisons of different models for instance segmentation (AP %).

methods	car	large vehicle	pede-strian	bicy-clist	motor-cyclist	traffic cone	others	average
SGPN [300]	83.5	62.8	33.2	24.0	42.7	34.7	30.9	44.5
VoxelNet* [378]	87.1	70.2	47.2	29.0	45.8	46.5	35.1	51.5
PointRCNN*[251]	88.1	71.2	49.4	32.1	49.2	45.2	37.0	53.2
PointPillar* [146]	87.7	71.9	51.2	33.7	50.1	44.7	37.9	53.9
feature 1 [47]	86.1	69.6	55.5	38.8	50.0	58.3	41.4	57.1
feature 2 [330]	88.0	69.3	54.9	38.9	51.1	56.9	44.5	57.6
feature 3 [378]	**89.2**	72.2	60.2	39.8	54.8	58.7	45.7	60.1
Ours	89.0	**74.9**	**62.4**	**45.1**	**55.0**	**61.8**	**48.2**	**62.4**

9.5 Experiments

In this section, we describe our experiments, including the training settings and strategies, ablation study, comparisons and analysis of the results. *Furthermore, efficiency comparisons and more examples of the results are provided in the supplementary materials*[1]. We use the evaluation metrics of points based APs in the experiments. They are similar to those in the COCO instance segmentation challenges [166].

9.5.1 Training Settings and Strategies

During training, all models are optimized with Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$) [134]. We train with a batch size of 16 on eight GPUs for all models. The learning rate is set to 0.01 for the initial training. We train different models for 32 epoches on simulation data. For fine tuning on the real data, we use a learning rate of 0.002 for another 64 epoches. We consider point clouds within the range of

$[-3.5, 1.5] \times [-40, 40] \times [-60, 60]$ meters along Z, Y, X axis respectively. We set the width and length of the grid to 0.125m, which leads to a resolution of 640×960 for the bird's-eye view representation. We set $K = 64$ for KNN searching, and in the self-attention block, n is set to 16.

There are significant class imbalances in the dataset. Along with the balanced weights for loss function, we also use data resampling to augment the minor classes (especially for bicyclist, traffic cone and motorcyclist). Our resampling strategy tries to increase the frequency of appearance for minor classes during training.

We categorize the seven classes of foreground objects into three types, according to the imbalance ratio. They are the major class $C_{major} = \{car\}$, minor classes $C_{minor} = \{$*bicyclist, motorcyclist, traffic cone*$\}$ and medium classes $C_{medium} = \{$*others, pedestrian, large vehicle*$\}$. Let the imbalance ratios (ratio between object numbers of two different classes) of the whole dataset be $r_1 = C_{major} : C_{minor}$ and $r_2 = C_{major} : C_{medium}$. Each point cloud sample is given a base value $f_i = 1.0$ as the appearance frequency. We then modify the frequency of each point cloud frame according to the imbalance ratio of this sample. If it is smaller than r_1, we double it with $f_i = 2f_i$; if it is smaller than r_2, we increase it by $f_i = 1.5f_i$. If the imbalance ratio is even larger than r_1 or r_2, we reduce it by $f_i = f_i/2.0$ and $f_i = f_i/1.5$ respectively.

We employ random horizontal flipping and rotation around $Z-$axis for data augmentation. The angles for rotations are among $[-30°, 30°]$. Extra augmentation tricks, such as those in [378], do not improve accuracy since the dataset is large and diverse enough to avoid overfitting.

9.5.2 Ablation Study

We conduct experiments with several settings to evaluate the effects of different architectures, loss functions and features *etc*. As listed in Table 9.3, our dense feature encoding improves the AP by 3.4%, which plays a key role in the performance

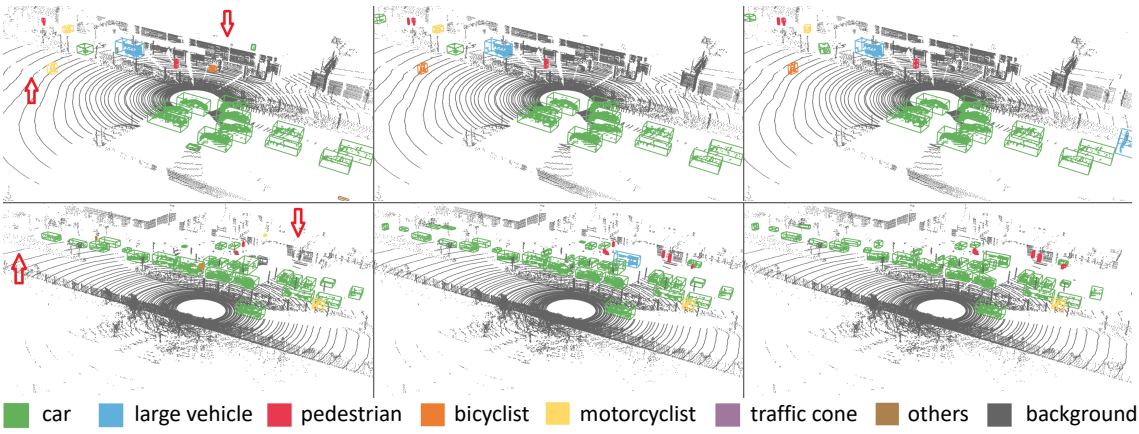


Figure 9.3: Visual comparisons between SGPN [300] and our model in complex and challenging scenes. ***Left:*** results of SGPN [300]. ***Middle:*** results of our model. ***Right:*** ground truths. For visual pleasure, bounding boxes are added which is estimated directly from the instance segments. As pointed by ***red arrows***, many small objects are missed in SGPN. It also picks up many false positives.

improvements. The densely connected stacked backbone contributes another 1.0% improvements. Focal loss and data re-sampling can produce better classification accuracy which help achieve $1 \sim 2\%$ improvements in APs.

9.5.3 Results and Comparisons

Since there is no existing instance segmentation method for LiDAR point as the baseline. We compare our method with the recent proposed SGPN [300] which is designed for small-scale indoor scenes. As shown in Table 9.4 and Fig. 9.3, SGPN is not robust for road scenes when points are very sparse in the distance and there are many small objects with only a few points. As a comparison, our method far outperforms the SGPN (by 17% in AP). It can pick up more small objects and avoid most of the false positives.

We also compare our proposed dense feature encoding with existing sparse feature representations [47, 330, 378] by implementing them into our models (other settings are all the same). As shown in Table 9.4, we find that for large objects (*e.g.* car), existing features also produce good performances for instance segmentation.

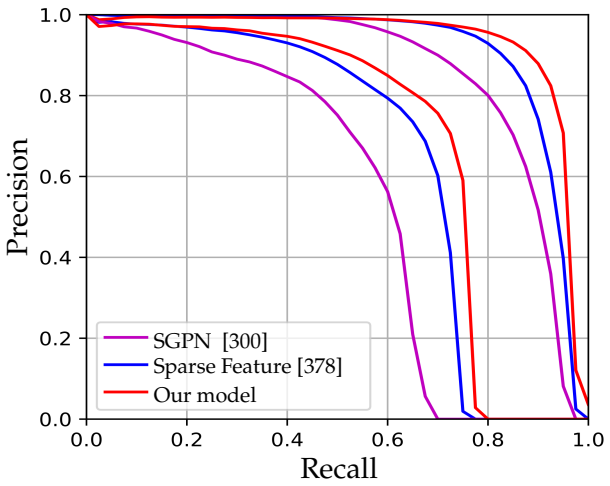


Figure 9.4: Precision-Recall curves with two IoU thresholds of 0.5 and 0.95. SGPN, our method and our model with sparse feature [378] are compared in this figure.

Table 9.5: Evaluations and comparisons in different ranges (%).

Range	0-30m			30-60m		
Methods	$AP^{.50}$	$AP^{.75}$	AP	$AP^{.50}$	$AP^{.75}$	AP
SGPN [300]	50.2	48.1	52.1	48.3	41.8	39.1
Feature 1 [47]	65.9	62.4	60.9	62.0	55.5	53.9
Feature 2 [330]	66.9	63.1	62.3	62.6	55.8	54.1
Feature 3 [378]	68.7	65.9	64.1	64.2	59.4	56.4
Ours	**70.1**	**67.6**	**65.7**	**67.4**	**63.1**	**59.5**

This is because large objects have rich information and are not sensitive to the feature encoding techniques. But, our dense features produce far better recognition accuracy (with $2 \sim 5\%$ improvements in AP) for small objects (pedestrian, bicyclist and traffic cone). This means that the proposed feature encoding can capture more reliable information from the original point cloud for small objects with fewer points. As shown in Table 9.5, for objects within $0 \sim 30m$ from the LiDAR's center, our dense feature outperforms the VoxelNet's feature by about 1%. While, for objects in the range of $30 \sim 60m$, where points are much more sparse, our dense feature has $3 \sim 3.5\%$ improvements in APs.

9.5.4 Instance Segmentation vs. Detection

In this section, state-of-the-art detection methods [378, 251, 146] are compared. Points in the detected bounding boxes are counted as segments for evaluations. As shown in Table 9.4, our instance segmentation model outperforms the state-of-the-art 3D detectors by $8 \sim 10\%$ in AP evaluations. This is because these detectors rely on the generation of accurate 3D bounding box proposals. As both the range and the variance of objects' sizes and shapes increase, especially for many small objects (*e.g.* pedestrian, cyclists *etc.*), the large diversity of the possible bounding boxes makes it difficult to produce accurate estimations for each object. Moreover, many false positives are mixed into the bounding boxes. (Even the 100% accurate ground truth bounding boxes have 12-16% outliers). Therefore, detectors usually produce lower APs for small objects which have fewer points. Any imperfection (errors of orientation, size, center, height *etc.*) of the detected bounding boxes will heavily reduce the precision (as illustrated in Fig. 9.1(b-c)).

9.5.5 Application in Autonomous Driving

For application in self-driving car, picking up as many as obstacles to avoid possible collisions is the most important thing. Here, we evaluate the models' abilities for accurately picking up obstacles. All kinds of objects are counted as a single class, namely, we calculate the precision and recall without the influence of the classification. We plot the AP curves with IoU thresholds of 0.5 and 0.95 in Fig. 9.4, which shows that our method can achieve a higher recall with better precision. It achieves a $\sim$90% precision at a recall of 90% in $AP^{0.5}$, which is the best among all these models.

9.6 Conclusion

In this paper, we propose a robust baseline instance segmentation solution for large-scale LiDAR point cloud. The proposed model extracts more reliable dense features for discovering far and small objects which have only a few points. We also contribute a large LiDAR point cloud dataset. The new dataset has both bounding box and point-wise labels for instance segmentation and is $3 \sim 20$ times as large as the existing LiDAR dataset.

Chapter 10

Conclusion

This integrated thesis focuses on dense prediction tasks, including dense correspondence estimation (stereo matching and optical flow) and 2D/3D semantic segmentation. We develop deep neural network models that demonstrated the improved accuracy on dense prediction benchmarks, the reduced dependency on a large amount of labeled data for training, and the boosted cross-domain generalization abilities to novel data/scenes. We implement the traditional 3D geometry constraints into the end-to-end stereo matching networks and achieve state-of-the-art accuracy on two stereo matching benchmarks (by publication date). This work (Chapter 3) has become a standard baseline in stereo-matching research and has also been extended to many scene flow networks. We also propose the first domain-invariant stereo matching network (Chapter 4) that is trained on the synthetic data but outperforms many models fine-tuned on real data. Furthermore, we develop the Separable Flow network (Chapter 5), which achieves the leading positions on two optical flow benchmarks [191, 25] (by the time of publication). It's also one of the best models when predicting optical flow on new, unseen datasets (by the time of the thesis submission). Moreover, we also study the unsupervised pre-training and domain adaptation for semantic segmentation (Chapter 7 and 6)

and further introduce the 2D image segmentation knowledge to the 3D semantic segmentation. The proposed FusionNet (Chapter 8) achieves the leading position on Semantic KITTI point-cloud segmentation benchmark [17] (by the time of publication).

The rest of this chapter summarises the contributions from each previous chapter and discusses the remaining challenges and future research directions.

10.1 Discussion of Contributions

Chapter 3: GA-Net: Guided Aggregation Net for End-to-end Stereo Matching. Stereo reconstruction is mainly decomposed into three important steps: feature extraction, matching cost aggregation, and disparity prediction. Feature-based matching is often ambiguous, with wrong matches having a lower cost than the correct ones due to occlusions, smoothness, reflections, noise, etc. Therefore, matching cost aggregation is crucial for obtaining accurate disparity estimations in challenging regions. We develop a new end-to-end stereo matching network with much more effective guided matching cost aggregation (GA) modules, including the semi-global aggregation (SGA) and the local guided aggregation (LGA) layers. The two GA layers significantly improve the accuracy of the disparity estimation in challenging regions. We demonstrate the performance of the proposed method on standard stereo matching benchmarks and show that the proposed GA layers can replace computationally costly 3D convolutions to get better accuracy. Furthermore, SGA, the differentiable approximation of the semi-global matching [93], is also shown able to improve the cross-domain generalization abilities to unseen datasets by applying whole-image geometric constraints to the matching cost aggregations. This work has become a standard baseline for stereo matching research and has also been used in scene flow estimation.

Chapter 4: Domain-invariant Stereo Matching Networks. State-of-the-art stereo matching networks (both supervised [355, 32, 128] and unsupervised [377, 282]) cannot generalize well to unseen data without fine-tuning or adaptation. This is because there are significant domain differences across different datasets. Thus, stereo matching networks require the target domain datasets with ground-truth annotations for training or fine-tuning. To boost the performance of stereo matching networks in various complex scenarios, we propose domain-invariant stereo matching network (DSMNet) with two end-to-end trainable neural network modules. Among them, the domain normalization module can fully regulate the distribution of learned features to address significant domain shifts. The semi-global filtering (SGF) module can capture more robust non-local structural and geometric features for accurate disparity estimation in cross-domain situations. We train the DSMNet on synthetic data and verify it on four real datasets to demonstrate its superior accuracy when compared to other state-of-the-art models. It beats both the traditional stereo reconstruction methods and the previous stereo matching networks. Finally, we find that by training on the mixture of the real and synthetic data, DSMNet can be directly used in many other new stereo datasets to produce accurate disparity estimation results. We have made the pre-trained checkpoint and the source code publicly available as a contribution to the community.

Chapter 5: Separable Flow: Learning Motion Cost Volumes for Optical Flow Estimation. Optical flow is another important dense prediction task. State-of-the-art optical flow networks usually rely on building cost volume for pixel-wise motion estimation. However, raw cost volumes generally do not encapsulate constraints to resolve ambiguities caused by occlusion, lack of texture, or other issues. Moreover, the cost volume size is exponential in the dimensionality of the search space. The memory and computational requirements for optical flow, with its 2D search space,

grow quadratically with the range of motion. We introduce the Separable Flow, a cost-volume computation module for optical flow estimation that exploits non-local cost aggregation through a separable cost volume representation. The 2D motion space is separated into two 1D pixel-to-pixel matching problems. Therefore, our Separable Flow is able to build two full-range cost volumes and leverage a more global, rather than local, understanding of the scene, as well as prior knowledge to resolve the ambiguities. When evaluating on the standard optical flow benchmarks (KITTI [191] and Sintel [25]), Separable Flow achieves the best accuracy (by the time of publication) among all published optical flow methods on these two benchmarks. Moreover, in the cross-domain testing of training on the synthetic data and testing on real data (*i.e.* KITTI), our results outperform the previous state-of-the-art by a great margin, even outperforming some popular stereo matching networks (*e.g.* FlowNet2 [115] and PWC-Net [268]) which are fine-tuned on the target KITTI scenes. We released a pre-trained model that is trained on a mixture of four synthetic and real optical flow datasets. Researchers can directly use it for accurate pixel-wise motion estimation on their own datasets or videos without fine-tuning or re-training.

Chapter 6: Looking Beyond Single Images for Contrastive Semantic Segmentation Learning. Like other dense prediction tasks, training deep neural networks for semantic image segmentation also requires a large amount of densely annotated images. Labeling high-quality, dense annotations for an adequate number of images is an extremely laborious and costly task [58]. Self-supervised contrastive representation learning [86, 41] has recently been shown promising in boosting the performance of DNNs by leveraging large amounts of unlabeled data without costly manual annotations. However, few works have studied contrastive learning in dense prediction tasks, such as semantic segmentation. We focus on taking auxiliary labels as guidance to build cross-image correspondences for contrastive

learning in semantic segmentation. Our method can fully utilize existing feature extractors to generate auxiliary labels. The requirements on the feature extractors are flexible. In the experiments, we demonstrate it is possible to use a broad set of feature extractors to generate our auxiliary labels, ranging from networks trained for image classification to semantic segmentation approaches trained on synthetic data. For all feature extractors, our method consistently outperforms the approaches of pre-training on ImageNet. We also design a new quality metric (Proxy mIoU) for auxiliary label exploration. It serves as the guidance to explore effective auxiliary labels without the need for training and testing the neural networks to compute the real mIoU. We test our contrastive segmentation model on three semantic segmentation datasets and two network architectures. We observed that contrastive learning with auxiliary labels consistently boosts semantic segmentation performance when compared to standard ImageNet pre-training and outperforms existing self-supervised and semi-supervised semantic segmentation approaches.

Chapter 7: Unsupervised Contrastive Domain Adaptation for Semantic Segmentation. Furthermore, we also studied the domain adaptation for semantic image segmentation. Since the unlabeled real images are easy to collect, synthetic labeled data can also be cheaply produced. The proposed method leverages the pixel category information from the synthetic data and the image feature distributions from unlabeled real data for joint training and adaptation. Specifically, we introduce contrastive learning and leverage both in-domain contrastive samples as well as cross-domain contrastive samples to align the feature representation of the source and the target domains. Our framework utilizes a student-teacher architecture that generates pseudo-labels to guide the selection of contrastive pairs. Two types of contrastive pairs are selected to help learn robust class-based feature alignment. We further introduced a label expansion strategy to discover contrastive pairs

from particularly hard classes. The effectiveness of our method is verified on two synthetic datasets and two real datasets. It consistently outperforms state-of-the-art for domain adaptation across synthetic and real datasets, particularly on hard classes that are small with fewer pixels or that potentially undergo significant class-specific domain shifts between the source (synthetic) and target (real) domains.

Chapter 8: Deep FusionNet for Point Cloud Semantic Segmentation. We further extend the 2D segmentation ideas to the 3D point-cloud segmentation task. Similar to the 2D dense prediction task, the point-cloud segmentation aims at assigning each 3D point/location a category label. In the 3D segmentation work, we mainly focus on developing new feature representation methods and feature aggregation neural network modules for irregular 3D point clouds. Previous point cloud segmentation methods mainly rely on transferring irregular points into a voxel-based regular representation. However, when a voxel contains points from different classes, the voxel-based convolutional nets will produce ambiguous or wrong predictions. Other PointNets or point-wise convolutions can take irregular points as input and learn point-wise representation for segmentation. Namely, they learn a feature representation for each pixel to avoid the ambiguity of the voxel representation. However, the high memory and computational costs limit their ability and accuracy for large-scale point cloud processing. We propose a deep fusion network architecture (FusionNet) with a unique voxel-based "mini-PointNet" for point cloud representation and learning. Based on the novel "mini-PointNet" representation, a new feature aggregation module is developed which realizes both the neighborhood voxel-level feature aggregation and the fine-grain point-wise feature learning. Therefore, FusionNet can learn more effective feature aggregations with lower memory and computational costs when compared to popular point-wise convolutions. Also, benefitting from the novel "mini-PointNet" of both

voxel and point-level representation, our FusionNet produces more accurate point-wise predictions when compared to voxel-based convolutional networks. The effectiveness of the proposed FusionNet is verified on three standard point-cloud datasets.

Chapter 9: Instance Segmentation of LiDAR Point Clouds. We also studied instance segmentation for large-scale driving scenes composed of LiDAR point clouds. We focus on the multi-class and multi-object instance segmentation of the LiDAR point clouds in large-scale driving scenes and propose a robust baseline instance segmentation solution. The proposed method extracts more reliable dense features for discovering distant and small objects comprised of only a few points. Moreover, since there was no LiDAR segmentation dataset before this paper's publication, we also developed a large synthetic LiDAR point cloud dataset. The new dataset has both bounding box and point-wise labels for instance segmentation, and is $3 \sim 20$ times as large as the existing LiDAR datasets. We also demonstrate that the dense feature representation has better cross-domain (*e.g.* synthetic to real) generalization abilities. Thus, the method can be trained on synthetic LiDAR point clouds and generalized to real LiDAR scenes. For example, it achieves similar precision and recall for the segmentation of vehicles when compared to the models trained on the real LiDAR dataset.

10.2 Remaining Challenges and Future Work

Multi-task Learning. In this thesis, we develop different deep neural network models to solve various dense prediction tasks. Each model is trained on different datasets under different supervisions. In many real application scenarios (*e.g.* robotics), depth, motion, and segmentation are all indispensable and assist each other for a robust system. Multi-task learning tries to solve multiple learning

tasks simultaneously by exploiting commonalities and differences across tasks. It's especially remarkable to study multi-task learning for dense prediction of depth, motion, and segmentation. However, although different dense prediction tasks share many similarities, it's still challenging to incorporate stereo matching, optical flow, and segmentation into a single neural network. The first challenge comes from the lack of a suitable dataset. Currently, existing datasets only support individual tasks. No data is collected and labeled for simultaneous stereo reconstruction, optical flow motion estimation, and segmentation. Even though it is possible to use different data samples to supervise different tasks, it lacks crucial information to make them benefit each other. The second difficulty lies in the differences of the targets. For instance, dense correspondence estimation of stereo matching and optical heavily relies on multi-view geometry for accurate predictions, while segmentation aims at recognizing different contents within a single image. Designing a unified network and training losses for these three tasks is difficult. Some existing methods learn joint stereo reconstruction and optical flow (scene flow [275, 191]) or simultaneous semantic segmentation and stereo matching [52]. But, none of them focus on the much more complicated problem of the joint scene flow estimation and segmentation. In the future, I will try to develop an efficient deep neural network for unified stereo matching, optical flow, and segmentation.

Data Generation. Despite the vast research efforts on dataset creation, data synthesis, and generation, the lack of large-scale, high-quality, densely annotated data is still a crucial limitation for developing new algorithms for accurate dense prediction estimation. There are some work that synthesize images for optical flow [266] or segmentation [209]. But, there are still big quality gaps and domain differences compared with the manually labeled real data. These generated data still cannot be used to replace the real data in training dense prediction models.

Recently, Neural Radiance Fields (NeRF) [193] have been proposed for photorealistic novel view synthesis. Given many views of the scene, it creates implicit multi-view geometry and learns for view synthesis. Furthermore, it has been extended for scene synthesis [139] with segmentation labels. Since NeRF reconstructs the real world and renders the real-world images, it can resolve the problem of domain difference between the synthetic scenes and real images. In the future, I will focus on using NeRF technologies for 3D scene reconstruction, photorealistic image view synthesis, and label generation for depth, motion, and segmentation learning.

10.3 Concluding Remarks

During the course of this thesis, many new deep neural network models have been proposed that achieve better performances on the evaluated dense prediction benchmarks [191, 25, 17, 60]. However, the three discussed issues are still the most important research challenges for dense prediction tasks. It still needs steady effort and rigorous research to explore different possibilities for a truly safe and reliable intelligent system that can finally resolve these issues completely and perfectly.

Bibliography

[1] "artificial intelligence", oxford english dictionary. *Oxford University Press, https://www.oed.com/viewdictionaryentry/Entry/271625*, November 2022. 1

[2] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 34(11), 2012. 112

[3] I. Alonso, A. Sabater, D. Ferstl, L. Montesano, and A. C. Murillo. Semi-supervised semantic segmentation with pixel-level contrastive learning from a class-wise memory bank. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 109, 131

[4] N. Araslanov and S. Roth. Self-supervised augmentation consistency for adapting semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 28, 130, 141

[5] E. Arazo, D. Ortego, P. Albert, N. E. O'Connor, and K. McGuinness. Pseudo-labeling and confirmation bias in deep semi-supervised learning. In *International Joint Conference on Neural Networks (IJCNN)*, 2020. 109

[6] I. Armeni, S. Sax, A. R. Zamir, and S. Savarese. Joint 2d-3d-semantic data for indoor scene understanding. *arXiv preprint arXiv:1702.01105*, 2017. 31, 176

[7] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese.

3d semantic parsing of large-scale indoor spaces. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 31, 162, 164, 165, 176

[8] Y. M. Asano, C. Rupprecht, and A. Vedaldi. Self-labelling via simultaneous clustering and representation learning. In *International Conference on Learning Representation (ICLR)*, 2020. 109

[9] M. Bai, W. Luo, K. Kundu, and R. Urtasun. Exploiting semantic information and deep matching for optical flow. In *European Conference on Computer Vision (ECCV)*, 2016. 22, 87

[10] Baidu. Apollo dataset, `http://data.apollo.auto/?locale=en-us&lang=en`. 176, 184

[11] C. Bailer, B. Taetz, and D. Stricker. Flow fields: Dense correspondence fields for highly accurate large displacement optical flow estimation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2015. 7, 21, 87, 102

[12] C. Bailer, K. Varanasi, and D. Stricker. Cnn-based patch matching for optical flow with thresholded hinge embedding loss. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 22, 87

[13] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. *International Journal of Computer Vision (IJCV)*, 92(1), 2011. 24

[14] Y. Balaji, S. Sankaranarayanan, and R. Chellappa. Metareg: Towards domain generalization using meta-regularization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. 20, 61, 65

[15] A. Bar-Haim and L. Wolf. Scopeflow: Dynamic scene scoping for optical flow.

In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 22, 87, 102

[16] M. A. Bautista, A. Sanakoyeu, E. Sutter, and B. Ommer. CliqueCNN: Deep unsupervised exemplar learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016. 109

[17] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 29, 31, 150, 152, 166, 167, 192, 199

[18] D. Berthelot, N. Carlini, I. J. Goodfellow, N. Papernot, A. Oliver, and C. Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 130

[19] F. Besse, C. Rother, A. Fitzgibbon, and J. Kautz. Pmbp: Patchmatch belief propagation for correspondence field estimation. *International Journal of Computer Vision (IJCV)*, 110(1), Oct 2014. 7, 40, 42

[20] M. J. Black and P. Anandan. A framework for the robust estimation of optical flow. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 1993. 2, 17, 22, 84, 87

[21] M. Bleyer, C. Rhemann, and C. Rother. Patchmatch stereo-stereo matching with slanted support windows. In *British Machine Vision Conference (BMVC)*, 2011. 42, 78, 79

[22] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 8, 61

[23] T. Brox, C. Bregler, and J. Malik. Large displacement optical flow. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 22, 87

[24] G. Bukchin, E. Schwartz, K. Saenko, O. Shahar, R. Feris, R. Giryes, and L. Karlinsky. Fine-grained angular contrastive learning with coarse labels. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 108

[25] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *European Conference on Computer Vision (ECCV)*, 2012. 24, 81, 85, 95, 191, 194, 199

[26] Y. Cabon, N. Murray, and M. Humenberger. Virtual kitti 2, 2020. 96, 102

[27] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. nuscenes: A multimodal dataset for autonomous driving. *arXiv preprint arXiv:1903.11027*, 2019. 174, 176, 184

[28] M. Caron, P. Bojanowski, A. Joulin, and M. Douze. Deep clustering for unsupervised learning of visual features. In *European Conference on Computer Vision (ECCV)*, 2018. 109

[29] M. Caron, P. Bojanowski, J. Mairal, and A. Joulin. Unsupervised pre-training of image features on non-curated data. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 109

[30] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016. 109, 118

[31] K. Chaitanya, E. Erdil, N. Karani, and E. Konukoglu. Contrastive learning of global and local features for medical image segmentation with limited

annotations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 26, 108, 131

[32] J.-R. Chang and Y.-S. Chen. Pyramid stereo matching network. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 4, 6, 7, 10, 17, 19, 38, 40, 42, 50, 54, 55, 56, 60, 63, 64, 66, 74, 76, 77, 78, 79, 80, 89, 98, 99, 100, 193

[33] W. Chang, H. Wang, W. Peng, and W. Chiu. All about structure: Adapting structural information across domains for boosting semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 27, 129, 137

[34] J. Chen, B. Gao, Z. Lu, J. Xue, C. Wang, and Q. Liao. Scnet: Enhancing few-shot semantic segmentation by self-contrastive background prototypes. *arXiv*, abs/2104.09216, 2021. 131

[35] J. Chen, B.-B. Gao, Z. Lu, J.-H. Xue, C. Wang, and Q. Liao. Scnet: Enhancing few-shot semantic segmentation by self-contrastive background prototypes. *arXiv preprint arXiv:2104.09216*, 2021. 27, 108

[36] L.-C. Chen, R. G. Lopes, B. Cheng, M. D. Collins, E. D. Cubuk, B. Zoph, H. Adam, and J. Shlens. Naive-student: Leveraging semi-supervised learning in video sequences for urban scene segmentation. In *European Conference on Computer Vision (ECCV)*, 2020. 109

[37] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 40(4), 2017. 2, 17

[38] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. Rethinking atrous convolution for semantic image segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 4, 111, 112, 115, 117

[39] M. Chen, H. Xue, and D. Cai. Domain adaptation for semantic segmentation with maximum squares loss. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 129, 137

[40] Q. Chen and V. Koltun. Full flow: Optical flow estimation by global optimization over regular grids. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 22, 87

[41] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning (ICML)*, 2020. 7, 26, 105, 106, 107, 108, 110, 117, 123, 131, 134, 194

[42] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. Hinton. Big self-supervised models are strong semi-supervised learners. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 26, 105, 107, 110, 131

[43] X. Chen, S. Bing Kang, J. Yang, and J. Yu. Fast patch-based denoising using approximated patch geodesic paths. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 69

[44] X. Chen, H. Fan, R. Girshick, and K. He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 26, 107, 110, 118, 120, 121, 131, 134

[45] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler, and R. Urtasun. Monocular 3d object detection for autonomous driving. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 175

[46] X. Chen, K. Kundu, Y. Zhu, A. G. Berneshawi, H. Ma, S. Fidler, and R. Urtasun. 3d object proposals for accurate object class detection. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2015. 175

[47] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia. Multi-view 3d object detection network for autonomous driving. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 172, 175, 178, 181, 185, 187, 188

[48] Y. Chen, W. Chen, Y. Chen, B. Tsai, Y. F. Wang, and M. Sun. No more discrimination: Cross city adaptation of road scene segmenters. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017. 27, 129

[49] Y. Chen, W. Li, and L. V. Gool. ROAD: reality oriented adaptation for semantic segmentation of urban scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 27, 129

[50] Y. Chen, Y. Lin, M. Yang, and J. Huang. Crdoco: Pixel-level domain transfer with cross-domain consistency. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 27, 129

[51] Z. Chen, X. Sun, L. Wang, Y. Yu, and C. Huang. A deep visual correspondence embedding model for stereo matching costs. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2015. 2, 9, 17, 18, 39

[52] J. Cheng, Y.-H. Tsai, S. Wang, and M.-H. Yang. Segflow: Joint learning for video object segmentation and optical flow. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017. 22, 87, 92, 198

[53] X. Cheng, P. Wang, and R. Yang. Depth estimation via affinity learned with convolutional spatial propagation network. In *European Conference on Computer Vision (ECCV)*, 2018. 47

[54] X. Cheng, P. Wang, and R. Yang. Learning depth with convolutional spatial propagation network. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 42(10), 2019. 89

[55] J. Choi, T. Kim, and C. Kim. Self-ensembling with gan-based data augmentation for domain adaptation in semantic segmentation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 129, 130

[56] C. Choy, J. Gwak, and S. Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 29, 149, 150, 151, 152, 157, 161, 163, 164, 165, 166, 167

[57] C. B. Choy, J. Gwak, S. Savarese, and M. Chandraker. Universal correspondence network. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016. 20, 65

[58] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 6, 8, 9, 75, 104, 116, 127, 140, 194

[59] G. Csurka, D. Larlus, F. Perronnin, and F. Meylan. What is a good evaluation measure for semantic segmentation? *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 26(1), 2004. 2, 3, 17

[60] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 29, 31, 152, 162, 164, 165, 176, 199

[61] J. Dai, K. He, and J. Sun. Instance-aware semantic segmentation via multi-task network cascades. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 173, 175

[62] T. DeVries and G. W. Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 117, 138

[63] J. Dong, Y. Cong, G. Sun, Y. Liu, and X. Xu. CSCL: Critical semantic-consistent learning for unsupervised domain adaptation. In *European Conference on Computer Vision (ECCV)*, 2020. 27, 129, 137

[64] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2015. 2, 7, 9, 17, 95

[65] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. Carla: An open urban driving simulator. *arXiv preprint arXiv:1711.03938*, 2017. 75

[66] X. Du, M. El-Khamy, and J. Lee. Amnet: Deep atrous multiscale stereo disparity estimation networks. *arXiv preprint arXiv:1904.09099*, 2019. 89

[67] M. Engelcke, D. Rao, D. Z. Wang, C. H. Tong, and I. Posner. Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2017. 175, 181

[68] M. Ester, H. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In E. Simoudis, J. Han, and U. M. Fayyad, editors, *KDD*, 1996. 121

[69] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html. 118, 119, 120, 121

[70] J. Fang, F. Yan, T. Zhao, F. Zhang, D. Zhou, R. Yang, Y. Ma, and L. Wang. Simulating lidar point cloud for autonomous driving using real-world scenes and traffic flows. *arXiv preprint arXiv:1811.07112*, 2018. 184

[71] J. Flynn, I. Neulander, J. Philbin, and N. Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 18, 39

[72] M. Gadelha, R. Wang, and S. Maji. Multiresolution tree networks for 3d point cloud processing. In *European Conference on Computer Vision (ECCV)*, September 2018. 31, 154

[73] D. Gadot and L. Wolf. Patchbatch: A batch augmented loss for optical flow. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 22, 87

[74] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11), 2013. 85, 174, 176, 184

[75] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 39, 49, 75

[76] M. Ghifary, W. Bastiaan Kleijn, M. Zhang, and D. Balduzzi. Domain gener-

alization for object recognition with multi-task autoencoders. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2015. 20, 65

[77] S. Gidaris, A. Bursuc, N. Komodakis, P. Pérez, and M. Cord. Learning representations by predicting bags of visual words. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 109

[78] R. Girshick. Fast r-cnn. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2015. 175

[79] B. Graham. Sparse 3d convolutional neural networks. *arXiv preprint arXiv:1505.02890*, 2015. 164

[80] B. Graham, M. Engelcke, and L. van der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 29, 149, 150, 152, 154, 157, 161, 163, 164, 167

[81] F. Groh, P. Wieschollek, and H. P. Lensch. Flex-convolution. In *Asian Conference on Computer Vision (ACCV)*, 2018. 30, 149, 152

[82] X. Guo, H. Li, S. Yi, J. Ren, and X. Wang. Learning monocular depth by distilling cross-domain stereo networks. In *European Conference on Computer Vision (ECCV)*, 2018. 8, 20, 61, 64

[83] X. Guo, K. Yang, W. Yang, X. Wang, and H. Li. Group-wise correlation stereo network. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 64, 78, 79, 80

[84] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik. Learning rich features from rgb-d images for object detection and segmentation. In *European Conference on Computer Vision (ECCV)*, 2014. 176

[85] T. Hackel, N. Savinov, L. Ladicky, J. D. Wegner, K. Schindler, and M. Pollefeys. SEMANTIC3D.NET: A new large-scale point cloud classification benchmark. In *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, volume IV-1-W1, pages 91–98, 2017. 176

[86] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 7, 26, 105, 106, 107, 108, 110, 111, 113, 115, 116, 118, 119, 120, 122, 123, 131, 134, 138, 194

[87] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017. 173, 175, 182

[88] K. He, J. Sun, and X. Tang. Guided image filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, (6), 2013. 40, 41

[89] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 5, 111

[90] T. He, H. Huang, L. Yi, Y. Zhou, C. Wu, J. Wang, and S. Soatto. Geonet: Deep geodesic networks for point cloud analysis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 30, 152

[91] O. Henaff. Data-efficient image recognition with contrastive predictive coding. In *International Conference on Learning Representation (ICLR)*, 2020. 26, 107, 110

[92] P. Hermosilla, T. Ritschel, P.-P. Vázquez, À. Vinacua, and T. Ropinski. Monte carlo convolution for learning on non-uniformly sampled point clouds. *ACM Transactions on Graphics (TOG)*, 37(6), 2018. 30, 149, 153, 156

[93] H. Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 30(2), 2008. 2, 17, 18, 23, 37, 38, 39, 40, 41, 42, 43, 45, 52, 56, 63, 72, 75, 78, 79, 88, 92, 192

[94] C.-H. Ho and N. Vasconcelos. Contrastive learning with adversarial examples. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 26, 108

[95] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. Efros, and T. Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *International Conference on Machine Learning (ICML)*, 2018. 129

[96] M. Hofinger, S. R. Bulò, L. Porzi, A. Knapitsch, and P. Kontschieder. Improving optical flow on a pyramidal level. In *European Conference on Computer Vision (ECCV)*, 2020. 22, 23, 87, 88

[97] B. K. Horn and B. G. Schunck. Determining optical flow. In *Techniques and Applications of Image Understanding*, volume 281. International Society for Optics and Photonics, 1981. 2, 17, 22, 84, 87

[98] A. Hosni, C. Rhemann, M. Bleyer, C. Rother, and M. Gelautz. Fast cost-volume filtering for visual correspondence and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 35(2), 2013. 18, 21, 22, 37, 38, 39, 40, 42, 46, 78, 79, 87, 88

[99] J. Hou, A. Dai, and M. Nießner. 3d-sis: 3d semantic instance segmentation of rgb-d scans. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 5, 29, 149, 152, 176, 182

[100] Q. Hu, X. Wang, W. Hu, and G.-J. Qi. AdCo: Adversarial contrast for efficient learning of unsupervised representations from self-trained negative

adversaries. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 26, 108

[101] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham. Randla-net: Efficient semantic segmentation of large-scale point clouds. *arXiv preprint arXiv:1911.11236*, 2019. 164, 167

[102] Y. Hu, Y. Li, and R. Song. Robust interpolation of correspondences for large displacement optical flow. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 21, 87

[103] Y. Hu, R. Song, and Y. Li. Efficient coarse-to-fine patchmatch for large displacement optical flow. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 21, 87

[104] B.-S. Hua, M.-K. Tran, and S.-K. Yeung. Pointwise convolutional neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 30, 153, 157, 161

[105] J. Huang, Q. Dong, S. Gong, and X. Zhu. Unsupervised deep learning by neighbourhood discovery. In *International Conference on Learning Representation (ICLR)*, 2019. 109

[106] J. Huang, H. Zhang, L. Yi, T. Funkhouser, M. Nießner, and L. J. Guibas. Texturenet: Consistent local parametrizations for learning from high-resolution signals on meshes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 30, 153

[107] Q. Huang, W. Wang, and U. Neumann. Recurrent slice networks for 3d segmentation of point clouds. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 31, 154

[108] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz. Multimodal unsupervised image-to-image translation. In *European Conference on Computer Vision (ECCV)*, 2018. 27, 127, 129, 132

[109] Z. Huang, X. Wang, L. Huang, C. Huang, Y. Wei, and W. Liu. Ccnet: Criss-cross attention for semantic segmentation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 71, 73, 77, 78

[110] T.-W. Hui and C. C. Loy. Liteflownet3: Resolving correspondence ambiguity for more accurate optical flow estimation. In *European Conference on Computer Vision (ECCV)*, 2020. 22, 23, 88

[111] T.-W. Hui, X. Tang, and C. Change Loy. Liteflownet: A lightweight convolutional neural network for optical flow estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 17, 22, 87, 96, 102

[112] T.-W. Hui, X. Tang, and C. C. Loy. A lightweight optical flow cnn–revisiting data fidelity and regularization. *arXiv preprint arXiv:1903.07414*, 2019. 22, 87, 102

[113] J. Hur and S. Roth. Iterative residual refinement for joint optical flow and occlusion estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 22, 87

[114] E. Ilg, O. Cicek, S. Galesso, A. Klein, O. Makansi, F. Hutter, and T. Brox. Uncertainty estimates and multi-hypotheses networks for optical flow. In *European Conference on Computer Vision (ECCV)*, 2018. 22, 87

[115] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 7, 12, 63, 81, 85, 96, 100, 102, 194

[116] W. Im, T.-K. Kim, and S.-E. Yoon. Unsupervised learning of optical flow with deep feature similarity. In *European Conference on Computer Vision (ECCV)*, 2020. 23, 87

[117] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 77

[118] J. Janai, F. Guney, A. Ranjan, M. Black, and A. Geiger. Unsupervised learning of multi-frame optical flow with occlusions. In *European Conference on Computer Vision (ECCV)*, 2018. 23, 87

[119] M. Jaritz, J. Gu, and H. Su. Multi-view pointnet for 3d scene understanding. In *IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, 2019. 164

[120] X. Ji, J. F. Henriques, and A. Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 109

[121] X. Jia, B. De Brabandere, T. Tuytelaars, and L. V. Gool. Dynamic filter networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016. 46

[122] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM International Conference on Multimedia*. 49, 54

[123] L. Jiang, H. Zhao, S. Liu, X. Shen, C.-W. Fu, and J. Jia. Hierarchical point-edge interaction network for point cloud semantic segmentation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 31, 154

[124] Z. Jiang, T. Chen, T. Chen, and Z. Wang. Robust pre-training by adversarial contrastive learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 26, 108

[125] R. Jonschkowski, A. Stone, J. T. Barron, A. Gordon, K. Konolige, and A. Angelova. What matters in unsupervised optical flow. *arXiv preprint arXiv:2006.04902*, 1(2), 2020. 23, 87

[126] Y. Kalantidis, M. B. Sariyildiz, N. Pion, P. Weinzaepfel, and D. Larlus. Hard negative mixing for contrastive learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 26, 108

[127] G. Kang, Y. Wei, Y. Yang, Y. Zhuang, and A. G. Hauptmann. Pixel-level cycle association: A new perspective for domain adaptive semantic segmentation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 130

[128] A. Kendall, H. Martirosyan, S. Dasgupta, P. Henry, R. Kennedy, A. Bachrach, and A. Bry. End-to-end learning of geometry and context for deep stereo regression. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017. 2, 4, 5, 10, 17, 19, 37, 38, 40, 42, 48, 54, 55, 56, 60, 63, 64, 66, 74, 79, 89, 92, 93, 94, 193

[129] S. Khamis, S. R. Fanello, C. Rhemann, A. Kowdle, J. P. C. Valentin, and S. Izadi. Stereonet: Guided hierarchical refinement for real-time edge-aware depth prediction. *CoRR*, abs/1807.08865, 2018. 19, 64

[130] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan. Supervised contrastive learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 108, 110, 113, 134

[131] M. Kim and H. Byun. Learning texture invariant representation for domain

adaptation of semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 129

[132] M. Kim, J. Tack, and S. J. Hwang. Adversarial self-supervised contrastive learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 26, 108

[133] W. Kim, A. Kanezaki, and M. Tanaka. Unsupervised learning of image segmentation based on differentiable feature clustering. *IEEE Transactions on Image Processing (TIP)*, 29, 2020. 109

[134] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 185

[135] A. Komarichev, Z. Zhong, and J. Hua. A-cnn: Annularly convolutional neural networks on point clouds. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 30, 153

[136] D. Kondermann, R. Nair, K. Honauer, K. Krispin, J. Andrulis, A. Brock, B. Gussefeld, M. Rahimimoghaddam, S. Hofmann, C. Brenner, et al. The hci benchmark suite: Stereo and flow ground truth with uncertainties for urban autonomous driving. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2016. 24, 95

[137] T. Kroeger, R. Timofte, D. Dai, and L. Van Gool. Fast optical flow using dense inverse search. In *European Conference on Computer Vision (ECCV)*, 2016. 22, 87

[138] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander. Joint 3d proposal generation and object detection from view aggregation. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018. 175

[139] A. Kundu, K. Genova, X. Yin, A. Fathi, C. Pantofaru, L. J. Guibas, A. Tagliasacchi, F. Dellaert, and T. Funkhouser. Panoptic neural fields: A semantic object-aware neural scene representation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 199

[140] J. Lahoud, B. Ghanem, M. Pollefeys, and M. R. Oswald. 3d instance segmentation via multi-task metric learning. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 29, 152

[141] W.-S. Lai, J.-B. Huang, and M.-H. Yang. Semi-supervised learning for optical flow with generative adversarial networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 87

[142] J. Lambert, Z. Liu, O. Sener, J. Hays, and V. Koltun. MSeg: A composite dataset for multi-domain semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 4

[143] S. Lan, R. Yu, G. Yu, and L. S. Davis. Modeling local geometric structure of 3d point clouds using geo-cnn. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 30, 153

[144] L. Landrieu and M. Boussaha. Point cloud oversegmentation with graph-structured deep metric learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 31, 154

[145] L. Landrieu and M. Simonovsky. Large-scale point cloud semantic segmentation with superpoint graphs. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 31, 154, 164

[146] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 185, 189

[147] T. Le and Y. Duan. Pointgrid: A deep network for 3d shape understanding. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 29, 149, 152

[148] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4), 1989. 17

[149] H. Lei, N. Akhtar, and A. Mian. Octree guided cnn with spherical kernels for 3d point clouds. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 31, 153

[150] A. Li and Z. Yuan. Occlusion aware stereo matching via cooperative unsupervised learning. In *Asian Conference on Computer Vision (ACCV)*, 2018. 79

[151] D. Li, Y. Yang, Y.-Z. Song, and T. M. Hospedales. Deeper, broader and artier domain generalization. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017. 20, 65

[152] D. Li, Y. Yang, Y.-Z. Song, and T. M. Hospedales. Learning to generalize: Meta-learning for domain generalization. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2018. 20, 65

[153] H. Li, S. Jialin Pan, S. Wang, and A. C. Kot. Domain generalization with adversarial feature learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 65

[154] J. Li, P. Zhou, C. Xiong, and S. C. H. Hoi. Prototypical contrastive learning of unsupervised representations. In *International Conference on Learning Representation (ICLR)*, 2021. 118

[155] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen. Pointcnn: Convolution on x-transformed points. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. 30, 149, 151, 152, 153, 154, 156, 164, 166, 168, 178

[156] Y. Li, J. Chen, X. Xie, K. Ma, and Y. Zheng. Self-loop uncertainty: A novel pseudo-label for semi-supervised medical image segmentation. In *International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI)*, 2020. 109

[157] Y. Li, X. Tian, M. Gong, Y. Liu, T. Liu, K. Zhang, and D. Tao. Deep domain generalization via conditional invariant adversarial networks. In *European Conference on Computer Vision (ECCV)*, 2018. 65

[158] Y. Li, N. Wang, J. Shi, X. Hou, and J. Liu. Adaptive batch normalization for practical domain adaptation. *Pattern Recognition*, 80, 2018. 77

[159] Y. Li, N. Wang, J. Shi, J. Liu, and X. Hou. Revisiting batch normalization for practical domain adaptation. *arXiv preprint arXiv:1603.04779*, 2016. 77

[160] Y. Li, L. Yuan, and N. Vasconcelos. Bidirectional learning for domain adaptation of semantic segmentation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 127, 141

[161] M. Liang, B. Yang, S. Wang, and R. Urtasun. Deep continuous fusion for multi-sensor 3d object detection. In *European Conference on Computer Vision (ECCV)*, 2018. 175

[162] Z. Liang, Y. Feng, Y. Guo, H. Liu, W. Chen, L. Qiao, L. Zhou, and J. Zhang. Learning for disparity estimation through feature constancy. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 19, 64

[163] D. Lin, S. Fidler, and R. Urtasun. Holistic scene understanding for 3d object detection with rgbd cameras. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2013. 176

[164] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 175

[165] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017. 175, 183

[166] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision (ECCV)*, 2014. 116, 119, 120, 185

[167] C. Liu and Y. Furukawa. Masc: Multi-scale affinity with sparse convolution for 3d instance segmentation. *arXiv preprint arXiv:1902.04478*, 2019. 176

[168] L. Liu, J. Zhang, R. He, Y. Liu, Y. Wang, Y. Tai, D. Luo, C. Wang, J. Li, and F. Huang. Learning by analogy: Reliable supervision from transformations for unsupervised optical flow estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 23, 87

[169] M.-Y. Liu, O. Tuzel, and Y. Taguchi. Joint geodesic upsampling of depth images. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 69, 70

[170] P. Liu, M. Lyu, I. King, and J. Xu. Selflow: Self-supervised learning of optical flow. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 23, 87

[171] S. Liu, S. De Mello, J. Gu, G. Zhong, M.-H. Yang, and J. Kautz. Learning affinity via spatial propagation networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 42, 71, 72, 73, 77, 78

[172] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *European Conference on Computer Vision (ECCV)*, 2016. 175

[173] W. Liu, D. Ferstl, S. Schulter, L. Zebedin, P. Fua, and C. Leistner. Domain adaptation for semantic segmentation via patch-wise contrastive learning. *arXiv*, abs/2104.11056, 2021. 28, 131

[174] Y. Liu, J. Deng, X. Gao, W. Li, and L. Duan. Bapa-net: Boundary adaptation and prototype alignment for cross-domain semantic segmentation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 141

[175] Y. Liu, B. Fan, G. Meng, J. Lu, S. Xiang, and C. Pan. Densepoint: Learning densely contextual representation for efficient point cloud processing. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 30, 149, 152

[176] Z. Liu, H. Tang, Y. Lin, and S. Han. Point-voxel cnn for efficient 3d deep learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 29, 152, 154, 164, 166, 167

[177] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 2, 3, 17

[178] Y. Lu, J. Valmadre, H. Wang, J. Kannala, M. Harandi, and P. Torr. Devon: Deformable volume network for learning optical flow. In *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2020. 22, 88

[179] W. Luo, A. G. Schwing, and R. Urtasun. Efficient deep learning for stereo matching. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 63, 79

[180] F. Lv, T. Liang, X. Chen, and G. Lin. Cross-domain semantic segmentation via domain-invariant interactive relation transfer. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 130

[181] H. Ma, X. Lin, Z. Wu, and Y. Yu. Coarse-to-fine domain adaptive semantic segmentation with photometric alignment and category-center regularization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 28, 127, 130, 134, 136, 138, 140, 141

[182] J. Maier, M. Humenberger, M. Murschitz, O. Zendel, and M. Vincze. Guided matching based on statistical optical flow for fast and robust correspondence analysis. In *European Conference on Computer Vision (ECCV)*, 2016. 21, 87

[183] J. Mao, X. Wang, and H. Li. Interpolated convolutional networks for 3d point cloud understanding. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 30, 149, 153, 156

[184] R. A. Marsden, A. Bartler, M. Döbler, and B. Yang. Contrastive learning and self-training for unsupervised domain adaptation in semantic segmentation. *arXiv*, abs/2105.02001, 2021. 28, 131

[185] D. Maturana and S. Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015. 29, 149, 151, 152, 154, 178

[186] D. Maurer, N. Marniok, B. Goldluecke, and A. Bruhn. Structure-from-motion-aware patchmatch for adaptive optical flow estimation. In *European Conference on Computer Vision (ECCV)*, 2018. 21, 87

[187] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 6, 7, 9, 11, 18, 24, 38, 39, 40, 49, 54, 55, 61, 63, 64, 66, 75, 79, 81, 95

[188] L. McInnes, J. Healy, and J. Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv*, abs/1802.03426, 2018. 133, 142

[189] K. Mei, C. Zhu, J. Zou, and S. Zhang. Instance adaptive self-training for unsupervised domain adaptation. In *European Conference on Computer Vision (ECCV)*, 2020. 28, 130

[190] H.-Y. Meng, L. Gao, Y.-K. Lai, and D. Manocha. Vv-net: Voxel vae net with group convolutions for point cloud segmentation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 29, 152

[191] M. Menze and A. Geiger. Object scene flow for autonomous vehicles. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 24, 39, 49, 61, 75, 79, 95, 191, 194, 198, 199

[192] M. Menze, C. Heipke, and A. Geiger. Discrete optimization for optical flow. In *German Conference on Pattern Recognition*, 2015. 22, 87

[193] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision (ECCV)*, 2020. 199

[194] I. Misra and L. v. d. Maaten. Self-supervised learning of pretext-invariant representations. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 26, 108

[195] T. Moriya, H. R. Roth, S. Nakamura, H. Oda, K. Nagara, M. Oda, and K. Mori. Unsupervised segmentation of 3d medical images based on clustering and deep representation learning. In *Medical Imaging 2018: Biomedical Applications in Molecular, Structural, and Functional Imaging*, volume 10578. SPIE, 2018. 109

[196] S. Motiian, M. Piccirilli, D. A. Adjeroh, and G. Doretto. Unified deep supervised domain adaptation and generalization. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017. 20, 65

[197] D. Mugnai, F. Pernici, F. Turchini, and A. D. Bimbo. Soft pseudo-labeling semi-supervised learning applied to fine-grained visual classification. In *Internal Conference on Pattern Recognition Workshops*, 2020. 109

[198] L. Musto and A. Zinelli. Semantically adaptive image-to-image translation for domain adaptation of semantic segmentation. In *British Machine Vision Conference (BMVC)*, 2020. 27, 129

[199] H. Nam and H.-E. Kim. Batch-instance normalization for adaptively style-invariant neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. 20, 65, 67, 77

[200] G. Narita, T. Seno, T. Ishikawa, and Y. Kaji. Panopticfusion: Online volumetric semantic mapping at the level of stuff and things. *arXiv preprint arXiv:1903.01177*, 2019. 29, 152

[201] G. Neuhold, T. Ollmann, S. Rota Bulo, and P. Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017. 140

[202] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. In *European Conference on Computer Vision (ECCV)*, 2016. 19, 38, 40

[203] G.-Y. Nie, M.-M. Cheng, Y. Liu, Z. Liang, D.-P. Fan, Y. Liu, and Y. Wang. Multi-level context ultra-aggregation for stereo matching. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 19, 64

[204] F. Pan, I. Shin, F. Rameau, S. Lee, and I. S. Kweon. Unsupervised intra-domain adaptation for semantic segmentation through self-supervision. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 140, 141

[205] H. Pan, S. Liu, Y. Liu, and X. Tong. Convolutional neural networks on 3d surfaces using parallel frames. *arXiv preprint arXiv:1808.04952*, 2018. 30, 149, 153, 156, 164

[206] X. Pan, P. Luo, J. Shi, and X. Tang. Two at once: Enhancing learning and generalization capacities via ibn-net. In *European Conference on Computer Vision (ECCV)*, 2018. 20, 65

[207] J. Pang, W. Sun, J. S. Ren, C. Yang, and Q. Yan. Cascade residual learning: A two-stage convolutional neural network for stereo matching. *IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, 2017. 18, 40, 64

[208] J. Pang, W. Sun, C. Yang, J. Ren, R. Xiao, J. Zeng, and L. Lin. Zoom and learn: Generalizing deep stereo matching to novel domains. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 20, 64

[209] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu. Semantic image synthesis with spatially-adaptive normalization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 67, 198

[210] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017. 94

[211] A. Patil, S. Malla, H. Gang, and Y.-T. Chen. The h3d dataset for full-surround 3d multi-object detection and tracking in crowded urban scenes. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2019. 176, 184

[212] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12, 2011. 112

[213] Q.-H. Pham, T. Nguyen, B.-S. Hua, G. Roig, and S.-K. Yeung. Jsis3d: Joint semantic-instance segmentation of 3d point clouds with multi-task pointwise networks and multi-value conditional random fields. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 31, 154

[214] P. O. Pinheiro, R. Collobert, and P. Dollár. Learning to segment object candidates. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2015. 175

[215] P. O. Pinheiro, T.-Y. Lin, R. Collobert, and P. Dollár. Learning to refine object segments. In *European Conference on Computer Vision (ECCV)*, 2016. 175

[216] M. Poggi, D. Pallotti, F. Tosi, and S. Mattoccia. Guided stereo matching. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 20, 64

[217] S. Prokudin, C. Lassner, and J. Romero. Efficient learning on point clouds with basis point sets. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 31, 154

[218] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 175

[219] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2, 17, 30, 149, 151, 152, 157, 159, 160, 161, 164, 167, 168, 178

[220] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. J. Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 7, 29, 149, 151, 152

[221] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems (NeurIPS)*. 30, 149, 150, 151, 152, 154, 157, 160, 161, 163, 164, 167, 168, 178

[222] B. Ranft and T. Strauß. Modeling arbitrarily oriented slanted planes for efficient stereo vision based on block matching. In *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2014. 54

[223] R. Ranftl, K. Bredies, and T. Pock. Non-local total generalized variation for optical flow estimation. In *European Conference on Computer Vision (ECCV)*, 2014. 22, 87

[224] A. Ranjan and M. J. Black. Optical flow estimation using a spatial pyramid network. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 22, 87

[225] Y. Rao, J. Lu, and J. Zhou. Spherical fractal convolutional neural networks for point cloud recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 30, 153

[226] S. N. Ravi, Y. Xiong, L. Mukherjee, and V. Singh. Filter flow made practical:

Massively parallel and lock-free. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 22, 87

[227] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 175

[228] J. Redmon and A. Farhadi. Yolo9000: better, faster, stronger. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 175

[229] J. Redmon and A. Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 175

[230] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2015. 175

[231] D. Rethage, J. Wald, J. Sturm, N. Navab, and F. Tombari. Fully-convolutional point networks for large-scale point clouds. In *European Conference on Computer Vision (ECCV)*, September 2018. 31, 154

[232] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid. Epicflow: Edge-preserving interpolation of correspondences for optical flow. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 21, 87

[233] S. R. Richter, Z. Hayder, and V. Koltun. Playing for benchmarks. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017. 118, 121

[234] S. R. Richter, V. Vineet, S. Roth, and V. Koltun. Playing for data: Ground truth from computer games. In *European Conference on Computer Vision (ECCV)*, 2016. 9, 127, 128, 140

[235] G. Riegler, A. Osman Ulusoy, and A. Geiger. Octnet: Learning deep 3d representations at high resolutions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 29, 152

[236] J. D. Robinson, C.-Y. Chuang, S. Sra, and S. Jegelka. Contrastive learning with hard negative samples. In *International Conference on Learning Representation (ICLR)*, 2021. 26, 108

[237] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI)*, 2015. 150, 159

[238] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 128, 140

[239] S. Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016. 117

[240] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *nature*, 323(6088), 1986. 17

[241] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. Labelme: a database and web-based tool for image annotation. *International Journal of Computer Vision (IJCV)*, 77(1-3), 2008. 176

[242] D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nešić, X. Wang, and P. Westling. High-resolution stereo datasets with subpixel-accurate ground truth. In *German conference on pattern recognition*, 2014. 75

[243] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision (IJCV)*, 47(1-3), 2002. 18, 37

[244] J. L. Schönberger, S. N. Sinha, and M. Pollefeys. Learning to fuse proposals from multiple scanline optimizations in semi-global matching. In *European Conference on Computer Vision (ECCV)*, 2018. 18, 40

[245] T. Schops, J. L. Schonberger, S. Galliani, T. Sattler, K. Schindler, M. Pollefeys, and A. Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 75

[246] R. Schuster, C. Bailer, O. Wasenmüller, and D. Stricker. Flowfields++: Accurate optical flow correspondences meet robust interpolation. In *IEEE International Conference on Image Processing*, 2018. 7, 21, 87, 102

[247] T. Schuster, L. Wolf, and D. Gadot. Optical flow requires multiple strategies (but only one network). In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 22, 87

[248] A. Seki and M. Pollefeys. Sgm-nets: Semi-global matching with neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 18, 39, 42, 56, 63

[249] L. Sevilla-Lara, D. Sun, V. Jampani, and M. J. Black. Optical flow with semantic segmentation and localized layers. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 22, 87

[250] L. Shi, Y. Zhang, J. Cheng, and H. Lu. Non-local graph convolutional networks for skeleton-based action recognition. *arXiv preprint arXiv:1805.07694*, 2018. 73

[251] S. Shi, X. Wang, and H. Li. Pointrcnn: 3d object proposal generation and detection from point cloud. *arXiv preprint arXiv:1812.04244*, 2018. 172, 173, 175, 181, 185, 189

[252] A. Shrivastava and A. Gupta. Building part-based object detectors via 3d geometry. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2013. 176

[253] N. Silberman and R. Fergus. Indoor scene segmentation using a structured light sensor. In *IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, 2011. 176

[254] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from RGBD images. In *European Conference on Computer Vision (ECCV)*, 2012. 8, 116

[255] M. Simon, S. Milz, K. Amende, and H.-M. Gross. Complex-yolo: An euler-region-proposal for real-time 3d object detection on point clouds. In *European Conference on Computer Vision (ECCV)*. 175, 181

[256] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 5

[257] L. Song, Y. Li, Z. Li, G. Yu, H. Sun, J. Sun, and N. Zheng. Learnable tree filter for structure-preserving feature transform. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 70

[258] S. Song, S. P. Lichtenberg, and J. Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 176

[259] S. Song and J. Xiao. Sliding shapes for 3d object detection in depth images. In *European Conference on Computer Vision (ECCV)*, 2014. 176

[260] S. Song and J. Xiao. Deep sliding shapes for amodal 3d object detection in rgb-d images. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 176

[261] X. Song, X. Zhao, L. Fang, and H. Hu. Edgestereo: An effective multi-task learning network for stereo matching and edge detection. *arXiv preprint arXiv:1903.01700*, 2019. 19, 64, 66

[262] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014. 44

[263] H. Su, V. Jampani, D. Sun, S. Maji, E. Kalogerakis, M.-H. Yang, and J. Kautz. Splatnet: Sparse lattice networks for point cloud processing. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 31, 154, 164

[264] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2015. 7, 29, 149, 151

[265] M. N. Subhani and M. Ali. Learning from scale-invariant examples for domain adaptation in semantic segmentation. In *European Conference on Computer Vision (ECCV)*, 2020. 28, 130

[266] D. Sun, D. Vlasic, C. Herrmann, V. Jampani, M. Krainin, H. Chang, R. Zabih, W. T. Freeman, and C. Liu. Autoflow: Learning a better training set for optical flow. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 198

[267] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz. Models matter, so does training: An empirical study of cnns for optical flow estimation. *arXiv preprint arXiv:1809.05571*, 2018. 102

[268] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 4, 7, 9, 12, 22, 24, 63, 80, 81, 84, 85, 87, 88, 89, 90, 93, 96, 99, 100, 102, 194

[269] R. Szeliski. Computer vision: applications and algorithms, 2011. 1

[270] K. D. Tang, V. Ramanathan, L. Fei-Fei, and D. Koller. Shifting weights: Adapting object detectors from image to video. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2012. 129

[271] A. Tarvainen and H. Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 28, 109, 133

[272] M. Tatarchenko, J. Park, V. Koltun, and Q.-Y. Zhou. Tangent convolutions for dense prediction in 3d. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3, 31, 153, 164

[273] G. Te, W. Hu, A. Zheng, and Z. Guo. Rgcnn: Regularized graph cnn for point cloud segmentation. In *Proceedings of the ACM International Conference on Multimedia*, 2018. 31, 153

[274] Z. Teed and J. Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European Conference on Computer Vision (ECCV)*, 2020. 2, 4, 5, 6, 17, 22, 23, 24, 84, 86, 87, 88, 89, 90, 91, 93, 94, 95, 96, 97, 98, 99, 100, 102

[275] Z. Teed and J. Deng. Raft-3d: Scene flow using rigid-motion embeddings. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 198

[276] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 30, 149, 152, 153, 161, 164

[277] Y. Tian, D. Krishnan, and P. Isola. Contrastive multiview coding. In *European Conference on Computer Vision (ECCV)*, 2020. 26, 108

[278] Y. Tian, C. Sun, B. Poole, D. Krishnan, C. Schmid, and P. Isola. What makes for good views for contrastive learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 26, 108

[279] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 1998. 40, 41

[280] A. Tonioni, M. Poggi, S. Mattoccia, and L. Di Stefano. Unsupervised adaptation for deep stereo. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct 2017. 20, 64

[281] A. Tonioni, O. Rahnama, T. Joy, L. D. Stefano, T. Ajanthan, and P. H. Torr. Learning to adapt for stereo. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 20, 64

[282] A. Tonioni, F. Tosi, M. Poggi, S. Mattoccia, and L. D. Stefano. Real-time self-adaptive deep stereo. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 8, 19, 20, 60, 61, 64, 66, 79, 193

[283] T.-D. Truong, C. N. Duong, N. Le, S. L. Phung, C. Rainwater, and K. Luu. Bimal: Bijective maximum likelihood approach to domain adaptation in semantic scene segmentation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 141

[284] Y. Tsai, W. Hung, S. Schulter, K. Sohn, M. Yang, and M. Chandraker. Learning to adapt structured output space for semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 27, 129

[285] Y. Tsai, K. Sohn, S. Schulter, and M. Chandraker. Domain adaptation for structured output via discriminative patch representations. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 27, 129

[286] S. Tulyakov, A. Ivanov, and F. Fleuret. Weakly supervised learning of deep metrics for stereo reconstruction. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017. 79

[287] S. Tulyakov, A. Ivanov, and F. Fleuret. Practical deep stereo (pds): Toward applications-friendly deep stereo matching. *arXiv preprint arXiv:1806.01677*, 2018. 40

[288] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016. 77

[289] W. Van Gansbeke, S. Vandenhende, S. Georgoulis, and L. Van Gool. Unsupervised semantic segmentation by contrasting object mask proposals. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 27, 108, 131

[290] N. Verma, E. Boyer, and J. Verbeek. Feastnet: Feature-steered graph convolutions for 3d shape analysis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 30, 149, 152, 153

[291] T. Vu, H. Jain, M. Bucher, M. Cord, and P. Pérez. ADVENT: adversarial entropy minimization for domain adaptation in semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 27, 129

[292] C. Wang, B. Samari, and K. Siddiqi. Local spectral graph convolution for point set feature learning. In *European Conference on Computer Vision (ECCV)*, September 2018. 31, 153

[293] D. Z. Wang and I. Posner. Voting for voting in online point cloud object detection. In *Robotics: Science and Systems*, volume 1, 2015. 175, 181

[294] H. Wang, T. Shen, W. Zhang, L. Duan, and T. Mei. Classes matter: A fine-grained adversarial approach to cross-domain semantic segmentation. In *European Conference on Computer Vision (ECCV)*, 2020. 27, 129, 140, 141

[295] J. Wang, Y. Zhong, Y. Dai, K. Zhang, P. Ji, and H. Li. Displacement-invariant matching cost learning for accurate optical flow estimation. *arXiv preprint arXiv:2010.14851*, 2020. 22, 23, 88, 90, 102

[296] K. Wang, C. Yang, and M. Betke. Consistency regularization with high-dimensional non-adversarial source-guided perturbation for unsupervised domain adaptation in segmentation. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2021. 130

[297] L. Wang, Y. Huang, Y. Hou, S. Zhang, and J. Shan. Graph attention convolution for point cloud semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 30, 31, 152, 153

[298] P. Wang, K. Han, X.-S. Wei, L. Zhang, and L. Wang. Contrastive learning based hybrid networks for long-tailed image classification. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 108

[299] S. Wang, S. Suo, W.-C. Ma, A. Pokrovsky, and R. Urtasun. Deep parametric continuous convolutional neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 30, 149, 151, 153, 156, 164

[300] W. Wang, R. Yu, Q. Huang, and U. Neumann. Sgpn: Similarity group proposal network for 3d point cloud instance segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 176, 182, 185, 187, 188

[301] W. Wang, T. Zhou, F. Yu, J. Dai, E. Konukoglu, and L. Van Gool. Exploring cross-image pixel contrast for semantic segmentation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 26, 105, 106, 108, 110, 131, 135

[302] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 71, 73

[303] X. Wang, J. He, and L. Ma. Exploiting local and global structure for point cloud semantic segmentation with contextual point representations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 31, 154

[304] X. Wang, R. Zhang, C. Shen, T. Kong, and L. Li. Dense contrastive learning for self-supervised visual pre-training. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 26, 105, 106, 108, 118, 119, 131

[305] Y. Wang, Z. Lai, G. Huang, B. H. Wang, L. Van Der Maaten, M. Campbell, and K. Q. Weinberger. Anytime stereo image depth estimation on mobile devices. *arXiv preprint arXiv:1810.11408*, 2018. 19, 64

[306] Y. Wang, J. Peng, and Z. Zhang. Uncertainty-aware pseudo label refinery for domain adaptive semantic segmentation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 141

[307] Y. Wang, Y. Yang, Z. Yang, L. Zhao, P. Wang, and W. Xu. Occlusion aware unsupervised learning of optical flow. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 23, 87

[308] Z. Wang, M. Yu, Y. Wei, R. Feris, J. Xiong, W. Hwu, T. S. Huang, and H. Shi. Differential treatment for stuff and things: A simple unsupervised domain adaptation method for semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 27, 127, 129, 141

[309] L. Wei, L. Xie, J. He, J. Chang, X. Zhang, W. Zhou, H. Li, and Q. Tian. Can semantic labels assist self-supervised visual representation learning? *arXiv preprint arXiv:2011.08621*, 2020. 106, 108, 110, 131

[310] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid. Deepflow: Large displacement optical flow with deep matching. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2013. 22, 87

[311] B. Wu, X. Zhou, S. Zhao, X. Yue, and K. Keutzer. Squeezesegv2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2019. 164

[312] W. Wu, Z. Qi, and L. Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 30, 149, 150, 151, 153, 156, 157, 161, 163, 164, 165, 166, 167, 168

[313] Z. Wu, X. Han, Y.-L. Lin, M. Gokhan Uzunbas, T. Goldstein, S. Nam Lim, and L. S. Davis. Dcan: Dual channel-wise alignment networks for unsupervised scene adaptation. In *European Conference on Computer Vision (ECCV)*, 2018. 127

[314] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 29, 149, 151, 152

[315] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin. Unsupervised feature learning via non-parametric instance discrimination. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 26, 108

[316] J. Wulff, L. Sevilla-Lara, and M. J. Black. Optical flow in mostly rigid scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 22, 87, 102

[317] Y. Xiang, W. Choi, Y. Lin, and S. Savarese. Data-driven 3d voxel patterns for object category recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 175

[318] T. Xiao, C. J. Reed, X. Wang, K. Keutzer, and T. Darrell. Region similarity representation learning. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 27, 108, 131

[319] T. Xiao, J. Yuan, D. Sun, Q. Wang, X.-Y. Zhang, K. Xu, and M.-H. Yang. Learnable cost volume using the Cayley representation. In *European Conference on Computer Vision (ECCV)*, 2020. 23, 88, 100, 102

[320] C. Xie, Y. Wu, L. v. d. Maaten, A. L. Yuille, and K. He. Feature denoising for improving adversarial robustness. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 71, 73, 77, 78

[321] J. Xie, R. Girshick, and A. Farhadi. Unsupervised deep embedding for clustering analysis. In *International Conference on Learning Representation (ICLR)*, 2016. 109

[322] Z. Xie, Y. Lin, Z. Zhang, Y. Cao, S. Lin, and H. Hu. Propagate yourself: Exploring pixel-level consistency for unsupervised visual representation learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 26, 27, 105, 106, 108, 118, 131

[323] D. Xu, D. Anguelov, and A. Jain. Pointfusion: Deep sensor fusion for 3d bounding box estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 175

[324] J. Xu, R. Ranftl, and V. Koltun. Accurate optical flow via direct cost volume processing. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 22, 23, 87, 88, 92, 102

[325] S. Xu, F. Zhang, X. He, X. Shen, and X. Zhang. Pm-pm: Patchmatch with potts model for object segmentation and stereo matching. *IEEE Transactions on Image Processing (TIP)*, 24(7), July 2015. 7, 40

[326] Y. Xu, T. Fan, M. Xu, L. Zeng, and Y. Qiao. Spidercnn: Deep learning on point sets with parameterized convolutional filters. In *European Conference on Computer Vision (ECCV)*, September 2018. 30, 149, 152, 156

[327] W. Yan, A. Sharma, and R. T. Tan. Optical flow in dense foggy scenes using semi-supervised learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 87

[328] X. Yan, I. Misra, A. Gupta, D. Ghadiyaram, and D. Mahajan. ClusterFit: Improving generalization of visual representations. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 109

[329] X. Yan, C. Zheng, Z. Li, S. Wang, and S. Cui. Pointasnl: Robust point clouds processing using nonlocal neural networks with adaptive sampling. In

IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2020. 164

[330] B. Yang, W. Luo, and R. Urtasun. Pixor: Real-time 3d object detection from point clouds. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 7, 172, 175, 178, 185, 187, 188

[331] B. Yang, J. Wang, R. Clark, Q. Hu, S. Wang, A. Markham, and N. Trigoni. Learning object bounding boxes for 3d instance segmentation on point clouds. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 31, 149, 151, 154

[332] G. Yang and D. Ramanan. Volumetric correspondence networks for optical flow. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 22, 23, 84, 87, 88, 100, 102

[333] G. Yang, H. Zhao, J. Shi, Z. Deng, and J. Jia. Segstereo: Exploiting semantic information for disparity estimation. *arXiv preprint arXiv:1807.11699*, 2018. 19, 64

[334] J. Yang, W. An, S. Wang, X. Zhu, C. Yan, and J. Huang. Label-driven reconstruction for domain adaptation in semantic segmentation. In *European Conference on Computer Vision (ECCV)*, 2020. 27, 129

[335] J. Yang, W. An, C. Yan, P. Zhao, and J. Huang. Context-aware domain adaptation in semantic segmentation. In *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2021. 27, 129

[336] J. Yang, C. Li, W. An, H. Ma, Y. Guo, Y. Rong, P. Zhao, and J. Huang. Exploring robustness of unsupervised domain adaptation in semantic segmentation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 28, 131

[337] J. Yang, D. Parikh, and D. Batra. Joint unsupervised learning of deep representations and image clusters. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 109

[338] J. Yang, Q. Zhang, B. Ni, L. Li, J. Liu, M. Zhou, and Q. Tian. Modeling point clouds with self-attention and gumbel subset sampling. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 31, 154

[339] Q. Yang. A non-local cost aggregation method for stereo matching. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 40, 69

[340] Q. Yang. Stereo matching using tree filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 37(4), 2014. 70

[341] Y. Yang, D. Lao, G. Sundaramoorthi, and S. Soatto. Phase consistent ecological domain adaptation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 129

[342] Y. Yang and S. Soatto. S2f: Slow-to-fast interpolator flow. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 21, 87

[343] Y. Yang and S. Soatto. Conditional prior networks for optical flow. In *European Conference on Computer Vision (ECCV)*, 2018. 23, 87

[344] Y. Yang and S. Soatto. FDA: Fourier domain adaptation for semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 27, 129, 141

[345] Z. Yang, Y. Sun, S. Liu, X. Shen, and J. Jia. Ipod: Intensive point-based object detector for point cloud. *arXiv preprint arXiv:1812.05276*, 2018. 175

[346] X. Ye, J. Li, H. Huang, L. Du, and X. Zhang. 3d recurrent neural networks with context fusion for point cloud semantic segmentation. In *European Conference on Computer Vision (ECCV)*, September 2018. 31, 154, 164

[347] L. Yi, W. Zhao, H. Wang, M. Sung, and L. Guibas. Gspn: Generative shape proposal network for 3d instance segmentation in point cloud. *arXiv preprint arXiv:1812.03320*, 2018. 176

[348] W. Yifan, S. Wu, H. Huang, D. Cohen-Or, and O. Sorkine-Hornung. Patch-based progressive 3d point set upsampling. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 31, 154

[349] Z. Yin, T. Darrell, and F. Yu. Hierarchical discrete distribution decomposition for match density estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 19, 22, 64, 77, 78, 79, 80, 87, 102

[350] C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime tv-l 1 optical flow. In *Joint Pattern Recognition Symposium*, 2007. 22, 87

[351] S. Zagoruyko and N. Komodakis. Learning to compare image patches via convolutional neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 18, 39

[352] J. Zbontar and Y. LeCun. Computing the stereo matching cost with a convolutional neural network. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 7, 11, 37, 42, 52, 56, 63, 79

[353] F. Zhang, L. Dai, S. Xiang, and X. Zhang. Segment graph based image filtering: fast structure-preserving smoothing. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2015. 40, 41, 69

[354] F. Zhang, C. Guan, J. Fang, S. Bai, R. Yang, P. Torr, and V. Prisacariu. Instance segmentation of lidar point clouds. *IEEE International Conference on Robotics and Automation (ICRA)*, 2020. 29, 149, 150, 151, 152, 154, 161

[355] F. Zhang, V. Prisacariu, R. Yang, and P. H. Torr. Ga-net: Guided aggregation net for end-to-end stereo matching. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 60, 61, 62, 63, 64, 66, 71, 72, 73, 74, 76, 78, 79, 80, 85, 89, 92, 93, 94, 98, 99, 100, 193

[356] F. Zhang, X. Qi, R. Yang, V. Prisacariu, B. Wah, and P. Torr. Domain-invariant stereo matching networks. In *European Conference on Computer Vision (ECCV)*, 2020. 85, 89

[357] F. Zhang and B. W. Wah. Supplementary meta-learning: Towards a dynamic model for deep neural networks. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017. 46

[358] F. Zhang and B. W. Wah. Fundamental principles on learning new features for effective dense matching. *IEEE Transactions on Image Processing (TIP)*, 27(2), 2018. 18, 22, 37, 39, 63, 87

[359] P. Zhang, B. Zhang, T. Zhang, D. Chen, Y. Wang, and F. Wen. Prototypical pseudo label denoising and target structure learning for domain adaptive semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 28, 130, 134, 141, 145

[360] Q. Zhang, J. Zhang, W. Liu, and D. Tao. Category anchor-guided unsupervised domain adaptation for semantic segmentation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 140, 141

[361] S. Zhang, S. Yan, and X. He. Latentgnn: Learning efficient non-local relations for visual recognition. *arXiv preprint arXiv:1905.11634*, 2019. 71, 73

[362] X. Zhang and M. Maire. Self-supervised visual representation learning from hierarchical grouping. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 119, 120, 131

[363] Y. Zhang, Y. Chen, X. Bai, S. Yu, K. Yu, Z. Li, and K. Yang. Adaptive unimodal cost volume filtering for deep stereo matching. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2020. 79

[364] Y. Zhang, Z. Qiu, T. Yao, C. Ngo, D. Liu, and T. Mei. Transferring and regularizing prediction for semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 27, 129, 141

[365] Z. Zhang, B.-S. Hua, and S.-K. Yeung. Shellnet: Efficient point cloud convolutional neural networks using concentric shells statistics. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 30, 153

[366] H. Zhao, L. Jiang, C.-W. Fu, and J. Jia. Pointweb: Enhancing local neighborhood features for point cloud processing. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 31, 154

[367] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2, 4, 17, 112, 115, 117, 140

[368] H. Zhao, Y. Zhang, S. Liu, J. Shi, C. Change Loy, D. Lin, and J. Jia. Psanet: Point-wise spatial attention network for scene parsing. In *European Conference on Computer Vision (ECCV)*, September 2018. 31, 154

[369] S. Zhao, Y. Sheng, Y. Dong, E. I. Chang, Y. Xu, et al. Maskflownet: Asymmetric feature matching with learnable occlusion mask. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 4, 22, 87, 102

[370] X. Zhao, R. Vemulapalli, P. Mansfield, B. Gong, B. Green, L. Shapira, and Y. Wu. Contrastive learning for label-efficient semantic segmentation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 108, 131

[371] Y. Zhao, T. Birdal, H. Deng, and F. Tombari. 3d point capsule networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 31, 154

[372] Y. Zheng, M. Zhang, and F. Lu. Optical flow in the dark. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 22, 87

[373] Z. Zheng and Y. Yang. Rectifying pseudo label learning via uncertainty estimation for domain adaptive semantic segmentation. *International Journal of Computer Vision (IJCV)*, 129(4):1106–1120, 2021. 28, 130

[374] Y. Zhong, Y. Dai, and H. Li. Self-supervised learning for stereo matching with self-improving ability. *arXiv preprint arXiv:1709.00930*, 2017. 19, 64

[375] Y. Zhong, P. Ji, J. Wang, Y. Dai, and H. Li. Unsupervised deep epipolar flow for stationary or dynamic scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 23, 87

[376] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba. Scene parsing through ADE20K dataset. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 8, 116

[377] C. Zhou, H. Zhang, X. Shen, and J. Jia. Unsupervised learning of stereo matching. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017. 19, 60, 64, 79, 193

[378] Y. Zhou and O. Tuzel. Voxelnet: End-to-end learning for point cloud based

3d object detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 172, 175, 178, 180, 181, 185, 186, 187, 188, 189

[379] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017. 27, 129

[380] Y. Zhu, Z. Zhang, C. Wu, Z. Zhang, T. He, H. Zhang, R. Manmatha, M. Li, and A. Smola. Improving semantic segmentation via self-training. *arXiv preprint arXiv:2004.14960*, 2020. 109

[381] C. Zhuang, A. L. Zhai, and D. Yamins. Local aggregation for unsupervised learning of visual embeddings. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 109

[382] M. Z. Zia, M. Stark, B. Schiele, and K. Schindler. Detailed 3d representations for object recognition and modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 35(11), 2013. 175

[383] B. Zoph, G. Ghiasi, T.-Y. Lin, Y. Cui, H. Liu, E. D. Cubuk, and Q. V. Le. Rethinking pre-training and self-training. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 26, 108, 109, 131

[384] Y. Zou, Z. Yu, B. V. K. V. Kumar, and J. Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *European Conference on Computer Vision (ECCV)*, 2018. 129

[385] Y. Zou, Z. Zhang, H. Zhang, C.-L. Li, X. Bian, J.-B. Huang, and T. Pfister. PseudoSeg: Designing pseudo labels for semantic segmentation. In *International Conference on Learning Representation (ICLR)*, 2021. 107, 118, 120

[386] S. Zweig and L. Wolf. Interponet, a brain inspired neural network for optical flow dense interpolation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 22, 87