Improved Methods and Analysis for Semantic Image Segmentation

A dissertation submitted towards the degree
Doctor of Engineering
(Dr.-Ing.)
of the Faculty of Mathematics and Computer Science
of Saarland University

by
Yang He, M.Sc.

Saarbrücken
2019

Day of Colloquium	3rd of December, 2019
Dean of the Faculty	Univ.-Prof. Dr. Sebastian Hack Saarland University, Germany

Examination Committee

Chair	Prof. Dr. Joachim Weickert
Reviewer, Advisor	Prof. Dr. Mario Fritz
Reviewer, Co-advisor	Prof. Dr. Bernt Schiele
Reviewer	Prof. Dr. Joachim Denzler
Academic Assistant	Dr. Paul Swoboda

ABSTRACT

Modern deep learning has enabled amazing developments of computer vision in recent years (Hinton and Salakhutdinov, 2006; Krizhevsky *et al.*, 2012). As a fundamental task, semantic segmentation aims to predict class labels for each pixel of images, which empowers machines perception of the visual world. In spite of recent successes of fully convolutional networks (Long *et al.*, 2015), several challenges remain to be addressed. In this thesis, we focus on this topic, under different kinds of input formats and various types of scenes. Specifically, our study contains two aspects: (1) Data-driven neural modules for improved performance. (2) Leverage of datasets w.r.t.training systems with higher performances and better data privacy guarantees.

In the first part of this thesis, we improve semantic segmentation by designing new modules which are compatible with existing architectures. First, we develop a spatio-temporal data-driven pooling, which brings additional information of data (i.e. superpixels) into neural networks, benefiting the training of neural networks as well as the inference on novel data. We investigate our approach in RGB-D videos for segmenting indoor scenes, where depth provides complementary cues to colors and our model performs particularly well. Second, we design learnable dilated convolutions, which are the extension of standard dilated convolutions, whose dilation factors (Yu and Koltun, 2016) need to be carefully determined by hand to obtain decent performance. We present a method to learn dilation factors together with filter weights of convolutions to avoid a complicated search of dilation factors. We explore extensive studies on challenging street scenes, across various baselines with different complexity as well as several datasets at varying image resolutions.

In the second part, we investigate how to utilize expensive training data. First, we start from the generative modelling and study the network architectures and the learning pipeline for generating multiple examples. We aim to improve the diversity of generated examples but also to preserve the comparable quality of the examples. Second, we develop a generative model for synthesizing features of a network. With a mixture of real images and synthetic features, we are able to train a segmentation model with better generalization capability. Our approach is evaluated on different scene parsing tasks to demonstrate the effectiveness of the proposed method. Finally, we study membership inference on the semantic segmentation task. We propose the first membership inference attack system against black-box semantic segmentation models, that tries to infer if a data pair is used as training data or not. From our observations, information on training data is indeed leaking. To mitigate the leakage, we leverage our synthetic features to perform prediction obfuscations, reducing the posterior distribution gaps between a training and a testing set. Consequently, our study provides not only an approach for detecting illegal use of data, but also the foundations for a safer use of semantic segmentation models.

ZUSAMMENFASSUNG

Modernes “deep learning" hat in den letzten Jahren erstaunliche Entwicklungen im Bereich Computer Vision ermöglicht (Hinton and Salakhutdinov, 2006; Krizhevsky *et al.*, 2012). Eine grundlegende Aufgabe der semantischen Segmentierung ist es, labels für jedes Pixel von Bildern vorherzusagen, wodurch die Wahrnehmung der visuellen Welt durch Maschinen verbessert wird. Trotz der jüngsten Erfolge von vollständig faltenden Netzwerken (fully convolutional networks) (Long *et al.*, 2015) müssen einige Herausforderungen noch gemeistert werden. In dieser Arbeit konzentrieren wir uns auf dieses Thema, unter verschiedenen Arten von Eingabeformaten und verschiedenen Arten von Szenen. Unsere Studie enthält insbesondere zwei Aspekte: (1) Datengesteuerte neuronale Module für eine verbesserte Leistung. (2) Nutzung von Datensätzen mit Trainingssystemen mit höherer Leistung und besseren Datenschutzgarantien.

Im ersten Teil der Arbeit verbessern wir die semantische Segmentierung, indem wir neue Module entwerfen, die mit vorhandenen Architekturen kompatibel sind. Zunächst entwickeln wir ein räumlich-zeitliches datengesteuertes Pooling, das zusätzliche Dateninformationen (d. h. Superpixel) in neuronale Netze einbringt, was sowohl dem Training neuronaler Netze als auch der Folgerung auf neue Daten zugute kommt. Wir untersuchen unseren Ansatz in RGB-D-Videos zur Segmentierung von Szenen in Innenräumen, bei denen die Tiefe ergänzende Hinweise zu Farben liefert. Es zeigt sich, dass unser Modell besonders leistungsfähig ist. Zweitens entwerfen wir lernbare erweiterte Faltungen, die die Erweiterung von erweiterten Standardfaltungen darstellen (Yu and Koltun, 2016), deren Erweiterungsfaktoren sorgfältig von Hand bestimmt werden müssen, um eine angemessene Leistung zu erzielen. Wir präsentieren eine Methode, um Dilatationsfaktoren zusammen mit Filtergewichten von Faltungen zu lernen, um eine komplizierte Suche nach Dilatationsfaktoren zu vermeiden. Wir untersuchen umfangreiche Studien zu herausfordernden Straßenszenen über verschiedene Baselines mit unterschiedlicher Komplexität sowie Datensätze mit unterschiedlichen Bildauflösungen.

Im zweiten Teil untersuchen wir den Umgang mit teuren Trainingsdaten. Wir beginnen mit der generativen Modellierung und untersuchen die Netzwerkarchitekturen sowie die Lernpipeline zur Generierung mehrerer Beispiele. Wir sind bestrebt, die Vielfalt der generierten Beispiele zu verbessern, aber die vergleichbare Qualität der generierten Beispiele zu bewahren. Zweitens entwickeln wir ein generatives Modell zur Synthese von Zwischenmerkmalen eines neuronalen Netzwerks. Mit einer Mischung aus realen Bildern und synthetischen Merkmalen können wir ein semantisches Segmentierungsmodell mit einer besseren Generalisierungsfähigkeit trainieren. Unser Ansatz wird anhand verschiedener Aufgaben zum Parsen von Szenen bewertet, um die Wirksamkeit der Vorschlagsmethode zu demonstrieren. Schließlich untersuchen wir die Inferenz der Zugehörigkeit zu einer semantischen

Segmentierungsaufgabe. Wir schlagen das erste Inferenzangriffssystem für die Mitgliedschaft gegen Black-Box-Semantik-Segmentierungsmodelle vor, bei dem versucht wird, zu schließen, dass ein Datenpaar als Trainingsdaten verwendet wird oder nicht. Aus unseren Beobachtungen geht hervor, dass Informationen zu Trainingsdaten tatsächlich undicht sind. Um die Leckage zu mindern, setzen wir unsere synthetischen Funktionen ein, um Vorhersageverschleierungen durchzuführen und die Lücken in der posterioren Verteilung zwischen Training und Testset zu verringern. Folglich bietet unsere Studie nicht nur einen Ansatz zur Aufdeckung der illegalen Verwendung von Daten, sondern auch die Grundlagen für eine sicherere Verwendung semantischer Segmentierungsmodelle.

ACKNOWLEDGEMENTS

First and foremost I would like to express my sincerest gratitude to my advisors Prof. Dr. Mario Fritz and Prof. Dr. Bernt Schiele for their great supports of my PhD study. Their deep academic insights and rich knowledges helped me a lot in the research work during my PhD. More importantly, their attitudes and encouragement will be always inspiring to me in my future work. Besides, working with them, I also learned how to define, discover, address, present and publish my research works, which will be absolutely important in my future research works. In addition, I would like to show my appreciation to my close collaborater: Prof. Dr. Margret Keuper. She always encouraged me to explore some new ideas, selflessly shared her experience and provided her high-quality feedback, which deeply influenced my understanding on good academic activity. Thank you all for your time and effort for reviewing my work and providing valuable feedback.

In the following, I would like to thank people who I have collaborated with: Prof. Dr. Wei-Chen Chiu, Yongqin Xian, Subhabrata Choudhury, Prof. Dr. Zeynep Akata. I also learned a lot from you and growed a lot working with you, from your academic ability or personality. Besides, I would like to thank many other people in the computer vision and machine learning group of MPI-Informatics, including Gerard Pons-Moll, Mykhaylo Andriluka, Rodrigo Benenson, Björn Andres, Paul Swoboda, Andreas Bulling, Anna Khoreva, Shanshan Zhang, Siyu Tang, Qianru Sun, Xuelin Huang, Qiuhong Ke, Jan Hosang, Rakshith Shetty, Max Losch, Mohamed Omran, Eldar Insafutdinov, Xucong Zhang, Wenbin Li, Apratim Bhattacharyya, Tribhuvanesh Orekondy, David Stutz, Julian Steil, Evgeny Levinkov, Hosnieh Sattar, Ning Yu, Bharat Lal Bhatnagar, Verica Lazova, etc. However, the people I would like to show appreciations are not only limited to the mentioned people, but all D2 members. Thanks for your talents and hard working, maintaining excellent academic circumstance. Your every amazing presentations and discussions are memorable to me.

Next, I would like to thank our secretary, Connie Balzert. Thanks a lot for your careful and patient work. You always kindly reminded me of many important issues and gave me a lot of suggestions, which were of great for my life in Saarbrucken. Besides, I would like to thank IT service staff, for their work on providing a stable and wonderful environment.

Finally, I would like to thank my parents. I could not have pursued my PhD without your supports and love.

CONTENTS

1 INTRODUCTION

Semantic segmentation is a fundamental problem in computer vision. The goal of semantic segmentation is to predict a category for each pixel in a defined label space. It has extremely broad real-world applications in various scenarios (Cordts *et al.*, 2016; Geiger *et al.*, 2012; Nathan Silberman and Fergus, 2012; Schwarz *et al.*, 2018; Menze *et al.*, 2015; Gong *et al.*, 2017). For instance, semantic segmentation systems are core components of autonomous vehicles, providing perceptions of environments (Cordts *et al.*, 2016) or drivable areas (Yu *et al.*, 2018). In robotic applications, semantic segmentation also plays a key role in manipulation of tools or navigation in indoor environments (Nathan Silberman and Fergus, 2012; Schwarz *et al.*, 2018). Benefiting from the development of machine learning that learns visual representations with large-scale image datasets (Deng *et al.*, 2009; Zhou *et al.*, 2018) as well as deep neural networks (Hinton and Salakhutdinov, 2006; LeCun *et al.*, 1998; Krizhevsky *et al.*, 2012; He *et al.*, 2016), the performance of semantic segmentation models has been dramatically improved.

Modern semantic segmentation models are built on fully convolutional architectures (Long *et al.*, 2015; Badrinarayanan *et al.*, 2017; Chen *et al.*, 2018b), which stack continuous convolution blocks into a network to produce structural predictions. They are trained by back-propagating errors (Rumelhart *et al.*, 1988) from pixel-level ground truth annotations, as shown in Figure 1.1. In particular, dilated convolution (Yu and Koltun, 2016; Chen *et al.*, 2018b) has been widely applied for semantic segmentation, which maintains high resolution feature maps as well as large receptive fields at the same time. In addition, modeling context and post processing are also essential to improve the fully convolution pipeline. Previous literature has proposed many solutions to effectively aggregate context information, which either introduce global context (Zhang *et al.*, 2018a; Liu *et al.*, 2015), or multiscale context features (Chen *et al.*, 2018c; Zhao *et al.*, 2017). Therefore, effective context modelling is essential for better recognition and segmentation, which provides critical evidences to determine the category of each location. Conditional random fields (Krähenbühl and Koltun, 2011) are a kind of widely used post processing technique for semantic segmentation, which can correct wrong predictions by considering their context relationships.

Data and model play critical roles in training a segmentation model, in terms of achieving decent performances as well as making limited annotation efforts from humans. Obviously, large-scale annotated data helps expanding the performance limitation of a network (Sun *et al.*, 2017). To obtain powerful segmentation models, academic and industry communities have created many successful datasets for different application scenarios, such as indoor scenes (Silberman *et al.*, 2012), street scenes (Cordts *et al.*, 2016), natural outdoor scenes (Zhou *et al.*, 2017), objects (Lin

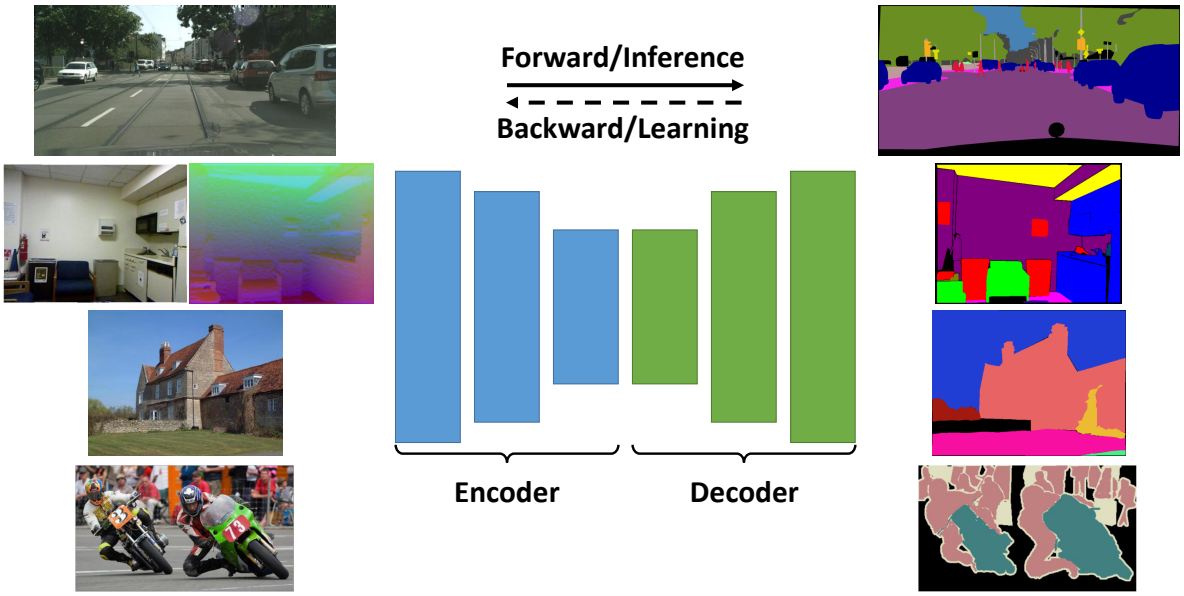


Figure 1.1: The pipeline of modern fully convolutional encoder/decoder architectures for semantic segmentation. It is able to take different input forms (RGB or RGB-D images) and produce dense structural predictions.

et al., 2014), etc. Even though large-scale densely annotated datasets have been released, how to make fully use of those data is still worth to explore, by developing data-driven models (e.g. fully convolutional networks) to train a segmentation model. Recently, except annotating real images, applying synthetic images has became more and more popular, where images and corresponding annotations are automatically acquired from a game engine (Richter *et al.*, 2016; Ros *et al.*, 2016). However, it is challenging to directly utilize synthetic images to train a segmentation model with a surpassed performance compared to a model trained on real images, due to domain shift between real and synthetic images, such as the shape of objects themselves, the ratio between different kinds of objects, as well as the appearances of objects. Those distribution differences tremendously limit the application of synthetic images. Consequently, effective leverage of synthetic images is still an open problem in exploring the performance limitation of semantic segmentation.

Furthermore, generative modelling is becoming an increasingly hot topic, benefiting from the development of neural networks. In particular, Varitional Auto-Encoders (VAEs) (Kingma and Welling, 2013) and Generative Adversarial Nets (GANs) (Goodfellow *et al.*, 2014) contribute to model data distributions with several attractive properties. First, it allows to control the synthesis procedure with a latent variable, and all the generated examples are able to create a manifold over the variable. Second, current generative models are able to synthesize examples with rich details, which are realistic enough to fool a human. In spite of the success of current generative modelling (Isola *et al.*, 2017; Wang *et al.*, 2018b; Kingma and Welling, 2013), there is still very few work showing that synthetic data from a generative model

can be applied as training data to acquire a classification model or a segmentation model. Therefore, it is interesting to study if we are able to extend the application of generative modelling for training a model on a specific task, such as semantic segmentation.

As discussed, data is quite important and rather expensive. Therefore, protecting the ownership of data is crucial for companies or organizations. Besides, data records some sensitive information which is related to personal privacy or has potential security risks. Membership inference attacks aim to determine if a given data sample has been used to train a model or not. From previous research (Shokri *et al.*, 2017), a classification model suffers from membership inference attacks. Hence, an important question is naturally asked: is there also similar data leakage happened for a semantic segmentation model, whose annotations are more expensive? Further, to provide safe services at high recognition rates, suitable usage of data with security mechanism are also crucial.

The goal of this thesis is to study semantic segmentation in terms of neural network architectures as well as leverage of datasets w.r.t.training systems with higher performances and better data privacy guarantees. Our models are built on the top of the fully convolutional network, which is the current state-of-the-art pipeline.

The first part of our work is to design data-driven models to improve semantic segmentation models. Even though convolution networks have numerous parameters and are learned from data, we are more interested in incorporating more priors from data and develop more flexible neural network architectures for improving semantic segmentation. We first investigate the combination between superpixels and neural networks in indoor scenes. Superpixels have a long history and many applications in computer vision. We show that superpixels are able to introduce adaptive pooling regions in neural networks, which helps networks to produce semantic predictions with better recognition accuracies and more precise boundaries. We further study the key component of fully convolutional networks, dilated convolution operations. We regard the dilation factor of a convolution as a learnable parameter, instead of a manually set hyperparameter before training. The dilation factor is learned with filter weights together, by propagating the errors. It helps us to set dilation factors easier by adjusting a proper dilation factor for different datasets and network architectures. To test our approach for learning dilation factors, we study semantic segmentation on street scenes, which contains objects and regions at varying sizes in a same image.

The second part of this thesis is to explore more effective and safer utilization of training data. We first study the conditional generative modelling, which is a reverse problem of semantic segmentation. We propose a stochastic regression model for improving the diversity of generated examples, which alleviates the mode collapse problem of generative modelling. Specifically, our network allows sampling novel examples with new latent codes, and thus we are able to synthesize examples from different distribution clusters with a same condition. Second, we study generative modeling of CNN features, which needs different generator architectures compared

to generating images. Unlike synthetic images, we show our synthetic features can be applied as additional training data for augmenting a semantic segmentation model, which achieves better performance on street scenes and natural scenes after applying our data augmentation technique. Last, we study membership inference attacks on black-box semantic segmentation models, and defense mechanisms to protect membership privacy. From our study, we show information leakage is even more pronounced than classification, that a predicted mask already leaks its membership.

1.1 CONTRIBUTIONS OF THE THESIS

The key contributions of the thesis are two main aspects: (1) improving semantic segmentation with data-driven networks; (2) data synthesis and its applications to semantic segmentation. First of all, we try to improve semantic segmentation by developing data-driven modules which are compatible with existing network architectures. Besides, we study the problem of conditional data synthesis, which is a reverse problem to semantic segmentation, and then shows several applications for semantic segmentation, especially with respective to protecting membership privacy. In the following, we detail the involved challenges in these tasks, as well as the solutions and contributions of this thesis.

1.1.1 Improving semantic segmentation with data-driven networks

The first goal of the thesis is designing new specific modules to improve the accuracy of semantic segmentation models under various settings. Particularly, we study the semantic image segmentation problems on single images, as well as auxiliary images from a video.

1.1.1.1 *Semantic segmentation from partially annotated videos*

Challenges. Multi-view appearances offer richer information comparing to single-view images. In this setting, we are interested in improving single-view segmentation with multi-view information. Videos contain continuous image sequences, which naturally provide multi-view information for the objects and entire environments with different level of context. Therefore, effectively utilizing videos is important to improve semantic segmentation. Correspondences between frames of a video refer to different pixels or regions belonging to a same thing or stuff, which allows summarizing information of all the frames and training a model with less annotations.

A video sequence is able to be a very long sequence, hence, the error of the correspondence could be accumulated, because correspondence between neighboring frames is not perfect due to potential illumination changes, drastic appearance variations, etc. One alternative solution is to leverage frames close to a target frame, however, those frames usually have similar view point to the target frame, and provide similar appearances of things and stuffs. Consequently, the first challenge is

how to construct trustworthy correspondences, which help integrating information between different frames. Besides, traditional neural networks are not able to deal with correspondences as additional inputs for semantic segmentation. Hence, second challenge is to design specific neural architectures to process images as well as correspondences.

Contributions. We propose a new network module which projects multi-view information to the target frame with their correspondences, namely Spatio-Temporal Data-Driven Pooling (STD2P). We propose to utilize region-level correspondences instead of pixel-level correspondence, which allows to reject poor superpixels or correspondences through a video. Superpixels (Pont-Tuset *et al.*, 2017; Stutz *et al.*, 2018) provide useful boundaries as well as pooling regions for a semantic segmentation. The proposed module contains two steps: data-driven spatial pooling and data driven temporal pooling, which introduce superpixels and their correspondences into a network for semantic segmentation. In particular, the spatial pooling can be employed along to perform single-view semantic segmentation, and the STD2P enables a network to handle image sequences with arbitrary lengths. Consequently, we provide an effective solution for improved semantic segmentation from videos with minimum efforts of human annotations.

1.1.1.2 *Learning dilation factors for convolutions*

Challenges. Dilated convolutions (Yu and Koltun, 2016; Chen *et al.*, 2018b) are widely utilized operations in semantic segmentation. We aim to relax the dilation factor in convolutions to a learnable parameter, instead of a fixed hyperparameter, which is manually set before training procedure starts. In order to learn the dilation factors together with their relevant filter weights, the dilation parameters have to be continuous differentiable, that can compute the gradients for the dilation factors and update with back propagation. Besides, another challenge is towards the processing of dilation ranges. Unlike filter weights, dilation factors must be the positive values, and we do not expect those factors too large, because input images have limited sizes. Hence, suitable processing of ranges is necessary for learning dilation factors.

Contributions. We present an alternative convolutions that learns multiple dilation factors for individual channels from data. The proposed module helps avoiding to set dilation factors manually for every datasets, but are learned with filter weights together by back propagation. Besides, our convolution module is able to capture multiscale features from a feature input, which is compatible with existing architectures and can be used to replace traditional convolutions in widespread convolutional neural networks.

1.1.2 Data synthesis and its applications

Next, we present our work on data systhesis and its applications to semantic segmentation. Specifically, we focus on conditional feature systhesis and applying synthetic feature as training data to improve accuracy and protect membership privacy via prediction obfuscations in semantic segmentation.

1.1.2.1 *Conditional image synthesis*

Challenges. Conditional image systhesis aims to generate realistic images from a conditional input, which can be different formats. The task is a one-to-many mapping problem, because it often has many suitable images matched well with the input. On one hand, every generated image is supposed to contain natural details and feasible structures. This challenge has been greatly improved by recent developments of generative modelling (Goodfellow *et al.*, 2014; Chen and Koltun, 2017; Kingma and Welling, 2013; Johnson *et al.*, 2016). However, on the other hand, it is still challenging to cover the distribution of all the training data, due to the so-called mode collapse problem. As a result, improving diversity becomes another goal and challenge of our research.

Contributions. We presented a stochastic regression model for image generation based on proposed depthwise dropout. Besides, we propose a new learning pipeline, that we train with dropout patterns at the beginning of training, and apply a new dropout pattern during inference. Finally, we propose a neighbor enhanced loss function to further improve the diversity of generated images, achieving comparable quality of generated images.

1.1.2.2 *Adversarial feature synthesis*

Challenges. Generating multiple diverse features is challenging, because they encode information of large areas as well as details, which cannot be ignored. Also, the synthetic features should follow a similar distribution as extracted real features. Different to image synthesis, the output of synthesized features have lower spatial dimension and a larger number of channels. Consequently, it is hard to apply existing image synthesis architectures to the feature generation task, and thus a new effective architecture is needed. A good feature generator allows us to sample multiple diverse features from one semantic mask input, and thus provides us numerous training examples. Second, synthetic features should follow a similar distribution as extracted features from real images, which are recognizable by a classification or segmentation model, similar to real features. The final challenge is that raw images contain many detailed information which are compressed in the feature domain, and thus a successful architecture should be powerful enough to model those important details.

Contributions. We propose to synthesize convolution features for data augmentation for semantic image segmentation, leading to improved results. We present an effective generative model for synthesizing features, whose effectiveness is shown according to a series of ablation studies. Several techniques are proposed to leverage the synthetic features, including online hard negative mining, generation from additional masks, and label smoothing regularization.

1.1.2.3 *Security issues in semantic segmentation*

Challenges. Membership inference attacks aim to recognize if a data sample is used as training data or not. Different to general classification on an input data point, semantic segmentation models produce structural predictions. As a result, attacks against a segmentation model need a different pipeline to previous ones for classification (Shokri *et al.*, 2017; Salem *et al.*, 2019). Besides, structures might leak information of training data, even when no posteriors are provided. Therefore, it potentially increase the challenge of protecting membership privacy. Last but not least, there exists a tradeoff between membership privacy and segmentation accuracy. It is challenging but important to protect membership privacy while preserving the utility of semantic segmentation.

Contributions. We present the first work on membership inference attacks against semantic segmentation models. From our study, we show information leakage is widely existing on black-box models for semantic segmentation task under various attacking conditions. In addition, we propose a prediction obfuscation method based on synthetic features for protecting membership privacy, and provide systematic analysis on series of defense techniques.

1.2 OUTLINE OF THE THESIS

In this section we summarise each chapter of the thesis. In addition, we also indicate the respective publications and connections with other previous works.

Chapter 2: Related Work. In this chapter, we review the related works on semantic segmentation with deep convolutional networks, generative modeling of conditional image generation, and membership inference attacks against machine learning models. We analyse the relations of previous and subsequent works to the research presented in this thesis.

Chapter 3: Indoor Scene Understanding. This chapter presents spatio-temporal data-driven pooling (STD2P) for indoor scene understanding. We focus on Kinect captured images with additional depth modality in indoor scenes. We apply the RGBD superpixels with decent performances into a segmentation network, and study semantic segmentation from multiple-view images as well as single-view images. We observe that indoor scenes provide images from many different

viewpoints for a same object, and thus our solution is quite suitable to improve semantic segmentation in indoor scenes.

The content of this chapter corresponds to the CVPR 2017 publication "STD2P: RGBD Semantic Segmentation using Spatio-Temporal Data-Driven Pooling" (He *et al.*, 2017a). Yang He was the lead author of this paper.

Chapter 4: Street Scene Understanding. In this chapter, we study street scene understanding from a single image. We generalize widely used dilated convolutions to a learnable version, to learn a suitable dilation factor in convolutions from data. In street scenes, there are different objects with various scales in a same view, such as pedestrians, cars, trains and buildings etc. To handle those objects with broad scales, we develop depthwise learnable dilated convolutions with our basic learnable version, to capture wider contexts as well as local details. Eventually, we show improved segmentation results for street scenes with our alternative convolutions.

The content of this chapter corresponds to the GCPR 2017 publication "Learning Dilation Factors for Semantic Segmentation of Street Scenes" (He *et al.*, 2017b). Yang He was the lead author of this paper.

Chapter 5: Conditional Image Generation. In this chapter, we design a new stochastic module depthwise dropout as well as a neighbor enhanced loss function for conditional image synthesis. We investigate our method and compare to previous methods on human face systhesis from landmarks and animal head systhesis from normal maps. Our goal is to generate multiple different outputs from a input condition, that all the generated outputs are compatiable with the input. We not only demonstrate successful generated images, but also provide a series of studies with respective to quantifying the diversity and accuracy of generated images, which shows the effectiveness of proposed method.

The content of this chapter corresponds to the ECCV 2018 publication "Diverse Conditional Image Generation by Stochastic Regression with Latent Drop-Out Codes" (He *et al.*, 2018). Yang He was the lead author of this paper.

Chapter 6: Adversarial Feature Generation. In this chapter, we present our generative model for synthesizing dense features, and apply our synthetic features as part of training data to perform data augmentation for semantic segmentation. We formulate the feature synthesis as a condition generation problem, where translate a semantic layout into a convolution feature inside a network. We compare our synthetic features based augmentation pipeline to synthetic images based, and show synthesizing features is a more feasible and effective solution. We also conduct the study of synthesizing features from different layout sources to show further improvement.

The content of this chapter corresponds to the work "DFGAN: Synthetic Dense Features for Improved Semantic Segmentation" (He *et al.*, 2019a). Yang He was the lead author of this paper.

Chapter 7: Segmentations Leak. In this chapter, we show the second application of sythetic features presented in chapter 6, with repective to protecting the membership privacy of a segmentation model. First of all, we study the membership inference attacks against a black-box semantic segmentation model. In the black-box setting, attackers can only intract with a segmentation model by querying an image and obtaining the returned posteriors. In the following, we study a set of defense techniques for protecting membership privacy, particularly on the prediction obfuscations with synthetic features.

The content of this chapter corresponds to the work "Segmentations-Leak: Membership Inference Attacks and Defenses in Semantic Image Segmentation" (He *et al.*, 2019b). Yang He was the lead author of this paper.

Chapter 8: Conclusions and Future Prospects. In this chapter we summarize the thesis and discuss possible future research directions for semantic segmentation and privacy issues of this task.

PUBLICATIONS

[5] *STD2P: RGBD Semantic Segmentation Using Spatio-Temporal Data-Driven Pooling.*
Yang He, Wei-Chen Chiu, Margret Keuper, and Mario Fritz.
In Proc. IEEE Conf. on Computer Vision and Pattern Recognition (**CVPR**), 2017.

[4] *Learning Dilation Factors for Semantic Segmentation of Street Scenes.*
Yang He, Margret Keuper, Bernt Schiele, and Mario Fritz.
In Proc. German Conf. on Pattern Recognition (**GCPR**), 2017.

[3] *Diverse Conditional Image Generation by Stochastic Regression with Latent Drop-Out Codes.*
Yang He, Bernt Schiele, and Mario Fritz.
In Proc. European Conf. on Computer Vision (**ECCV**), 2018.

[2] *Segmentations-Leak: Membership Inference Attacks and Defenses in Semantic Image Segmentation.*
Yang He, Shadi Rahimian, Bernt Schiele, and Mario Fritz.
In Submission, 2019.

[1] *DFGAN: Synthetic Dense Features for Improved Semantic Segmentation.*
Yang He, Bernt Schiele, and Mario Fritz.
In Submission, 2019.

RELATED WORK 2

Semantic segmentation is a basic research topic in computer vision with a long history and great progress has been enabled through deep learning in recent years. Besides, generative modeling has profited from recent development of convolutional neural networks (CNN) (LeCun *et al.*, 1998) as well. Recently, membership inference attacks against machine learning models draw much attention on the widespread security risks of deep neural networks. Therefore, we give an overview of related work in this chapter, focusing on the directions explored in this thesis.

This chapter is organised as follows. We first review previous works on semantic segmentation with convolutional neural networks based pipeline in Section 2.1. We then make a tour of recent development of generative modeling, particularly on conditional generative models in Section 2.2. Last but not least, we give a brief summary on recent approaches for membership inference attacks against machine learning models in Section 2.3.

2.1 SEMANTIC SEGMENTATION WITH NEURAL NETWORKS

In this section, we review semantic segmentation research. We first introduce the current pipeline for semantic segmentation based on fully convolutional neural networks in Section 2.1.1. Then, we present different component choices for semantic segmentation in neural networks. Particularly, we discuss convolutions and its variations in Section 2.1.3, which inspire us to develop our learnable dilated convolutions in Chapter 4. We compare different pooling strategies for network backbones as well as segmentation models in Section 2.1.4, which is related to our spatio-temporal data-driven pooling in Chapter 3. We compare supervisions for training a network in Section 2.1.5, where we apply different loss functions for our synthetic features in Chapter 6. Last, we discuss existing datasets with different types and capacities of provided data in Section 2.1.6, as used in our experiments of Chapter 3, 4, 6 and 7.

2.1.1 Pipeline of fully convolution networks

The rise of deep learning significantly speeds up the development of computer vision. With modern GPU acceleration for parallel computing for the deep architectures, successfully training of a network on large-scale labeled datasets becomes possible. More importantly, the pretraining of a network on large-scale datasets is quite helpful for other computer vision tasks, which provides learned visual representations. For example, ImageNet (Deng *et al.*, 2009) and Places (Zhou *et al.*,

2018) offer a network to learn the representations for objects and scenes respectively. Consequently, the current computer vision pipeline contains two steps: (1) Design a network architectures and train them on large-scale datasets. (2) Use a pretrained network as a backbone, and add some modules for specific tasks, such as object detection (Ren *et al.*, 2015), and semantic segmentation (Long *et al.*, 2015). As shown in Figure 2.1, in the first step, networks are trained with image-level annotations, for which is feasible to collect a very large dataset, providing enough training data for deep networks to learn powerful visual features. Second, in semantic segmentation, a pretrained network is finetuned with dense pixel-level annotations, which is able to predict a class label for each location of an input image.

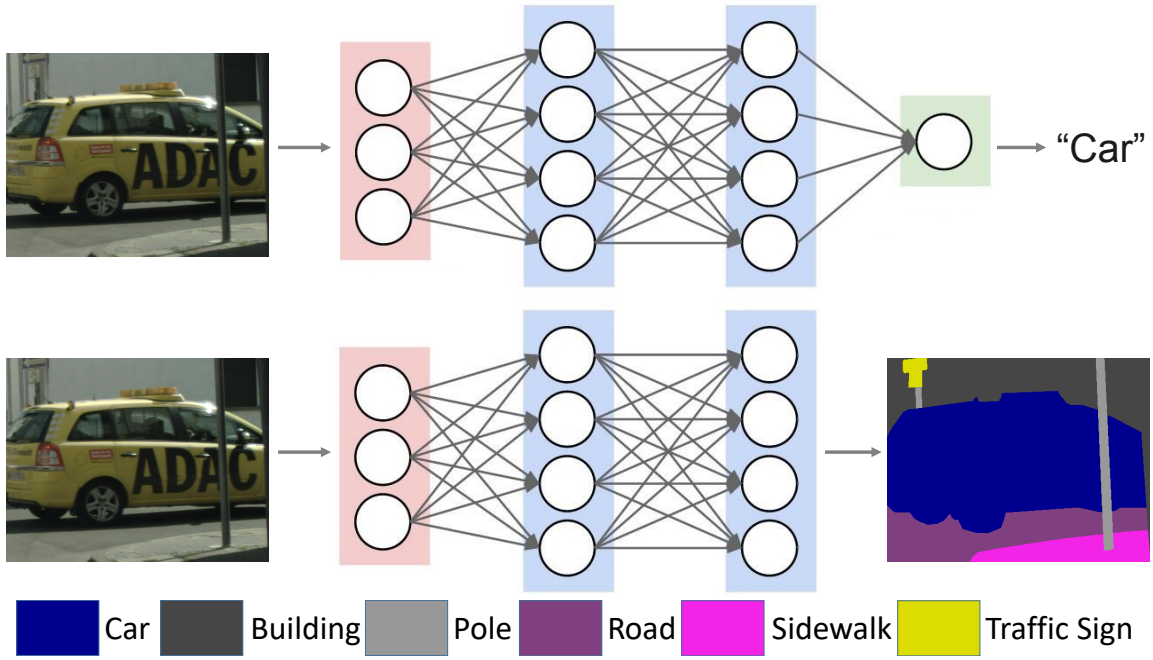


Figure 2.1: Current neural networks based pipeline for training a semantic segmentation model.

For semantic segmentation, fully convolutional networks (Long *et al.*, 2015; Chen *et al.*, 2018b; Badrinarayanan *et al.*, 2017) are the base architecture, which is comprised of several convolution blocks only. To keep the dense structures, fully connected layers are removed, which are widely used in image classification. It takes an image as input, and emits a dense output, assigning the categories of various images locations. Each convolution block contains convolution, normalization, pooling, and nonlinear activation. After several convolution blocks, spatially compressed features are obtained. Final high resolution predictions are achieved by upsampling (Chen *et al.*, 2018b) or deconvolution operations (Long *et al.*, 2015). Besides, to overcome losing the information of small objects, deconvolution networks (Noh *et al.*, 2015) learn multiple deconvolution operations on the top of fully convolutional architectures.

In the following sections, we discuss more details for semantic segmentation. Besides, there are also post-processing operations or augmentation techniques applied for refinement or improvement. Conditional random fields (Lafferty *et al.*, 2001; Krähenbühl and Koltun, 2011) are widely used to refine and smoothen the prediction from CNN, as first proposed in previous work (Chen *et al.*, 2018b). Flipping and multi-scale inputs are able to significantly improve original prediction of CNN, which are also utilized in training stage.

Backbones. As the base of network model for specific tasks, convolutional architectures play an important role in achieving decent performance at acceptable speed. In the following, we discuss the progress of developments in network architectures, which are used in Chapter 3, 4, 6 and 7. AlexNet (Krizhevsky *et al.*, 2012) is the first successful model on large-scale dataset, which learns useful visual representations. Furthermore, networks are designed to be deeper, and VGGNet (Simonyan and Zisserman, 2014) replaces the convolution kernels in AlexNet with 3×3 kernels, and attaches more convolution layers with more parameters to fit better on the large-scale dataset and obtain stronger representations. GoogleNet (Szegedy *et al.*, 2015) proposes an inception module which integrates multi-scale features from multiple paths with different sizes of convolution kernels.

In addition, to design very deep neural networks, ResNet (He *et al.*, 2016) proposes residual connections between convolution blocks and further increases the fitting capability on data, by observing simply attaching convolutions cannot bring improvements, and thus learning residual is necessary. With residual connections and very deep architectures, superior representations are learned, leading to significant improvements. In the following, DenseNet (Huang *et al.*, 2017) builds a network with more connections for dense blocks instead of residual blocks, which shows stronger ability in some tasks. Recently, research on neural architecture search (Zoph and Le, 2017; Zoph *et al.*, 2018) shows some stronger network architectures by searching the hyperparameters within a predefined architecture space with reinforcement learning.

Last, to fulfill the requirement of real-time and computation limited applications, some simplified backbones are proposed. MobileNet (Howard *et al.*, 2017) applies depthwise convolutions in constructing convolution layers of a network, which extremely reduce the parameters for a convolution layer, whereas only little decrease in performance is happened. Furthermore, ShuffleNet (Zhang *et al.*, 2018c) proposes pointwise group convolutions and achieves comparable performance to AlexNet while computing at a $13\times$ faster speed on mobile devices.

2.1.2 Superpixels for segmentation:

Superpixels have a long history research and application in semantic segmentation. Beyond traditional superpixels, region proposals aim to provide pre-segmetation at object level with semantics instead of focusing on low level colors only. In the following, we discuss two aspects: (1) Traditional methods of superpixels, region

proposals and applications in semantic segmentation. (2) Modern deep learning based methods with superpixels, which is relevant to our work in Chapter 3.

A comprehensive analysis on superpixels and region proposals can be viewed in previous literature reviews (Neubert and Protzel, 2012; Stutz *et al.*, 2018; Hosang *et al.*, 2015). Instead of discussing all the methods in the history, we presented most popular and relevant superpixel methods to recent and our work. SLIC (Achanta *et al.*, 2012) proposed a fast superpixel method, which receives much attention because of its simplicity. SLIC calculates local K-means clustering to generate a segmentation with K superpixels. Therefore, it is able to group the pixels with similar colors into a superpixel. To overcome the drawbacks of only considering low-level information, region proposals aims to provide an intial segmentation with object-level superpixels. Selective search starts by over-segmenting an image based on intensity of the pixels using a graph-based segmentation method by Felzenszwalb and Huttenlocher (2004), for high recall of interesting objects. Multiscale combinatorial grouping (MCG) (Pont-Tuset *et al.*, 2017) is a bottom-up hierarchical image segmentation method combining multi-scale information, built on the fast normalized cuts algorithm. MCG first computes a boundary probability map, and then performs normalized cuts to output several regions. Furthermore, MCG has been improved by providing stronger boundary probability maps. Gupta *et al.* (2014) propose to leverage additionally depth to enhance MCG for indoor scene. which is applied in this thesis of Chapter 3. Convolutional oriented boundary improves boundary maps by incorporating estimated boundaries from different orientations with a neural network (Maninis *et al.*, 2016).

To begin with, superpixels have been utilized in semantic segmentation with traditional methods based on hand-crafted features. Gould *et al.* (2008) regard an over-segmented image as a graph and modeled it with conditional random fields. This method leverages a set of appearance features to represent each superpixel, including RGB colors, Lab colors and textures etc. Strassburg *et al.* (2015) analyze the influence of superpixel methods for semantic segmentation with hand crafted features. Particularly, semantic segmentation with superpixels on multi-modal data has been studied, by kernel descriptors for RGB-D images and Markov random fields (Ren *et al.*, 2012), as well as 3D features based conditional random fields with two different sources of information (Cadena and Košecká, 2014). Except directly applying graphical models over superpixels, SuperParsing (Tighe and Lazebnik, 2010) first retrieves similar images with global appearances from a dataset, and then produces a final output by combining the semantics of all the retrieved images.

Superpixels were also applied with CNN learned features. Farabet *et al.* (2012) leverage multi-scale learned hierarchical features from a CNN for scene labeling on pre-computed superpixels. Couprie *et al.* (2013) extend this pipeline to RGB-D data and applies temporal aggregation of surrounding frames of a video with optical flow. Gupta *et al.* (2014) learn rich features for RGB-D with bounding box annotations, and applies the learned features over superpixels for semantic segmentation, which is relevant to our approach in Chapter 3. Additionally, hypercolumns (Hariharan *et al.*, 2015) proposes to leverage multi-stage features from a learned CNN and projects

them into superpixels by averaging individual predictions.

Finally, in the research of introducing superpixel information into end-to-end modern neural networks, there are several highly relevant works. Bilateral inception (Gadde *et al.*, 2016) designs a filtering module over superpixels, which computes SLIC superpixels and performs bilateral filtering like operations over neighboring superpixels, passing the information to each other. Besides, the region-based end-to-end segmentation model (Caesar *et al.*, 2016) proposes to introduce overlapped selective search superpixels by max pooling into a network, and train the network with back-propagation in an end-to-end manner. Furthermore, a message passing procedure over superpixels has been explored in semantic segmentation (Lin *et al.*, 2017). Besides, this method generates discrete depth images to control different parts of processed images and predicts with different branches separately. Last, superpixels have also been applied in weakly supervised semantic segmentation where only image-level labels are provided (Kwak *et al.*, 2017).

Different to previous work, our approach is simple and effective to introduce superpixels into networks to perform refinement, summarizing information of a superpixel with pooling operation. Besides, our solution allows us to leverage unlabeled temporal data and thus leading a semi-supervised learning framework, by establishing region correspondences over time.

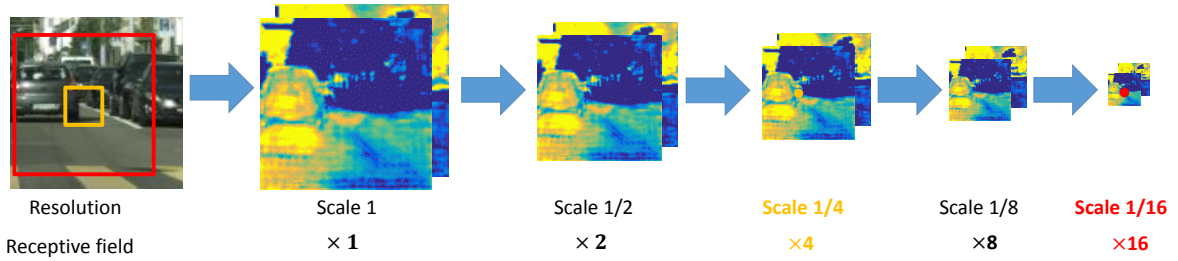
2.1.3 Convolution operations:

Convolution is the basic operation of a CNN (LeCun *et al.*, 1998), which aims to learn a group of convolution filters and apply those on inputs for output activations. Formally, the convolution operation in CNN filters an input $X \in \mathcal{R}^{K_1 \times H \times W}$ with kernels $W \in \mathcal{R}^{K_1 \times K_2 \times k \times k}$, which contains $K_1 \times K_2$ filters with size $k \times k$. As a result, the operation produces an output $Y \in \mathcal{R}^{K_2 \times H \times W}$, where the i-th slice of the output is the sum of all the filtered results with corresponding filters in W.

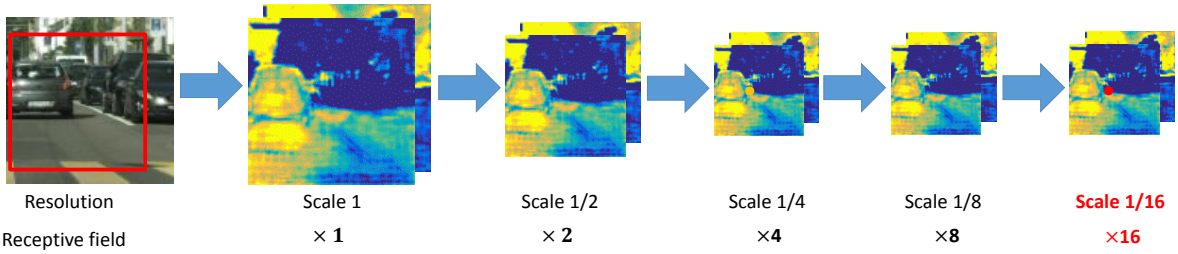
In the following, we discuss several variations to the standard convolution, related to our work in Chapter 4, mainly including dilated convolution and its variations, simplified convolutions and dynamic convolutions.

Dilated (Atrous) convolution. Dilated convolution(Yu and Koltun, 2016; Chen *et al.*, 2018b) has been broadly used in semantic segmentation particularly. It introduces an extra hyperparameter, i.e. dilation factor, for sampling input activations at different locations. Instead filtering inputs with dense kernels, a dilated convolution normally samples farther activations as inputs with the same number of parameters in convolution filters.

In real-world images, there are many regions at small scale, therefore, CNN with pooling operations might ignore those regions due to its downsampling fact. However, pooling operations allow network to learn visual representations at a large receptive field, which is important for recognizing large objects. Consequently, large receptive fields as well as keeping small regions are both necessary to address the challenges in semantic segmentation. Then dilated convolution is able to sample



(a) Pipeline with pooling operations



(b) Pipeline with dilated convolutions

Figure 2.2: Comparison of semantic segmentation pipelines. (a) Pooling operations reduce the resolution of feature maps. (b) Dilated convolutions help keeping high resolution feature maps.

wider CNN activations and thus learn visual features at a large receptive field, while preserving the spatial resolution of feature maps in a CNN model. However, in dilated convolutions, the dilation factor is a hyperparameter defined by users. To avoid manually set dilation factors, we present an approach to learn dilation factors from training data in Chapter 4.

Based on dilated convolutions, there are several effective modules in semantic segmentation, as presented in the following paragraph.

Hybrid dilated convolution. Atrous spatial pyramid pooling (ASPP) (Chen *et al.*, 2018b) integrates multi-scale features by calculating the features from convolutions with different dilation factors, which is able to capture different levels of context information. Besides, instead of applying kernels with square shapes, dense prediction cell (Chen *et al.*, 2018a) develops an ASPP-like module with irregular rectangle kernels. As pointed out in previous work (Wang *et al.*, 2018a; Yu *et al.*, 2017), dilated convolution tends to produce feature maps of final outputs with regular holes. To handle this phenomenon, hybrid dilated convolutions are proposed, which equips several convolutions with complementary dilation factors sequentially.

Different to previous work, our solution in Chapter 4, channelwise dilated convolutions, integrates multi-scale features by learning individual dilation factors for each channel.

Simplified large kernel convolution. Kernel size is an important hyperparameter in convolutions. A Large kernel is able to capture a wider context but introduces

more parameters and thus suffers from heavy computations. Simplified large kernel convolution (Peng *et al.*, 2017) decomposes a large kernel into the multiplication between two small kernels, which shows improved semantic segmentation compared to employing large kernels.

Dynamic convolutions. In standard convolution, once the parameters, i.e. filter weights, are learned after training stage, it becomes fixed during inference. Dynamic convolutions aim to learn to generate filter weights or other parameters for convolutions. In other words, it applies different filters for varying inputs. As an example, dynamic filtering network (Jia *et al.*, 2016) proposes to learn a filter generating network in video prediction and stereo prediction task. Deformable ConvNet (Dai *et al.*, 2017; Zhu *et al.*, 2019) learns spatial bias for convolution to handle widely existing object deformations in semantic segmentation and object deformation.

2.1.4 Pooling operations

As a basic operation, pooling is used to aggregate and extract local informative features. Max pooling and average pooling are most frequent operations in CNN, which return the maximum or average values inside a sliding window. Normally, pooling operations are performed with combination of downsampling, which enlarges the receptive field of a neural network, as shwon in Figure 2.2 (a). Besides, previous work (Springenberg *et al.*, 2014) also shows simply downsampling is able to replace max or average pooling in CNN, and achieves comparable performance. Generalized pooling (Lee *et al.*, 2016) shows superior performance over max or average pooling, which is based on a tree structure or linear combination, which combines max pooling and average pooling.

For semantic segmentation, global pooling shows to be quite important to integrate context information by several models (Liu *et al.*, 2015; Zhao *et al.*, 2017; Chen *et al.*, 2018c). Global pooling computes global context statistics and provides image-level features. Particularly, pyramid spatial pooling(Zhao *et al.*, 2017) extends global pooling with respective to providing different levels of context by pooling operations with various kernel sizes.

Observing above approaches, all of them employs regular pooling region defined by users, while our pooling regions are computed from data itself, leading to a new effective model architecture in Chapter 3.

2.1.5 Supervisions

A semantic segmentation model produces dense outputs, which is trained with supervision from pixel-wise dense annotations (Long *et al.*, 2015), that each location provides a classification loss to compute the gradients for the parameters of a CNN model. Except dense supervisions, discrete pixel-level supervisions also show effectiveness in training a semantic segmentation model (Bansal *et al.*, 2017). Deep

neural networks are trained with softmax loss for classification, which aims to minimize the cross-entropy error, i.e. negative log-likelihood for the ground truth categories. Formally, let $r \in \mathbb{R}^{I \times J \times K}$ be the prediction scores for an input image X, with resolution of $I \times J$ and K classes. The per class probability at the (i, j)-th location ($1 \leq i \leq I, 1 \leq j \leq J$) for the k-th category is calculated by

$$p_{ij}(k|X) = \frac{\exp(r_{ij}^k)}{\sum_{k=1}^{K} \exp(r_{ij}^k)} \tag{2.1}$$

for each label $k \in \{1, \cdots, K\}$, where r_{ij}^k is the unnormalized log probabilities of the k-th class at (i, j)-th location. Finally, a segmentation model is learned by optimizing the following objective

$$\min_{\theta} \sum_{i=1}^{I} \sum_{j=1}^{J} p_{ij}(k|X; \theta). \tag{2.2}$$

In this thesis, we apply dense pixelwise softmax loss function to train a semantic segmentation model, in case we do not have any other explicit explanations.

Further, it is hard to avoid annotation noises because of ambiguous examples. Therefore, to alleviate overfitting on the potential noisy labels and improve generalizations, label smoothing regularization (LSR) (Szegedy *et al.*, 2016) is applied on softmax, which encourages a network to produce less confident predictions. In Chapter 6, we apply LSR to train a segmentation model with imperfect synthetic features, which is a part of our data augmentation strategy for semantic segmentation.

Except supervising a network at the end of a network, deeply supervised network (Lee *et al.*, 2015) has been shown effective in improving classification capability of a network, which enforces mid-level CNN features recognisable by providing another supervision. Besides, this strategy has also been demonstrated useful in semantic segmentation (Zhao *et al.*, 2017). Last, semantic encoding loss considers global contextual information and regularizes the training by recognizing the presence of categories in images (Zhang *et al.*, 2018a).

2.1.6 Datasets

There are series of datasets proposed for semantic segmentation. In the following, we focus on the discussion of datasets used in our experiments for scene parsing including indoor scenes, onboard captured street scenes and more flexible and general scenes.

NYUDv2. This is a dataset for indoor scene understanding (Nathan Silberman and Fergus, 2012), which is collected with Kinect RGB-Depth sensors. This dataset contains widely existing indoor environments, including office room, kitchen, bedroom, restroom etc, and also rich categories of structures and objects such as floor, ceiling, wall, table, chair, sofa, mirror and so on. As a result, for semantic segmentation tasks, there are 3 popular annotation branches, i.e. 4-, 13- and 40-class

(a) NYUDv2

(b) SUN3D

Figure 2.3: Examples of indoor scenes in this thesis. In both datasets, images are captured across various scenes with different objects.

tasks as first proposed in Nathan Silberman and Fergus (2012), Couprie *et al.* (2013) and Gupta *et al.* (2013), respectively.

Statistically, NYUDv2 provides 795 training images and 654 testing images with ground truth annotations. Besides, all the annotated images are extracted from a video sequence, whose lengths are varying from dozens to hundreds frames. As a result, it also provides us many unlabeled but highly relevant images to annotated

images, enabling the possibility of semi-supervised learning. In Chapter 3, we apply NYUDv2 dataset to conduct the experiments with additional temporal unlabeled images to improve semantic segmentation.

SUN3D. This dataset captures the full 3D extent of many places (Xiao *et al.*, 2013). Similar to NYUDv2, it also provides video sequences. The images of SUN3D are captured in the similar scenes to NYUDv2, but different data distribution of various styles for moving camera and environments. For semantic segmentation, it provides a 33-class task with 65 images, which is a small-scale dataset. As a result, it is hard to train a successful segmentation model with this dataset, but can be used to test the performance of a pretrained segmentation model learned from other dataset. It is able to indicate the generalization capability of the segmentation model, as discussed in Chapter 3.

Cityscapes. Cityscapes is a high quality dataset for onboard street scene understanding (Cordts *et al.*, 2016) at high resolution of 1024×2048. The dataset is captured in 50 European cities under different weather conditions and capturing equipments.

It provides a two-stage label space, forming 8 categories and 19 classes. Normally, the 19-class definition is used for most methods, therefore, we follow other works to report the performance on 19-class task. Totally, Cityscapes provides 2975 training images, 500 validation images with ground truths, and 1525 testing images without ground truths. Last, it also provides coarse annotations, which only label the majority parts of regions and objects. Coarse annotations are able to be used as pretraining of a network before applying 2975 fine annotations. We evaluate our street scene semantic segmentation model on Cityscapes in Chapter 4 and 6.

BDD100K. Berkeley Deep Driving 100K (BDD100K) is a large and diverse dataset, which provides several tasks for the study of autonomous driving (Yu *et al.*, 2018). For semantic segmentation, it provides 19 classes, which is compatible with Cityscapes. The resolution of images in this dataset is 720×1280, and those images are captured from different vehicles in various environments like city or landscape. Besides, as a major difference to Cityscapes, this dataset provides more examples at nighttime driving scenarios. Finally it contains 7000, 2000 and 1000 images in training, validation and testing set, respectively. We study membership inference attacks on black-box semantic segmetation models in Chapter 7 with BDD100K dataset.

CamVid. CamVid is a dataset for street scene semantic segmentation (Brostow *et al.*, 2008) focusing on the driving scenarios in the downtown and nearby regions. Different to previous datasets, it is a small-scale dataset with 367, 100, 233 images in training, validation and testing set respectively. The images are at resolution of 480×640, and dense annotations with 11 classes are provided. We evaluate our street scene semantic segmentation model on CamVid in Chapter 4.

Figure 2.4: Examples of street scenes in this thesis. From top row to bottom, images from Cityscapes, BDD100K, CamVid and Mapillary Vistas, are shown respectively.

Mapillary Vistas. Mapillary Vistas is a large-scale dataset collected from industry community (Neuhold *et al.*, 2017). The images have varying resolutions whose heights are from hundreds to thousands, providing 65 predefined labels. The label space provides a fine-grained semantic segmentation problem, containing categories on very complicated driving scenarios, such as snow, water and sand. It contains images driving at different places, such as city, high way and landscape. This dataset contains 18000, 2000 images in training and validation set with detailed annotations.

Figure 2.5: Examples of ADE20K dataset.

In testing set, 5000 images are provided. We study membership inference attacks on black-box semantic segmetation models in Chapter 7 with Mapillary Vistas dataset.

ADE20K. ADE20K is a dataset focusing on general scene parsing with mixture of indoor scenes as well as outdoor natural scenes (Zhou *et al.*, 2017), as shown in Figure 2.5. The most popular branch of ADE20K has 150 categories. It contains indoor environments such as bedroom, meeting room, etc, including classes window, wall, bed, person. For outdoor scenes, it contains natural landscapes as well as city views, including classes bridge, mountain, grass, bus, etc. Except semantic segmentation annotations, it also provides instance labels for some object categories like person. Finally, it has 20210 training images as well as 2000 validation images. We evaluate our synthetic feature based pipeline for improving semantic segmentation on ADE20K in Chapter 6.

2.2 IMAGE GENERATION WITH NEURAL NETWORKS

In recent years, generative modeling achieves great progress and arises many interesting research topics and real-world applications, such as image translation and manipulation. Recent generative modeling aims to learn the distribution of training data with a neural network, and thus sample new examples from the learned network. Particularly, conditional generative modeling transforms a condition input

into an output with designed forms. Summarizing recent approaches of generative modeling, there are two main successful schemes, i.e., generative adversarial nets (GANs) (Goodfellow *et al.*, 2014) as discussed in Section 2.2.1 and variational auto-encoders (VAEs) (Kingma and Welling, 2013) as presented in Section 2.2.2. Besides, perceptual loss (Dosovitskiy and Brox, 2016; Johnson *et al.*, 2016) shows stable training property and capability to generate realistic images, and thus draws much attention, where it is discussed in Section 2.2.3.

2.2.1 Generative adversarial nets

Generative adversarial nets (GANs) (Goodfellow *et al.*, 2014) aim to learn a generator with coupling a discriminator in an adversarial way, which becomes the most popular method of generative modeling in recent years. The generator and the discriminator are learned together, that the generator takes a random vector as an input, and produces an output, which is feed to the discriminator. Except generated fake data, other part of input data for discriminator is real data, therefore, the discriminator perform binary classification on fake and real data. In other words, the goal of discriminator is to successfully distinguish a fake example from real ones, i.e. minimizing the classification errors. On the other hand, generator tries to emit examples, which are able to fool the discriminator, i.e. maximizing the classification errors. Finally, with the adversarial game, generator is able to generate examples of high quality and realistic rich details.

In the following, there are several improved GANs are proposed. For example, deep convolutional generative adversarial network (*DCGAN*) is an excellent instance of convolution architectures, which is more stable than vanilla version of GANs (Goodfellow *et al.*, 2014). Except architectures, there are also variations of GANs in terms of objectives. Least square GAN (*LSGAN*) (Mao *et al.*, 2017) replaces original cross entropy loss with mean square error for classification of the discriminator. Wasserstein GAN (*WGAN*) (Gulrajani *et al.*, 2017) applies a wasserstein distance as the measurement for two distributions, to improve the stability of training GANs and provide meaningful learning curves for debugging and hyperparameter searches.

Furthermore, there are additional strategies proposed to stabilize the training of GANs, particularly on high resolution images. *StackGAN* (Zhang *et al.*, 2017a) proposes to learn multiple generators stacking each other, where the first generator produces low resolution images, while other generators produce high resolution images. In the following, progressively growing GAN (*PGGAN*) (Karras *et al.*, 2017) learns a generator to synthesize images from low resolution images to high resolution by gradually adding new layers. Recently, *BigGAN* (Brock *et al.*, 2019) models large-scale dataset ImageNet with applying orthogonal regularization to the generator, and achieves current state-of-the-art performance on training generative adversarial nets.

2.2.1.1 *Conditional GANs for image generation*

On the applications of GANs, image translation becomes increasingly popular in recent years. GANs have been extended to conditional version (Mirza and Osindero, 2014), which feeds extra information like class labels into a generator, allowing to model the conditional distribution of data. For example, the first text-to-image system applies *DCGAN* architecture and adversarial training to synthesize different flower and bird images from text descriptions (Reed *et al.*, 2016). *ACGAN* adds an auxiliary classifier on the generator to recognize the generated data as the input class of generator, and is able to synthesize images on ImageNet dataset with 1000 classes (Odena *et al.*, 2017). Particularly, *pix2pix* (Isola *et al.*, 2017) proposes a general framework for image translation learned from paired images, such as generating colorful shoe images from skeleton images, or generating realistic street scenes from semantic layouts. Besides, image translation have been exploited for unpaired images, which is more flexible. *CycleGAN* (Zhu *et al.*, 2017a) adds a cycle-consistency constraint to learn two generator for translation between different domains. Recently, *BicycleGAN* (Zhu *et al.*, 2017b) models the distribution of latent representations under the *pix2pix* framework and generates multiple output images from different modalities.

2.2.1.2 *GANs for semantic segmentation*

GANs have been applied to semantic segmentation with respective to providing additional loss term (Luc *et al.*, 2016) or leveraging unlabeled training data (Souly *et al.*, 2017; Hung *et al.*, 2018). For example, Luc *et al.* first apply GANs into semantic segmentation area, which learns a discriminator taking posteriors from a segmentation model as the input, and tries to fool the discriminator with the posteriors. Therefore, the discriminator provides additional loss term for a semantic segmentation model and it is updated within the adversarial training. Besides, people have made efforts in leveraging unlabeled data for semi-supervised learning settings (Souly *et al.*, 2017; Hung *et al.*, 2018). On one hand, unlabeled data provide real distribution of natural images for adversarial training, and they may be helpful to train a GAN model. One the other hand, the discriminator is able to provide penalty gradients for those unlabeled data, thus it is possible to utilize more data to improve the performance.

2.2.1.3 *GANs for data augmentation*

There are several works utilizing generated data with GANs in computer vision tasks (Antoniou *et al.*, 2017; Frid-Adar *et al.*, 2018; Xian *et al.*, 2018; Sixt *et al.*, 2016; Peng *et al.*, 2018; Zheng *et al.*, 2017; Bowles *et al.*, 2018; Shrivastava *et al.*, 2017; Mueller *et al.*, 2018). The most related work to ours is Xian *et al.* (2018). This work proposes to generate embedding visual features from attributes with a GAN based generator for zero-shot image classification. This method trains a classifier with mixture of synthetic features for unseen classes and real features of seen

classes. As a result, significant improvement can be achieved in generalized zero shot image classification. Besides, Sixt *et al.* generate large amount of realistic labeled images by combining a 3D model (Sixt *et al.*, 2016). Peng *et al.* apply adversarial training to generate many hard occlusion and rotation patterns for augmentation in human pose estimation task (Peng *et al.*, 2018). Zheng *et al.* leverage large amount of unlabeled generated images with a smoothing regularization to improve person re-identification task (Zheng *et al.*, 2017). GANs are also applied to generate image/label pairs in semantic segmentation. Bowles *et al.* (2018) regard label image as an additional channel, and generate four-channel outputs from noises, and augment the training set in semantic segmentation. Finally, Shrivastava *et al.* (2017) and Mueller *et al.* (2018) address the problem of gaze estimation and hand pose estimation by utilizing the data from a rendering system and training a GAN to eliminate the distribution gap between synthetic data and real data.

Different to above methods, we learn a GAN based generator for synthesizing intermediate CNN features for improving semantic segmentation performance in Chapter 6, as well as protecting data privacy against membership inference attacks in Chapter 7.

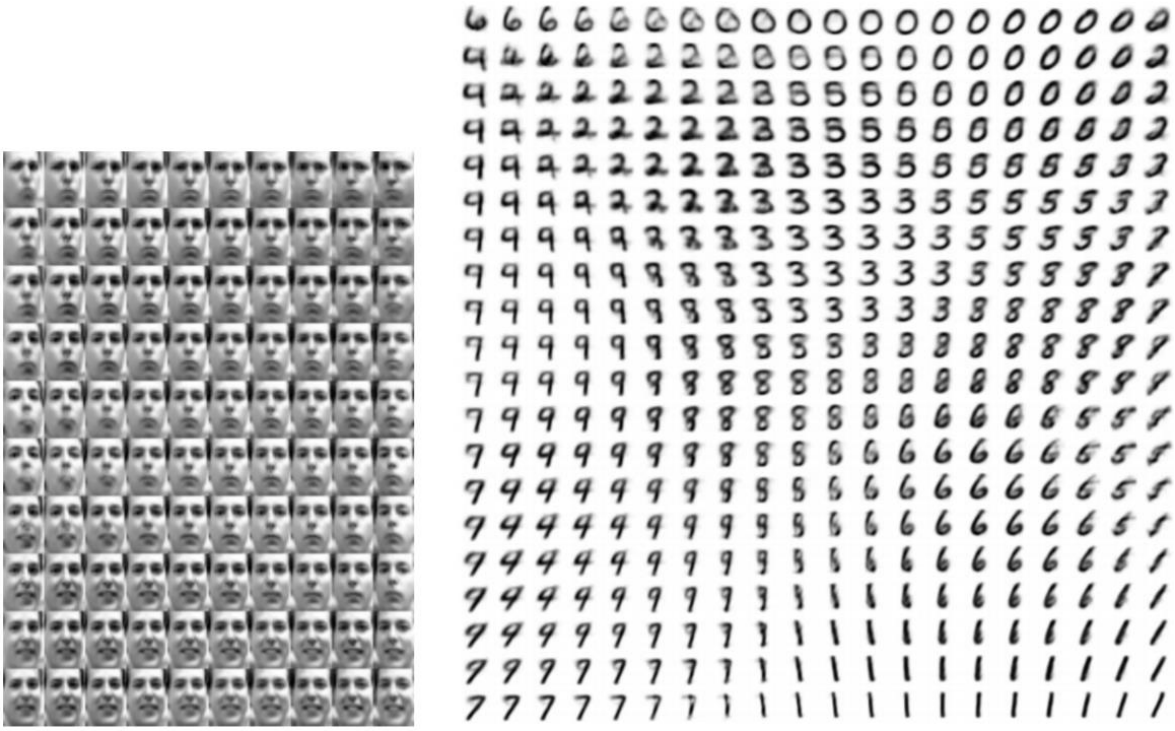


Figure 2.6: Learned manifold of synthetic examples from VAEs, as shown in (Kingma and Welling, 2013).

2.2.2 Variational auto-encoders

Variational auto-encoder (VAE) is a directed graphical model with certain types of latent variables, such as Gaussian latent variables (Kingma and Welling, 2013). It is comprised of an encoder and a decoder. Encoder aims to compress raw images into a latent vector, and decoder recover information from latent representations into final reconstruction outputs. Besides, the variables in the latent vector are obey a prior distribution, such as Gaussian distribution, which can be enforced by measuring KL divergence. Once a generator is learned, the generation process is as follows: (1) a set of latent variables is sampled from a prior distribution used in training. (2) feed the latent variables into the decoder and generate examples. It is able to map individual examples (data points) into a continuous latent space, which forms a manifold, such as the examples in Figure 2.6. It can be observed that similar examples can be drawn from close latent variables.

Besides, VAEs have been extented into conditional versions (Sohn *et al.*, 2015), which receive conditional inputs and generate compatible output examples via sampling different latent variables. Conditional VAEs have been shown to alleviate mode collapse problem, and thus produce multiple reasonable outputs. Furthermore, *BicycleGAN* (Zhu *et al.*, 2017b) combines the formulation of variational auto-encoders and generative adversarial nets into a framework with two branches. The first VAE branch tries to reconstruct input images perfectly, and the second GAN branch aims to sample reasonable images from a conditional input.

Different to VAEs, we propose an alternative pipeline which introduces latent variables via dropout into networks in Chapter 5, and our model is also able to sample multiple examples during test time.

2.2.3 Regression with perceptual loss

Except GANs or VAEs, generative modeling is also directly formulated as a regression problem with perceptual loss, which is another useful tool for generating realistic images (Johnson *et al.*, 2016; Dosovitskiy and Brox, 2016).

Unlike adversarial training, regression-based approaches are normally quite stable in training. However, they are lack of capability to model realistic details if only pixel-level loss function is applied. To overcome this, perceptual loss is utilized as the similarity measurement for regression (Johnson *et al.*, 2016). Perceptual loss is proposed to perform image style transfer (Gatys *et al.*, 2016; Johnson *et al.*, 2016), which computes the activation differences at various layers of a pretrained CNN model between a synthetic image to its corresponding ground truth. For a CNN model, its layers at different locations measure low-level visual details (i.e., edges, colors) as well as high-level semantics (i.e., parts, objects). Besides, the gradients are able to pass to the generator via back-propagation from the pretrained network. As a result, the generator is forced to produce synthetic images with similar concepts and local details to real ones. As one of successful architectures, cascaded refinement networks (CRN) focus on generating photographic street scenes with perceptual loss

and a carefully designed architecture (Chen and Koltun, 2017).

In addition, regression-based approaches are lack of sampling capability during inference time. As we all know, GANs and VAEs are able to model data distribution and thus sample multiple examples with latent variables. To generate multiple outputs, previous work (Chen and Koltun, 2017) applies multiple choice learning scheme (Guzman-Rivera *et al.*, 2012), which produces multiple hypothesis examples and only optimizes the best one by computing the distance between each hypothesis and ground truth. In addition, *pix2pixHD* (Wang *et al.*, 2018b) and *SPADE* (Park *et al.*, 2019) combine perceptual loss and generative adversarial training to synthesize high resolution street scenes with realistic details. *pix2pixHD* computes the clustering of encoded features, and produces multiple outputs from the clusters. *SPADE* introduces a spatial adaptive normalization in a generator and achieves better results. Qi *et al.* (2018) propose a semi-supervised method to produce multiple outputs from partial labeled semantic layouts by retrieving the similar structure layouts from a training dataset.

In Chapter 5, we adopt perceptual loss and generalize the deterministic network (i.e., CRN) into a stochastic version for synthesizing multiple realistic images with stronger diversity.

2.3 MEMBERSHIP INFERENCE ATTACKS AND DEFENSES

The goal of membership inference attacks is to determine if a sample of data was used in the training dataset of a machine learning model. There are various membership inference attacks on different models and data formats, including biomedical data (Backes *et al.*, 2016), locations (Pyrgelis *et al.*, 2018), purchasing records (Salem *et al.*, 2019), and images (Shokri *et al.*, 2017), etc. In the following, we first review membership inference attacks against machine learning models in Section 2.3.1. We present discussions on the relevant security issues of machine learning in Section 2.3.2. Last, we discuss defense techniques against membership inference attacks and privacy preserving machine learning in Section 2.3.3.

2.3.1 Attacks against machine learning models

It has been shown that machine learning models can be attacked to steal its membership of training data. Shokri *et al.* (2017) propose the first membership inference attack approach against machine learning tools, which apply multiple shadow models to mimic prediction behaviors of a victim model for individual class. Shadow models are trained by attackers with querying a black-box model. Examples with stronger confidences predicted by the target victim model are regarded as membership data and used to train a shadow model. Finally, a binary classifier as an attacker is trained, where its training data are the posteriors from shadow models. The binary classifier is applied to attack the victim target, by determining an example as membership or nonmembership data. Further, Salem *et al.* (2019) demonstrate

only one shadow model is enough to obtain similar attack performance compared to multiple shadow models. Besides, Model and data distribution of a shadow are able to be different to a target victim model. In this setting, membership inference attacks are also possible, which clearly indicates the risk of information leakage from machine learning models.

2.3.2 Security in machine learning

Except attacking data membership, the community also studies the security issues of inferring the functionality or architectures of a model in computer vision. Oh *et al.* (2018) propose to estimate the architecture configurations with machine learning tools from the observations of prediction patterns. Orekondy *et al.* (2019) steal the functionality of a black-box image classification model by querying random images. In other words, black-box models leak their information on the model configurations only from returned posteriors. Besides, as another family of attacks, adversary examples (Moosavi-Dezfooli *et al.*, 2016, 2017; Oh *et al.*, 2017; Fischer *et al.*, 2017; Xie *et al.*, 2017) attempt to fool a classifier after adding some noises on the images. Particularly, adversarial examples are also demonstrated in semantic segmentation problem (Fischer *et al.*, 2017; Xie *et al.*, 2017). The adversary is able to make a network lose the ability of recognizing specific categories (Fischer *et al.*, 2017), or only output a designed structural prediction.

2.3.3 Defenses and privacy preserving machine learning

Privacy-perserving machine learning aims to reduce information leakage during training with limited access to training data, which has been applied to deep learning (Abadi *et al.*, 2016; Shokri and Shmatikov, 2015). Besides, differential privacy (Dwork, 2011) provides the accuracy of statistical datasets while minimizes the privacy impact for individual example. Besides, Nasr *et al.* (2018) provide membership protection for a classifier by training a coupled attacker in an adversary manner. Zhang *et al.* (2018b) obfuscate training data before feeding them to a model, which hides the statistical properties of an original dataset by adding random noises or providing new samples. Particularly, differential privacy stochastic gradient decent (Abadi *et al.*, 2016) is an optimization method for deep neural networks that provides theoretical privacy guarantee. It adds noises on the gradients during training, which produces smoother predictions compared to standard stochastic gradient decent, and thus protects membership privacy.

Different to above membership inference attacks, we present the first attack system for black-box semantic segmentation models in Chapter 7, which reveals the information leakage happens more pronounced for semantic segmentation models. Besides, we also propose a simple defense approach with our synthetic features in Chapter 6, to mitigate information leakage effectively.

Part I

NEURAL ARCHITECTURES FOR IMPROVING SEMANTIC SEGMENTATION

Semantic segmentation has been significantly improved with end-to-end learned fully convolutional networks compared to traditional methods using hand-crafted features or modern learned CNN features from several stages. The success of modern convolutional networks comes from large-scale labeled training data as well as deep networks with numerous parameters, by fast parallel computations. To leverage large-scale datasets, designing effective models to fit those data is crucial. Therefore, in this part, we focus on designing data-driven neural modules which are compatible with existing fully convolutional architectures and learning frameworks, to improve semantic image segmentation from partially labeled videos or still images.

In Chapter 3, we present our spatio-temporal data-driven pooling (STD2P) layer, which introduces superpixels into networks and integrates temporal information from unlabeled frames of a video. We study our model on RGB-D semantic image segmentation task, where additional depth provides us superior superpixel computations as well as better recognition of each region. Our method allows for a series of training and inference settings, leading to a semi-supervised learning framework to leverage unlabeled frames, as well as enhancement during inference with temporal frames. Besides, our STD2P is extremely simply, which is a parameter-free module, and thus supports strong generalizations on other datasets. In Chapter 4, we present our learnable dilated convolutions, which is a natural extension of traditional dilated convolutions. Even though dilated convolutions have achieved great success in semantic segmentation, dilation factors are regarded as hyperparameters, which are manually set and fixed during training. In contrast, we provide a solution to learn dilation factors together with filter weights by propagating segmentation errors. It can replace the original dilated convolutions in baseline models. Through evaluations on several baseline segmentation models over two public datasets demonstrate the effectiveness of our proposed method.

3 RGB-D SEMANTIC SEGMENTATION FROM PATIALLY ANNOTATED VIDEOS

We propose a novel superpixel-based multi-view convolutional neural network for semantic image segmentation. The proposed network produces a high quality segmentation of a single image by leveraging information from additional views of the same scene. Particularly in indoor videos such as captured by robotic platforms or handheld and bodyworn RGBD cameras, nearby video frames provide diverse viewpoints and additional context of objects and scenes. To leverage such information, we first compute region correspondences by optical flow and image boundary-based superpixels. Given these region correspondences, we propose a novel spatio-temporal pooling layer to aggregate information over space and time. We evaluate our approach on the *NYU–Depth–V2* and the *SUN3D* datasets and compare it to various state-of-the-art single-view and multi-view approaches. Besides a general improvement over the state-of- the-art, we also show the benefits of making use of unlabeled frames during training for multi-view as well as single-view prediction.

3.1 INTRODUCTION

Consumer friendly and affordable combined image and depth-sensors such as *Kinect* are nowadays commercially deployed in scenarios such as gaming, personal 3D capture and robotic platforms. Interpreting this raw data in terms of a semantic segmentation is an important processing step and hence has received significant attention. The goal is typically formalized as predicting for each pixel in the image plane the corresponding semantic class.

For many of the aforementioned scenarios, an image sequence is naturally collected and provides a substantially richer source of information than a single image. Multiple images of the same scene can provide different views that change the observed context, appearance, scale and occlusion patterns. The full sequence provides a richer observation of the scene and propagating information across views has the potential to significantly improve the accuracy of semantic segmentations in more challenging views as shown in Figure 3.1.

Hence, we propose a multi-view aggregation method by a spatio-temporal data-driven pooling (STD2P) layer which is a principled approach to incorporate multiple frames into any convolutional network architecture. In contrast to previous work on superpixel-based approaches (Gadde *et al.*, 2016; Caesar *et al.*, 2016; Arnab *et al.*, 2016), we compute correspondences over time which allows for knowledgeable and consistent prediction over space and time.

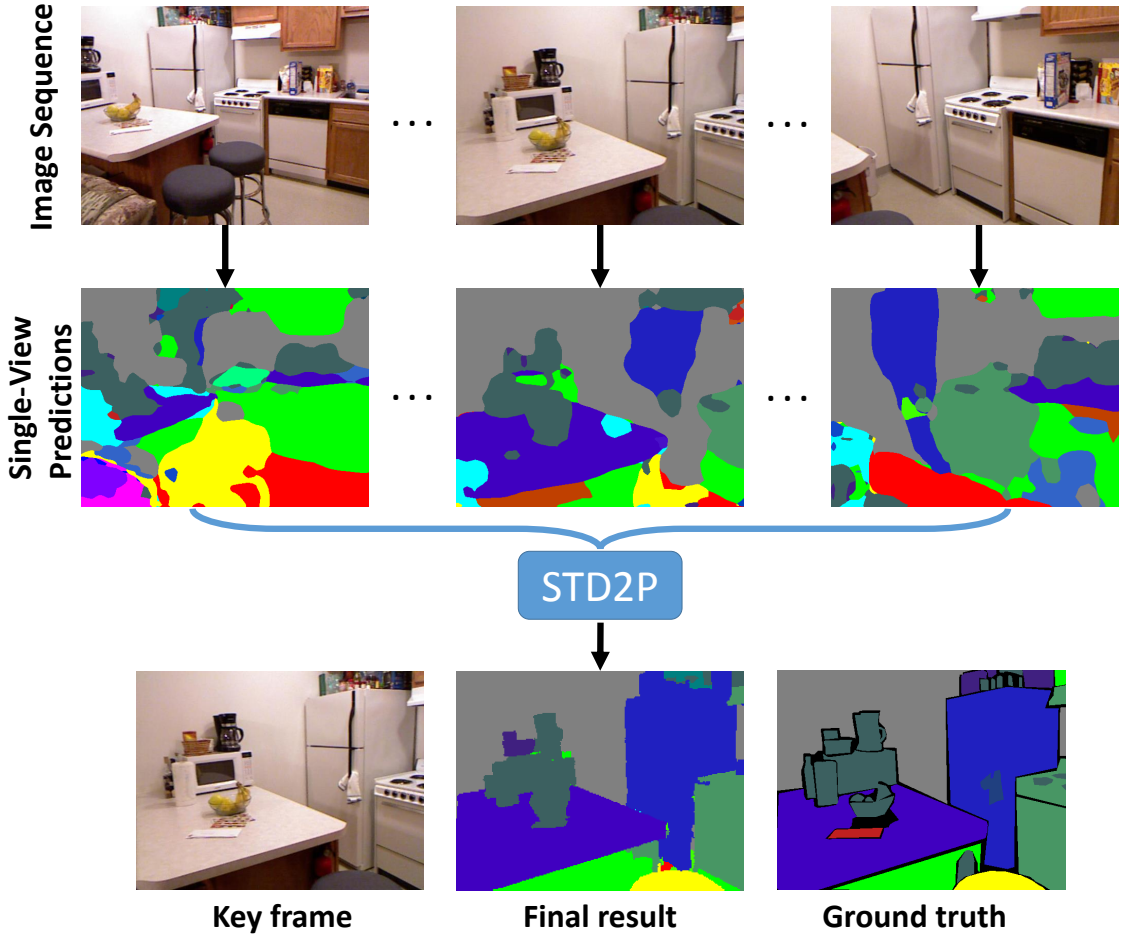


Figure 3.1: An image sequence can provide rich context and appearance, as well as unoccluded objects for visual recognition systems. Our *Spatio-Temopral Data-Driven Pooling (STD2P)* approach integrates the multi-view information to improve semantic image segmentation in challenging scenarios.

As dense annotation of full training sequences is time consuming and not available in current datasets, a key feature of our approach is training from partially annotated sequences. Notably, our model leads to improved semantic segmentations in the case of multi-view observation *as well as* single-view observation at test time. The main contributions of our paper are:

- We propose a principled way to incorporate superpixels and multi-view information into state-of-the-art convolutional networks for semantic segmentation. Our method is able to exploit a variable number of frames with partial annotation in training time.

- We show that training on sequences with partial annotation improves semantic segmentation for multi-view observation *as well as* single-view observation.

- We evaluate our method on the challenging semantic segmentation datasets *NYU–Depth–V2* and *SUN3D*. There, it outperforms several baselines as well as the state-of-the-art. In particular, we improve on difficult classes not well captured by other methods.

3.2 SPATIO-TEMPORAL DATA-DRIVEN POOLING

Our goal is a multi-view semantic segmentation scheme, that integrates seamlessly into exciting deep architectures and produces highly accurate semantic segmentation of a single view. We further aim at facilitating training from partially annotated input sequences, so that existing datasets can be used and the annotation effort stays moderate for new datasets. To this end, we draw on prior work on high quality non-semantic image segmentation and optical flow which is input to our proposed Spatio-Temporal Data-Driven Pooling (STD2P) layer.

Overview. As illustrated in Figure 3.2, our method starts from an image sequence. We are interested in providing an accurate semantic segmentation of one view in the sequence, called *target frame*, which can be located at any position in the image sequence. The two components that distinguish our approach from a standard fully convolutional architecture for semantic segmentation are, first, the computation of region correspondences and, second, the novel spatio-temporal pooling layer that is based on these correspondences.

We first compute the superpixel segmentation of each frame and establish region correspondences using optical flow. Then, the proposed data-driven pooling allows to aggregate information first within superpixels and then along their correspondences inside a CNN architecture. Thus, we achieve a tight integration of the superpixel segmentation and multi-view aggregation into a deep learning framework for semantic segmentation.

3.2.1 Region correspondences

Motivated by the recent success of superpixel based approaches in deep learning architectures (Gadde *et al.*, 2016; Caesar *et al.*, 2016; Arbeláez *et al.*, 2012; Deng *et al.*, 2015) and the reduced computational load, we decide for a region-based approach. In the following, we motivate and detail our approach on establishing robust correspondences.

Motivation. One key idea of our approach is to map information from potentially unlabeled frames to the target frame, as diverse view points can provide additional context and resolve challenges in appearance and occlusion as illustrated in Figure 3.1. Hence, we do not want to assume visibility or correspondence of objects across all frames (e.g. the nightstand in the target frame as shown in Figure 3.2). Therefore, video supervoxel methods such as Grundmann *et al.* (2010) that force interframe

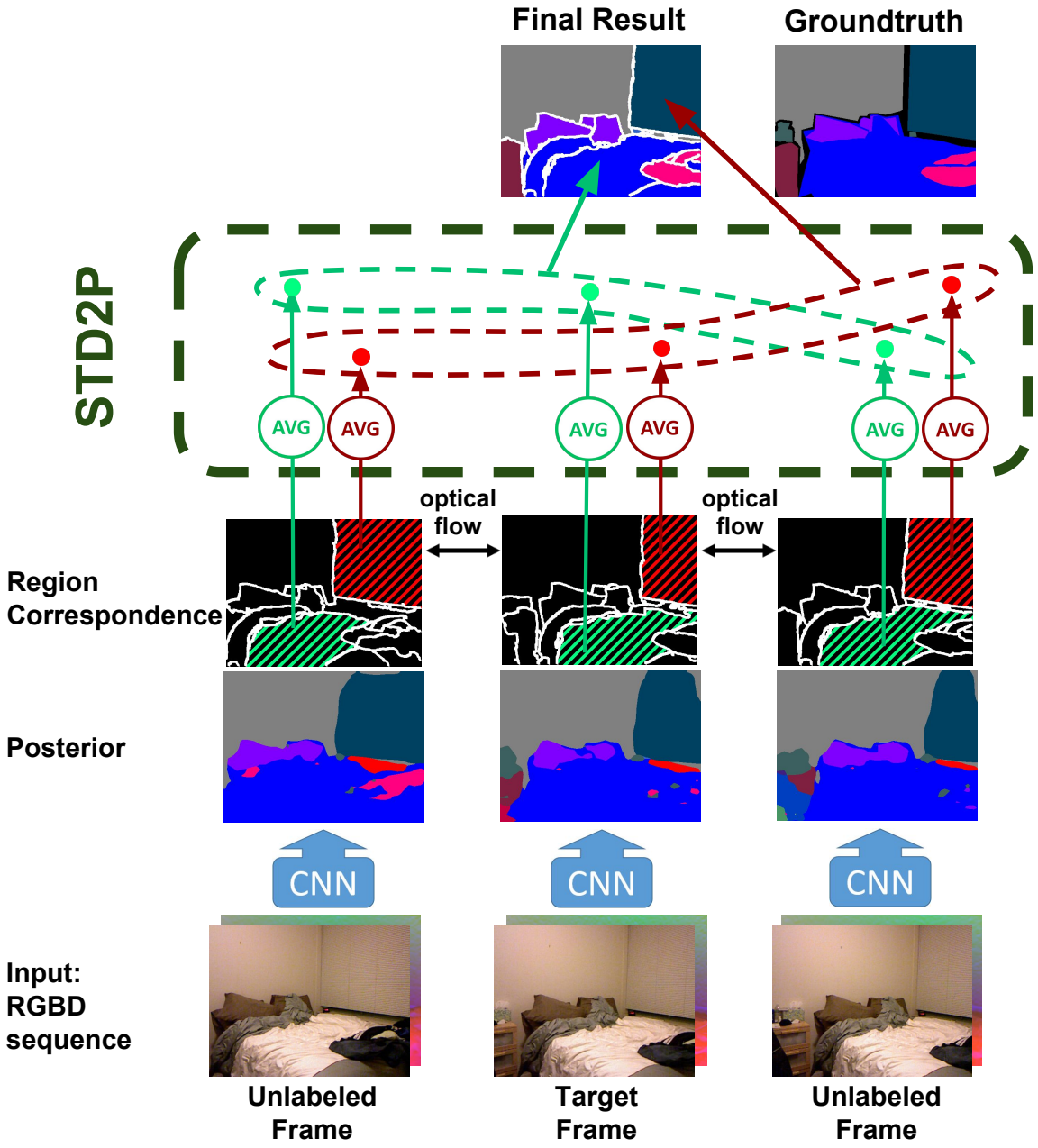


Figure 3.2: Pipeline of the proposed method. Our multi-view semantic segmentation network is built on top of a CNN. It takes a RGBD sequence as input and computes the semantic segmentation of a target frame with the help of unlabeled frames. We use superpixels and optical flow to establish region correspondences, and fuse the posterior from multiple views with the proposed Spatio-Temporal Data-Driven Pooling (STD2P).

correspondences and do not offer any confidence measure are not suitable. Instead, we establish the required correspondences on a frame-wise region level.

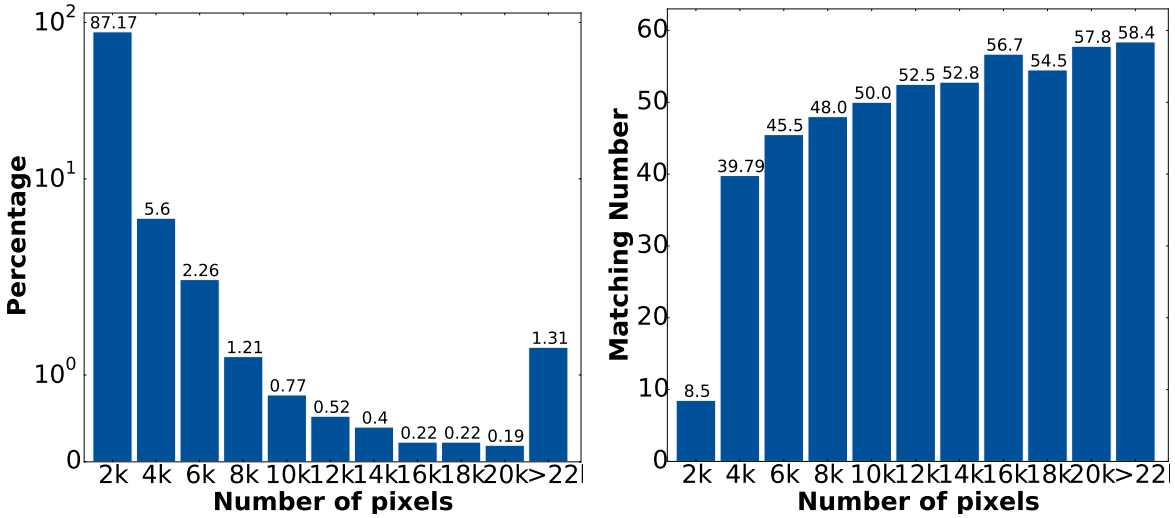


Figure 3.3: Statistics of region correspondences on the NYUDv2 dataset. (left) Distribution of region sizes; (right) Histogram of the average number of matches over region sizes.

Superpixels & optical flow. We compute RGBD superpixels (Gupta *et al.*, 2014) in each frame to partition a RGBD image into regions, and apply Epic-flow (Revaud *et al.*, 2015) between each pair of consecutive frames to link these regions. To take advantage of the depth information, we utilize the RGBD version of the structured edge detection (Dollár and Zitnick, 2013) to generate boundary estimates. Then, Epic-flow is computed in forward and backward directions.

Robust spatio-temporal matching. Given the precomputed regions in the target frame and all unlabeled frames as well as the optical flow between those frames, our goal is to find highly reliable region correspondences. For any two regions R_t in the target frame f_t and R_u in an unlabeled frame f_u, we compute their matching score from their intersection over union (IoU). Let us assume w.l.o.g. that $u < t$. Then, we warp R_u from f_u to $R_u^{'}$ in f_t using forward optical flow. The IoU between R_t and $R_u^{'}$ is denoted by $\overrightarrow{IoU}_{tu}$. Similarly, we compute $\overleftarrow{IoU}_{tu}$ with backward optical flow. We regard R_t and R_u as a successful match if their matching score meets $\min(\overleftarrow{IoU}_{tu}, \overrightarrow{IoU}_{tu}) > \tau$. We keep the one with the highest matching score if R_t has several successful matches. We show the statistics of region correspondences on the NYUDv2 dataset in Figure 3.3.

It shows that 87.17% of the regions are relatively small (less than 2000 pixels) The plot on the right shows that those small regions generally only find less than 10 matches in a whole video. Contrariwise, even slightly bigger regions can be matched more easily and they cover large portions of images. They usually have more than 40 matches in a whole video, and thus provide adequate information for our multi-view network.

3.2.2 Spatio-Temporal Data-Driven Pooling (STD2P)

Here, we describe our Spatio-Temporal Data-Driven Pooling (STD2P) model that uses the spatio-temporal structure of the computed region correspondences to aggregate information across views as illustrated in Figure 3.2. While the proposed method is highly compatible with recent CNN and FCN models, we build on a per frame model using (Long *et al.*, 2015). In more detail, we refine the output of the deconvolution layer with superpixels and aggregate the information from multiple views by three layers: a spatial pooling layer, a temporal pooling layer and a region-to-pixel layer.

Spatial pooling layer. The input to the spatial pooling layer is a feature map $I_s \in R^{N\times C\times H\times W}$ for N frames, C channels with size $H \times W$ and a superpixel map $S \in R^{N\times H\times W}$ encoded with the region index. It generates the output $O_s \in R^{N\times C\times P}$, where P is the maximum number of superpixels. The superpixel map S guides the forward and backward propagation of the layer. Here, $\Omega_{ij} = \{(x,y)|S(i,x,y) = j\}$ denotes a superpixel in the i-th frame with region index j. Then, the forward propagation of spatial average pooling can be formulated as

$$O_s(i,c,j) = \frac{1}{|\Omega_{ij}|} \sum_{(x,y)\in\Omega_{ij}} I_s(i,c,x,y) \tag{3.1}$$

for each channel index c of the i-th frame and region index j. We train our model using stochastic gradient descent. The gradient of the input $I_s(i,c,x,y)$, where $(x,y) \in \Omega_{ij}$, in our spatial pooling is calculated by back propagation (Rumelhart *et al.*, 1988),

$$\begin{aligned}\frac{\partial L}{\partial I_s(i,c,x,y)} &= \frac{\partial L}{\partial O_s(i,c,j)}\frac{\partial O_s(i,c,j)}{\partial I_s(i,c,x,y)} \\ &= \frac{1}{|\Omega_{ij}|}\frac{\partial L}{\partial O_s(i,c,j)}.\end{aligned} \tag{3.2}$$

Temporal pooling layer. Similarly, we formulate our temporal pooling which fuses the information from N frames $I_t \in R^{N\times C\times P}$, which is the output of spatial pooling layer, to one frame $O_t \in R^{C\times P}$. This layer also needs superpixel information Ω_{ij}, which is the superpixel with index j of the i-th input frame. If $\Omega_{ij} \neq \varnothing$, there exists correspondence. The forward propagation can be expressed as

$$O_t(c,j) = \frac{1}{K} \sum_{\Omega_{ij}\neq\varnothing} I_t(i,c,j) \tag{3.3}$$

for channel index c and region index j, where $K = |\{i|\Omega_{ij} \neq \varnothing, 1 \leq i \leq N\}|$, which is the number of matched frames for j-th region. The gradient is calculated by

$$\begin{aligned}\frac{\partial L}{\partial I_t(i,c,j)} &= \frac{\partial L}{\partial O_t(c,j)}\frac{\partial O_t(c,j)}{\partial I_t(i,c,j)} \\ &= \frac{1}{K}\frac{\partial L}{\partial O_t(c,j)}.\end{aligned} \tag{3.4}$$

Region-to-pixel layer. To directly optimize a semantic segmentation model with dense annotations, we map the region based feature map $I_r \in R^{C\times P}$ to a dense pixel-level prediction $O_r \in R^{C\times H\times W}$. This layer needs a superpixel map on the target frame $S_{\text{target}} \in R^{H\times W}$ to perform forward and backward propagation. The forward propagation is expressed as

$$O_r(c,x,y) = I_r(c,j), \quad S_{\text{target}}(x,y) = j. \tag{3.5}$$

The gradient is computed by

$$\begin{aligned}\frac{\partial L}{\partial I_r(c,j)} &= \sum_{S_{\text{target}}(x,y)=j} \frac{\partial L}{\partial O_r(c,x,y)}\frac{\partial O_r(c,x,y)}{\partial I_r(c,j)} \\ &= \sum_{S_{\text{target}}(x,y)=j} \frac{\partial L}{\partial O_r(c,x,y)}.\end{aligned} \tag{3.6}$$

Implementation details. We regard the frames with groundtruth annotations as target frames. For each target frame, we equidistantly sample up to 100 frames around it with the static interval of 3 frames. Next, we compute the superpixels (Gupta *et al.*, 2014) and Epic-flow (Revaud *et al.*, 2015) with the default settings provided in the corresponding source codes. The threshold τ for the computation of region correspondence is 0.4 (cf. section 3.2.1). Finally, for each RGBD sequence, we randomly sample 11 frames including the target frame together with their correspondence maps as the input for our network. We use RGB images and HHA representations of depth (Gupta *et al.*, 2014) and train the network by stochastic gradient descent with momentum term. Due to the memory limitation, we first run FCN and cache the output *pool4_rgb* and *pool4_hha*. Then, we finetune the layers after *pool4* with a new network which is the copy of the higher layers in FCN. We use a minibatch size of 10, momentum 0.9, weight decay 0.0005 and fixed learning rate 10^{-14}. We finetune our model by using cross entropy loss with 1000 iterations for all our models in the experiments. We implement the proposed network using the *Caffe* framework (Jia *et al.*, 2014).

3.3 EXPERIMENTS

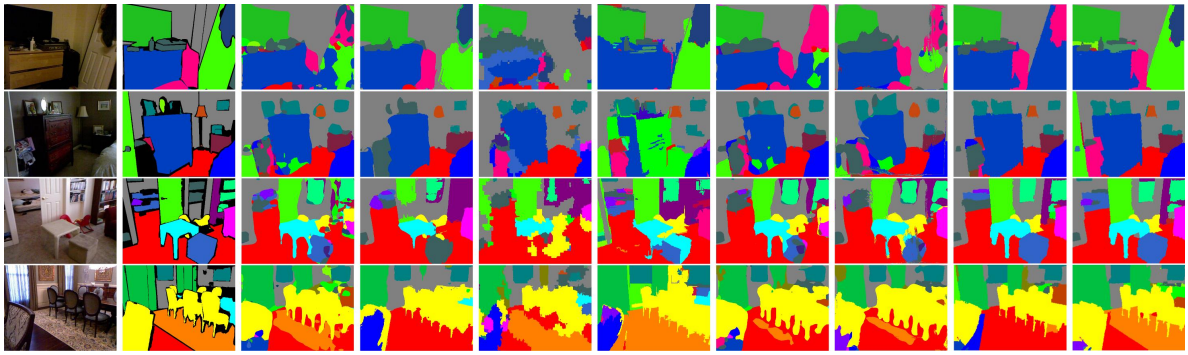
We evaluate our approach on the 4-class (Nathan Silberman and Fergus, 2012), 13-class (Couprie *et al.*, 2013), and 40-class (Gupta *et al.*, 2013) tasks of the *NYU–Depth–V2* (NYUDv2) dataset (Nathan Silberman and Fergus, 2012), and 33-class task of the SUN3D dataset (Xiao *et al.*, 2013).

Table 3.1: Configurations of competing methods

	RGB	RGBD
Single-View	Eigen and Fergus (2015) Kendall *et al.* (2015)	Caesar *et al.* (2016) Chen *et al.* (2014) Chen *et al.* (2018b) Deng *et al.* (2015) Gadde *et al.* (2016) Gupta *et al.* (2014) Long *et al.* (2015) Wang *et al.* (2014) Wang *et al.* (2016) Zheng *et al.* (2015)
Multi-View	/	Couprie *et al.* (2013) Hermans *et al.* (2014) Stückler *et al.* (2015) McCormac *et al.* (2016)

We compare our models of different settings to previous state-of-the-art multi-view methods as well as single-view methods, which are summarized in Table 3.1. We report the results on the labeled frames, using the same evaluation protocol and metrics as Long *et al.* (2015), which are also used in other Chapters of this thesis. Let n_{ij} be the number of pixels of class i predicted to belong to class j, where there are n_{cl} different classes, and let $t_i = \sum_j n_{ij}$ be the total number of pixels of class i. We define the formulation of all four metrics in the following:

- mean pixel accuracy (PixelAcc):
  $\sum_i n_{ii} / \sum_i t_i$.
- mean class accuracy (ClassAcc):
  $(1/n_{cl}) \sum_i n_{ii}/t_i$.
- mean region intersection over union (mIoU):
  $(1/n_{cl}) \sum_i n_{ii}/(t_i + \sum_j n_{ji} - n_{ii})$.



(1) (2) (3) (4) (5) (6) (7) (8) (9) (10)

Figure 3.4: Visualization examples of the semantic segmentation on NYUDv2. Column 1 shows the RGB images and column 2 shows the ground truth (black represents the unlabeled pixels). Columns 3 to 6 show the results from CRF-RNN (Zheng *et al.*, 2015), DeepLab-LFOV (Chen *et al.*, 2018b), BI(3000) (Gadde *et al.*, 2016) and E2S2 (Caesar *et al.*, 2016), respectively. Columns 7 to 9 show the results from FCN (Long *et al.*, 2015), single-view superpixel and multi-view pixel baselines. The results from our whole system are shown in column 10. Best viewed in color.

- frequency weighted region intersection over union (fwIoU):

  $(\sum_t t_k)^{-1} \sum_i t_i n_{ii} / (t_i + \sum_j n_{ji} - n_{ii})$.

3.3.1 Results on NYUDv2 40-class task

Table 3.2 evaluates performance of our method on NYUDv2 40-class task and compares to state-of-the-art methods and related approaches (Long *et al.*, 2015; Deng *et al.*, 2015; Gupta *et al.*, 2014; Kendall *et al.*, 2015; Eigen and Fergus, 2015; Zheng *et al.*, 2015; Chen *et al.*, 2014, 2018b; Gadde *et al.*, 2016; Caesar *et al.*, 2016) [1]. We include 3 versions of our approach:

Our *superpixel* model. is trained on single frames without additional unlabeled data, and tested using a single target frame. It improves the baseline FCN on all four metrics by at least 2 percentage points (*pp*), and it achieves in particular better performance than recently proposed methods based on superpixels and CNN(Gadde *et al.*, 2016; Caesar *et al.*, 2016).

[1] For Long *et al.* (2015); Deng *et al.* (2015); Gupta *et al.* (2014); Kendall *et al.* (2015); Eigen and Fergus (2015), we copy the performance from their paper. For Zheng *et al.* (2015); Chen *et al.* (2014, 2018b); Gadde *et al.* (2016); Caesar *et al.* (2016), we run the code provided by the authors with RGB+HHA images. Specifically, for Gadde *et al.* (2016), we also increase the maximum number of superpixels from 1000 to 3000. The original coarse version and the fine version are abbreviated as BI(1000) and BI(3000).

Our *superpixel+* model. leverages additional unlabeled data in the training while it only uses the target frame for test. It obtains 3.4*pp*, 2.1*pp*, 1.1*pp* improvements over the *superpixel* model on *Mean Acc.*, *Mean IoU* and *f.w. IoU*, leading to more favorable performance than many state-of-the-art methods (Deng *et al.*, 2015; Gupta *et al.*, 2014; Kendall *et al.*, 2015; Eigen and Fergus, 2015; Zheng *et al.*, 2015; Chen *et al.*, 2014; Gadde *et al.*, 2016; Caesar *et al.*, 2016). This highlights the benefits of leveraging unlabeled data.

Our *full model*. leverages additional unlabeled data both in the training and test. It achieves a consistent improvement over the *superpixel+* model and outperforms all competitors in *Mean Acc.*, *Mean IoU* and *f.w. IoU* by $0.9pp, 0.7pp, 1.0pp$ respectively. Particularly strong improvements are observed on challenging object classes such as dresser(+7.2*pp*), door(+4.8*pp*), bed(+4.7*pp*) and TV(+3.1*pp*).

Figure 3.4 demonstrates that our method is able to produce smooth predictions with accurate boundaries. We present the most related methods, which either apply CRF (Zheng *et al.*, 2015; Chen *et al.*, 2018b) or incorporate superpixels (Gadde *et al.*, 2016; Caesar *et al.*, 2016), in the columns 3 to 6 of this figure. According to the qualitative comparison to these approaches, we can see the benefit of our method. It captures small objects like chair legs, as well as large areas like floormat and door. In addition, we also present FCN and the *superpixel* model at the 7-th and 8-th column of Figure 3.4. The FCN is boosted by introducing superpixels but not as precise as our *full model* using unlabeled data.

Table 3.2: Performance of the 40-class semantic segmentation task on NYUDv2. We compare our method to various state-of-the-art methods: Long *et al.* (2015); Gupta *et al.* (2014); Kendall *et al.* (2015); Eigen and Fergus (2015) are also based on convolutional networks, Chen *et al.* (2014); Zheng *et al.* (2015); Chen *et al.* (2018b) are the models based on convolutional networks and CRF, and Gadde *et al.* (2016); Caesar *et al.* (2016); Deng *et al.* (2015) are region labeling methods, and thus related to ours. We mark the best performance in all methods with **BOLD** font, and the second best one is written with UNDERLINE.

Categories	Deng *et al.* (2015)	Gupta *et al.* (2014)	Kendall *et al.* (2015)	Eigen and Fergus (2015)	Zheng *et al.* (2015)	Chen *et al.* (2014)	Chen *et al.* (2018b)	Gadde *et al.* (2016) (1000)	Gadde *et al.* (2016) (3000)	Caesar *et al.* (2016)	Long *et al.* (2015)	Ours (*superpixel*)	Ours (*superpixel+*)	Ours (*full model*)
Wall	65.6	68.0	-	-	70.3	67.9	70.2	62.8	61.7	56.9	69.9	70.9	72.4	**72.7**
Floor	79.2	81.3	-	-	81.5	83.0	85.2	66.8	68.1	67.8	79.4	83.4	84.3	**85.7**
Cabinet	51.9	44.9	-	-	49.6	53.1	55.3	44.2	45.2	50.0	50.3	52.6	52.0	**55.4**
Bed	66.7	65.0	-	-	64.6	66.8	68.9	47.7	50.6	59.5	66.0	68.5	71.5	**73.6**
Chair	41.0	47.9	-	-	51.4	57.8	**60.5**	35.8	38.9	43.8	47.5	54.1	54.3	58.5
Sofa	55.7	47.9	-	-	50.6	57.8	59.8	35.9	40.3	44.3	53.2	56.0	58.8	**60.1**
Table	36.5	29.9	-	-	35.9	43.4	**44.5**	10.9	26.2	31.3	32.8	40.4	37.9	42.7
Door	20.3	20.3	-	-	24.6	19.4	25.4	18.3	20.9	24.6	22.1	25.5	28.2	**30.2**
Window	33.2	32.6	-	-	38.1	45.5	**47.8**	21.5	36.0	37.9	39.0	38.4	41.9	42.1
Bookshelf	32.6	18.1	-	-	36.0	41.5	**42.6**	35.9	34.4	32.7	36.1	40.9	38.5	41.9
Picture	44.6	40.3	-	-	48.8	49.3	47.9	41.5	40.8	46.1	50.5	51.5	52.3	**52.9**
Counter	53.6	51.3	-	-	52.6	58.3	57.7	30.9	31.6	45.0	54.2	54.8	58.2	**59.7**
Blinds	49.1	42.0	-	-	47.6	47.8	**52.4**	47.4	48.3	51.8	45.8	47.3	49.7	49.7
Desk	10.8	11.3	-	-	13.2	15.5	**20.7**	12.8	9.3	15.8	11.9	11.3	14.3	13.5
Shelves	9.1	3.5	-	-	7.6	7.3	9.1	8.5	7.9	9.1	8.6	7.5	8.1	**9.4**
Curtain	**47.6**	29.1	-	-	34.8	32.9	36.0	29.3	30.8	38.0	32.5	34.5	42.9	40.7
Dresser	27.6	34.8	-	-	33.2	34.3	36.9	20.3	22.9	34.8	31.0	41.6	35.9	**44.1**
Pillow	**42.5**	34.4	-	-	34.7	40.2	41.4	21.7	19.5	31.5	37.5	37.7	40.8	42.0
Mirror	30.2	16.4	-	-	20.8	23.7	32.5	13.0	13.9	31.7	22.4	20.1	27.7	**34.5**
Floormat	32.7	28.0	-	-	24.0	15.0	16.0	18.2	16.1	25.3	13.6	15.9	31.9	**35.6**
Clothes	12.6	4.7	-	-	18.7	20.2	17.8	14.1	13.7	14.2	18.3	20.1	19.3	**22.2**
Ceiling	56.7	60.5	-	-	**60.9**	55.1	58.4	44.7	42.5	39.7	59.1	56.8	55.6	55.9
Books	8.9	6.4	-	-	29.5	22.1	20.5	10.9	21.3	26.7	27.3	28.8	28.2	**29.8**
Fridge	21.6	14.5	-	-	31.2	30.6	**45.1**	21.5	16.6	27.1	27.0	23.8	38.3	41.7
TV	19.2	31.0	-	-	41.1	49.4	48.0	30.4	30.9	35.2	41.9	51.8	46.9	**52.5**
Paper	**28.0**	14.3	-	-	18.2	21.8	21.0	18.8	14.9	17.8	15.9	19.1	17.6	21.1
Towel	28.6	16.3	-	-	25.6	32.1	**41.5**	22.3	23.3	21.0	26.1	26.6	31.2	34.4
Showercur.	22.9	4.2	-	-	23.0	6.4	9.4	17.7	17.8	19.9	14.1	**29.3**	11.0	15.5
Box	1.6	2.1	-	-	7.4	5.8	**8.0**	5.5	3.3	7.4	6.5	6.8	6.5	7.8
Whiteboard	1.0	14.2	-	-	13.9	14.8	14.3	12.4	9.9	**36.9**	12.9	4.7	28.2	29.2
Person	9.6	0.2	-	-	57.9	55.3	**67.0**	45.9	44.7	35.0	57.6	66.1	66.7	**60.7**
Nightstand	30.6	27.2	-	-	31.4	37.7	41.8	15.8	15.8	17.6	30.1	37.4	34.1	**42.2**
Toilet	48.4	55.1	-	-	57.2	57.9	**69.7**	56.5	53.8	31.8	61.3	56.1	62.8	62.7
Sink	41.8	37.5	-	-	45.4	47.7	46.8	32.2	32.1	36.3	44.8	46.3	**47.8**	47.4
Lamp	28.1	34.8	-	-	36.9	40.0	**40.1**	24.7	22.8	14.8	32.1	34.5	35.1	38.6
Bathtub	27.6	38.2	-	-	39.1	44.7	**45.1**	17.1	19.0	26.0	39.2	26.7	26.4	28.5
Bag	0	0.2	-	-	4.9	6.6	2.1	0.1	0.1	**9.9**	4.8	5.8	8.8	7.3
Other struct.	9.8	7.1	-	-	14.6	18.0	**20.7**	12.2	12.3	14.5	15.2	12.7	19.3	18.8
Other furni.	7.6	6.1	-	-	9.5	12.9	12.4	6.7	5.3	9.3	7.7	12.3	10.9	**15.1**
Other props.	24.5	23.1	-	-	29.5	**33.8**	33.5	21.9	23.2	20.9	30.0	30.6	29.2	31.4
PixelAcc	63.8	60.3	68.0	65.6	66.3	68.7	**70.3**	57.7	58.9	58.1	65.4	68.5	68.4	70.1
ClassAcc	-	-	45.8	45.1	48.9	46.9	49.6	37.8	39.3	52.9	46.1	48.7	52.1	**53.8**
mIoU	31.5	28.6	32.4	34.1	35.4	36.8	39.4	27.1	27.7	31.0	34.0	36.0	38.1	**40.1**
fwIoU	48.5	47.0	-	51.4	51.0	52.5	54.7	41.9	43.0	44.2	49.5	52.9	54.0	**55.7**

Average vs. max spatio-temporal data-driven pooling. Our data-driven pooling aggregates the local information from multiple observations within a segment and across multiple views. Average pooling and max pooling are canonical choices used in many deep neural network architectures. Here we test average pooling and max pooling both in the spatial and temporal pooling layer, and show the results in Table 3.3. All the models are trained with multiple frames, and tested on multiple frames. Average pooling turns out to perform best for spatial and temporal pooling. This result confirms our design choice.

Table 3.3: Comparison of different configurations of our spatio-temporal data-driven pooling (STD2P). We employ average (AVG) and max (MAX) operations in our STD2P. For all the models, we train them with multiple frames, and test with multiple frames.

Spatial/Temporal	PixelAcc	ClassAcc	mIoU	fwIoU
AVG / AVG	**70.1**	**53.8**	**40.1**	**55.7**
AVG / MAX	69.4	51.0	38.0	54.4
MAX / AVG	66.4	45.4	33.8	49.6
MAX / MAX	64.9	44.5	32.1	47.9

Region vs. pixel correspondences. We compare our *full model*, which is built on the region correspondences, to the model with pixel correspondences. It only uses per-pixel correspondences by optical flow and applies average pooling to fuse the information from multiple views. The visualization results of this baseline are presented in column 9 of Figure 3.4. Obtaining accurate pixel correspondences is challenging because the optical flow is not perfect and the error can accumulate over time. Consequently, the model with pixel correspondences only improves slightly over the FCN baseline, as it is also reflected in the numbers in Table 3.4. Establishing region correspondences with the proposed rejection strategy described in section 3.2.1 seems indeed to be favorable over pixel correspondences. Our *full model* shows a significant improvement over the pixel-correspondence baseline and FCN in all measures.

Table 3.4: Comparison results with baselines on NYUDv2 40-class task, including our basic segmentation model FCN (Long *et al.*, 2015) and multi-view baseline with pixel correspondence. The benefits of superpixel correspondence are clearly observed.

Methods	PixelAcc	ClassAcc	mIoU	fwIoU
FCN	65.4	46.1	34.0	49.5
Pixel Correspondence	66.2	45.9	34.6	50.2
Superpixel Correspondence	**70.1**	**53.8**	**40.1**	**55.7**

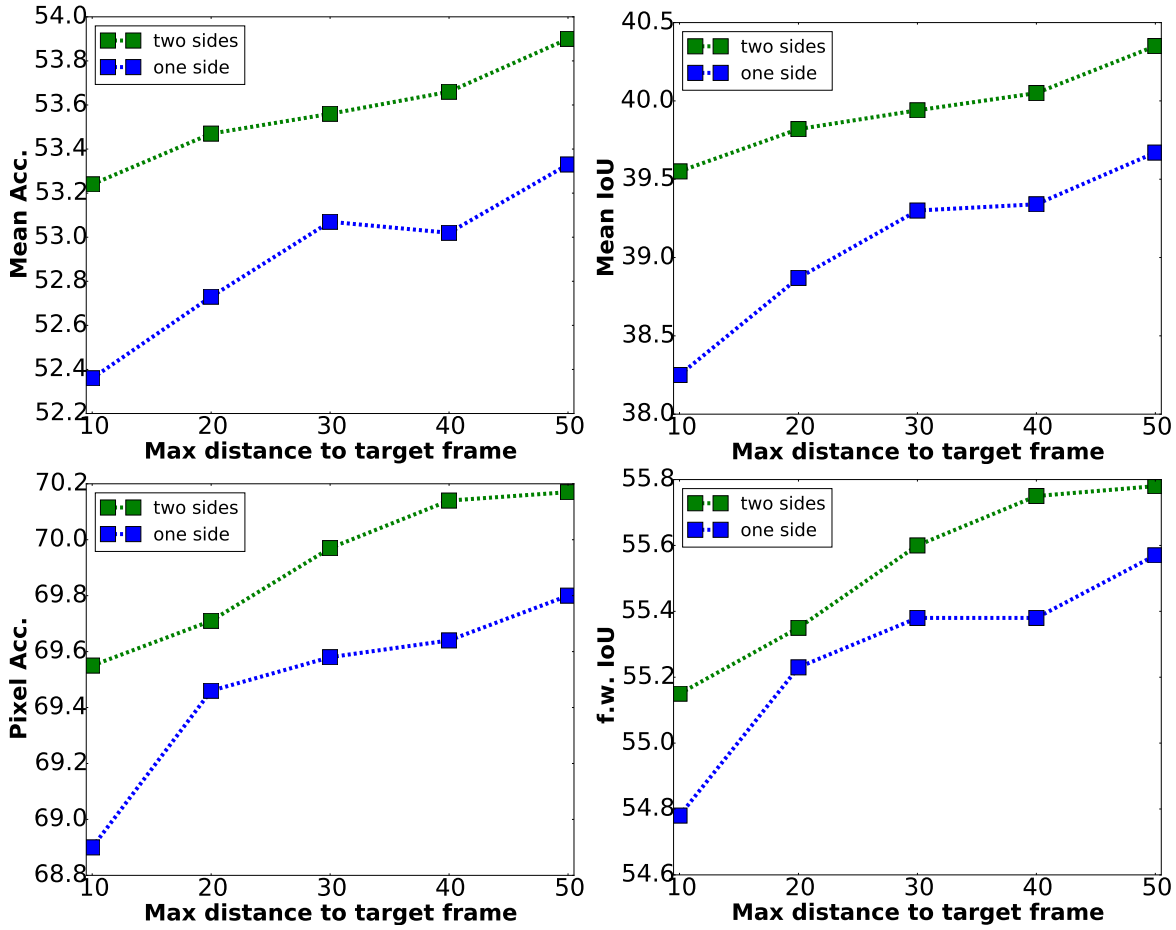


Figure 3.5: The performance of multi-view prediction with varying maximum distance. Green lines show the results of using future and past views. Blue lines show the results of only using past views.

Analysis of multi-view prediction. In our multi-view model, we subsample frames from a whole video for computational considerations. There is a trade-off between close-by and distant frames to be made. If we select frames far away from the target frames, they can provide more diverse views of an object, while matching is more challenging and potentially less accurate than for close-by frames. Hence, we analyze the influence of the distance of selected frames to target frames, and report the *Mean Acc.* and *Mean IoU* in Figure 3.5. In results, providing wider views is helpful, as the performance is improved with the increase of max distance. And selecting the data in the future, which is another way to provide wider views, also contributes to the improvements of performance.

3.3.2 Results on NYUDv2 4-class and 13-class tasks

To show the effectiveness of our multi-view semantic segmentation approach, we compare our method to previous state-of-the-art multi-view semantic segmentation methods (Couprie *et al.*, 2013; Hermans *et al.*, 2014; Stückler *et al.*, 2015; McCormac *et al.*, 2016) on the 4-class and 13-class tasks of NYUDv2 as shown in Table 3.5. Besides, we also present previous state-of-the-art single-view methods (Eigen and Fergus, 2015; Wang *et al.*, 2016, 2014). We observe that our *superpixel+* model already outperforms all the multi-view competitors, and the proposed temporal pooling scheme further boosts *Pixel Acc.* and *Mean Acc.* by more than $1pp$ and then outperforms the state-of-the-art (Eigen and Fergus, 2015). In particular, the recent proposed method by McCormac *et al.* McCormac *et al.* (2016) is also built on CNN, however, their performance on 13-class task is about $5pp$ worse than ours.

Further, We provide the qualitative results of 4-class and 13-class tasks of NYUDv2 dataset in Figure 3.6 and Figure 3.7 respectively.

Table 3.5: Performance of the 4-class (left) and 13-class (right) semantic segmentation tasks on NYUDv2 and comparison to state-of-the-art methods at the publication time. We observe that our *superpixel+* and *full model* achieve significant improvements compared to previous methods.

Methods	PixelAcc	ClassAcc	mIoU	fwIoU
Couprie *et al.* (2013)	64.5	63.5	52.4	36.2
Hermans *et al.* (2014)	69.0	68.1	54.2	48.0
Stückler *et al.* (2015)	70.6	66.8	-	-
McCormac *et al.* (2016)	-	-	69.9	63.6
Wang *et al.* (2014)	-	65.3	-	42.2
Wang *et al.* (2016)	-	74.7	-	52.7
Eigen and Fergus (2015)	<u>83.2</u>	<u>82.0</u>	<u>75.4</u>	66.9
Ours (*superpixel+*)	82.7	81.3	74.8	<u>67.0</u>
Ours (*full model*)	**83.6**	**82.5**	**75.8**	**68.4**

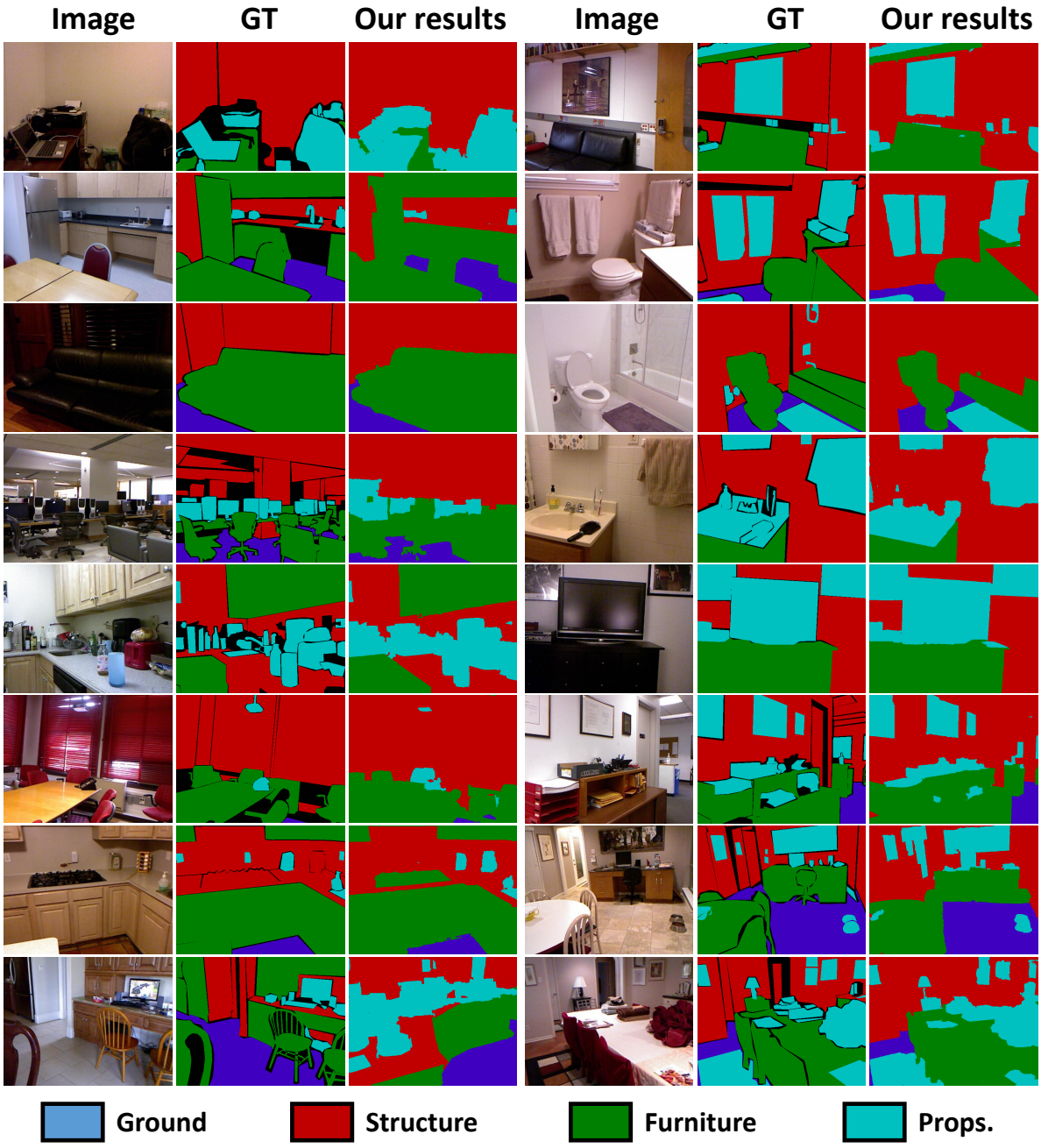


Figure 3.6: Semantic segmentation results of 4-class task on NYUDv2.

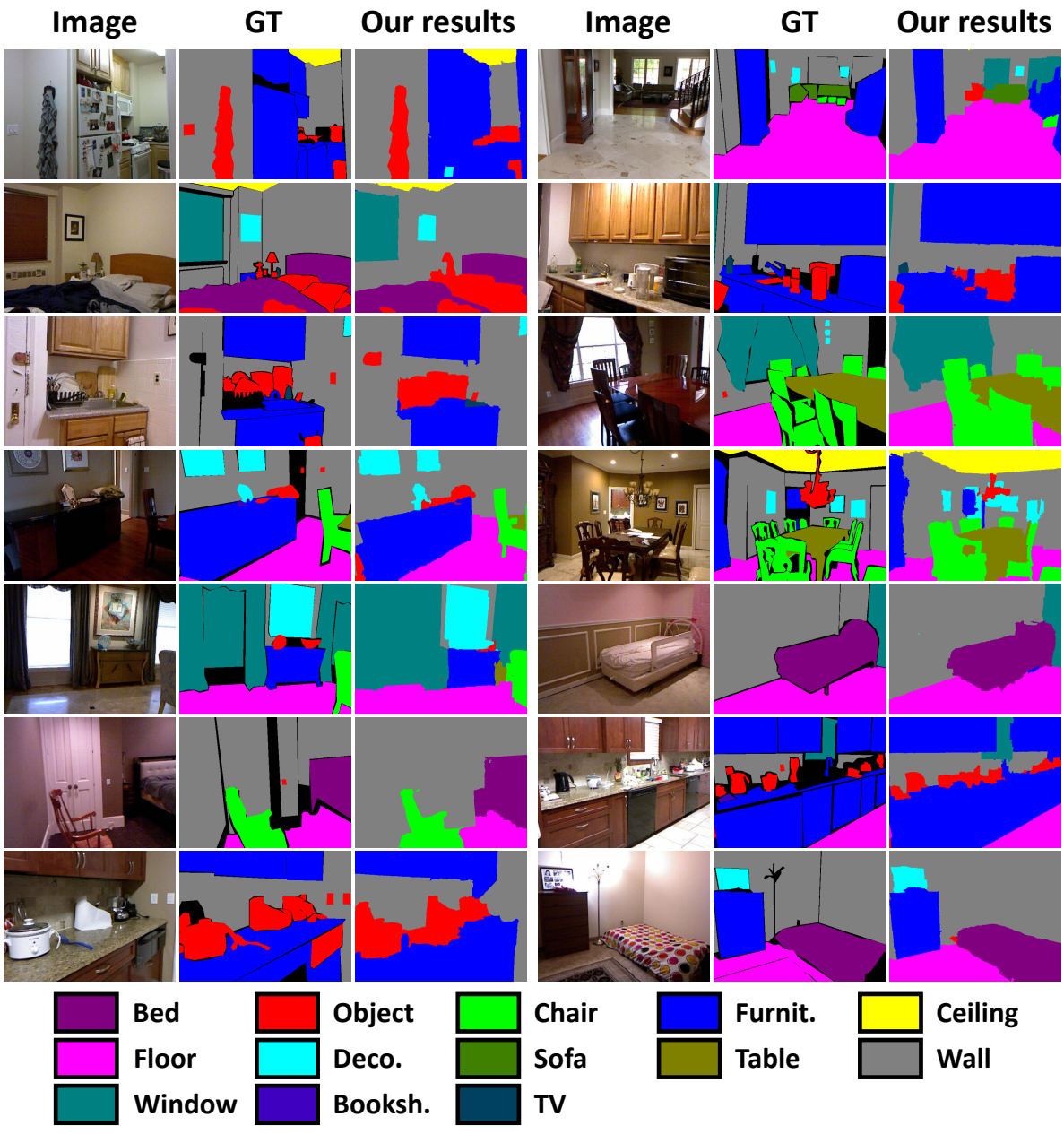


Figure 3.7: Semantic segmentation results of 13-class task on NYUDv2.

3.3.3 Results on SUN3D 33-class task

Table 3.6 shows the results of our method and baselines on the SUN3D dataset. We follow the experimental settings of Deng *et al.* (2015) to test all the methods (Deng *et al.*, 2015; Zheng *et al.*, 2015; Chen *et al.*, 2014, 2018b; Gadde *et al.*, 2016; Caesar *et al.*, 2016; Long *et al.*, 2015) on all 65 labeled frames in SUN3D, which are trained with the NYUDv2 40-class annotations. After computing the 40-class prediction, we map 7 unseen semantic classes into 33 classes. Specifically, *floormat* is merged to *floor*, *dresser* is merged to *other furni* and five other classes are merged to *other props*. Among all the methods, we achieve the best *Mean IoU* score that our *superpixel+* and *full model* are 1.2pp and 4.7pp better than Deng *et al.* (2015) and Chen *et al.* (2018b) . For *Pixel Acc.*, our method is comparable to the previous state of the art (Deng *et al.*, 2015). In addition, we observe that our *superpixel+* model boosts the baseline FCN by 3.7pp, 2.3pp, 3.3pp, 3.9pp on the four metrics, and applying multi-view information further improves 3.0pp, 0.4pp, 3.5pp, 3.7pp, respectively. Besides, we achieve much better performance than DeepLab-LFOV, which is comparable to our model on the NYUDv2 40-class task. This illustrates the generalization capability of our model, even without finetuning on the new domain or dataset. Finally, we provide the qualitative comparison between different models in Figure 3.8, which clearly demonstrate the stronger generalization capability of our parameter-free module spatial-temporal data-driven pooling.

Table 3.6: Performance of the 33-class semantic segmentation task on SUN3D. All 65 images are used as the test set.

Methods	PixelAcc	ClassAcc	mIoU	fwIoU
Deng *et al.* (2015)	**65.7**	-	28.2	51.0
Zheng *et al.* (2015)	59.8	-	25.5	43.3
Chen *et al.* (2014)	60.9	30.7	24.0	44.1
Chen *et al.* (2018b)	62.3	35.3	28.2	46.2
Gadde *et al.* (2016) (1000)	53.8	31.1	20.8	37.1
Gadde *et al.* (2016) (3000)	53.9	31.6	21.1	37.4
Caesar *et al.* (2016)	56.7	**47.7**	27.2	43.3
Long *et al.* (2015)	58.8	38.5	26.1	43.9
Ours (*superpixel+*)	62.5	40.8	29.4	47.8
Ours (*full model*)	65.5	41.2	**32.9**	**51.5**

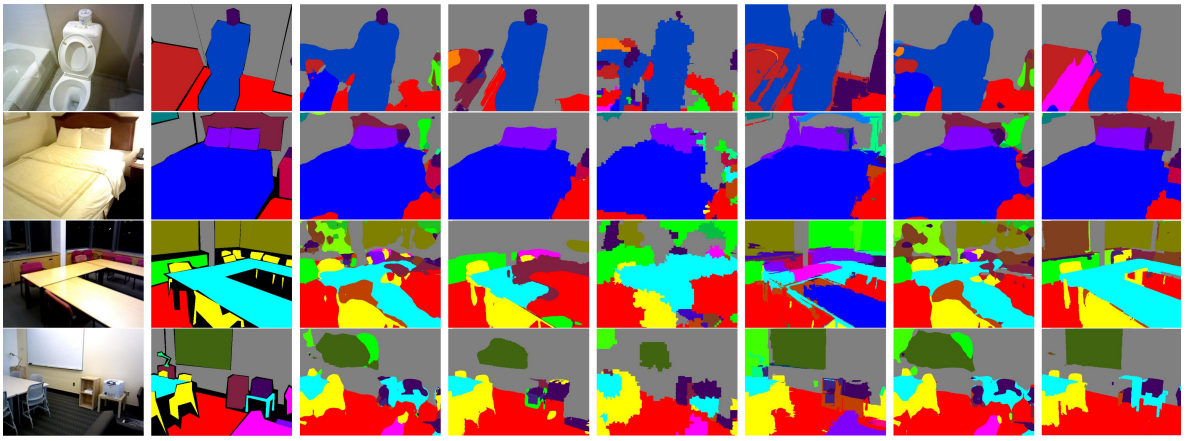


Figure 3.8: Qualititive results of the SUN3D dataset. For each example, the images are arranged from top to bottom, from left to right as color image, groundtruth, CRF-RNN (Zheng *et al.*, 2015), DeepLab-LFOV (Chen *et al.*, 2018b), BI (Gadde *et al.*, 2016), E2S2 (Caesar *et al.*, 2016), FCN (Long *et al.*, 2015), our full model.

3.3.4 Boundary accuracy for semantic segmentation

In order to quantify the improvement on semantic boundary localization based on the proposed data-driven pooling scheme, we use Boundary Precision Recall (BPR), as also used in image or video segmentation benchmark (Galasso *et al.*, 2013; Arbelaez *et al.*, 2011) for evaluation. Figure 3.9 shows the resulting semantic boundary average precision-recall curve. We conclude that our method generates more accurate boundaries than FCN, which achieve 0.477 BPR score while our method achieves 0.647. Besides, our method even improves on the superpixel (Gupta *et al.*, 2014) we build on, which means our method can successfully merge over-segmentations or non-semantic boundaries between adjacent instances of the same semantic class.

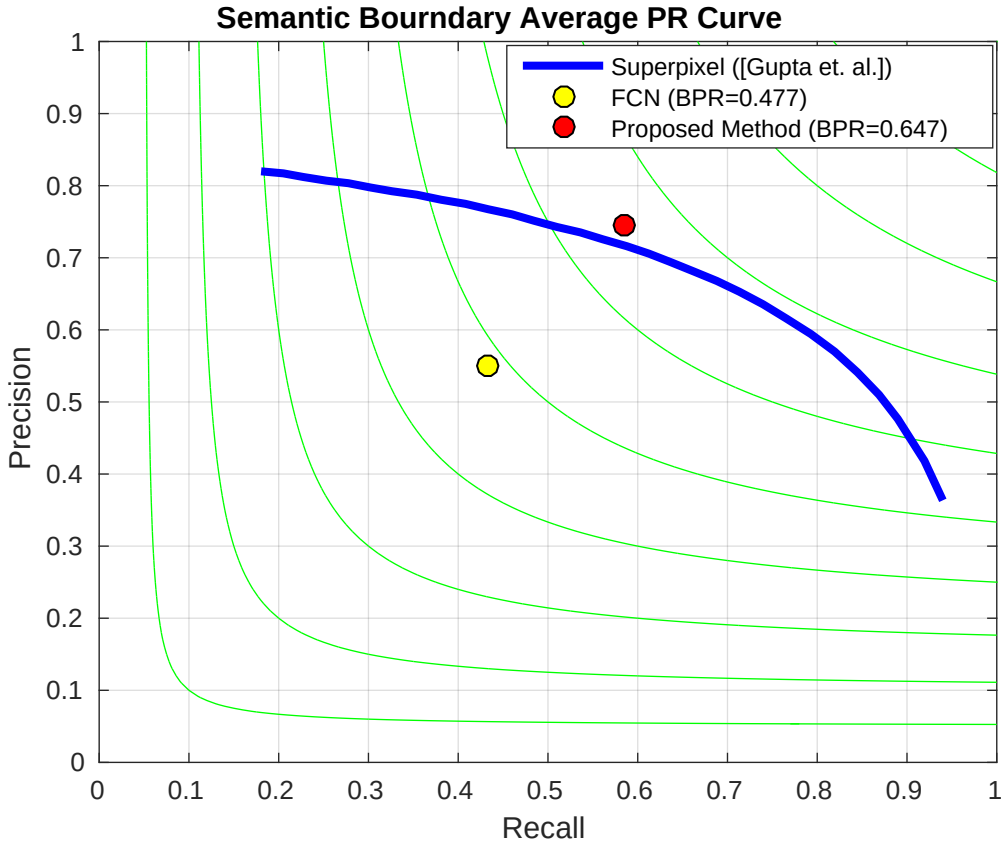


Figure 3.9: Precision-recall curve on semantic boundaries on the NYUDv2 dataset.

3.4 DISCUSSION

Our method provides a solution for improving semantic image segmentation by incorporating superpixels and leveraging temporal unlabeled frames. The experimental results show that our superpixels-based networks perform well in various settings, that achieves constant improvement by utilizing unlabeled images to boost the training of segmentation, or just leveraging multi-view information during inference.

3.4.1 Application Scenarios

The core techinique of our approach over fully convolutional networks is to introduce superpixel prior into networks. Based on superpixels, our approach is able to leverage temporal unlabeled frames for multi-view information processing for semantic segmentation. Therefore, the quality of superpixels plays an important role to obtain a significant improvement. As a result, our method allows for following application scenarios.

First, as discussed above, our approach performs particularly well when high quality superpixels are acquired. In our study, we show additional depth information benefits superpixel computation a lot, which is able to handle occlusions under low contrast background and provide more precise boundary between objects. As a result, our model with superpixels is quite suitable in indoor scene applications, such robotics and navigation systems.

Second, our approach is able to leverage large-scale unlabeled frames in partial labeled videos, to augment the training of a semantic segmentation model. As we all know, data annotations are quite expensive for semantic segmentation. While in our study, we show that it is possible to leverage unlabeled images in the past and future. Our solution provides a method to make fully use of unlabeled images, and thus achieves better segmentation performance.

Finally, our approach is able to produce continuous semantic segmentation results over a video. Although we do not train a semantic segmentation with unlabeled frames, we can still employ our spatio-temporal data-driven pooling during inference, because it is a parameter-free module. Therefore, our method is able to adapt a semantic image segmentation to sequential applications with temporal information, such as autonomous driving and robotic navigation.

Differet to previous methods incorporating superpixels into networks Caesar *et al.* (2016); Gadde *et al.* (2016), our method can leverage superpixels with correspondences, and thus integrate temporal information, which has boarder applications.

3.4.2 Technical Limitations

The key issue of our method is acquiring reliable superpixels, which is relevant to segmentation performance.

This holds true in most cases that our method is able to improve the performance, however, superpixels probably introduce new errors. On one hand, superpixels are normally computed with a threshold to determine the scale of superpixels, producing either larger superpixels, or smaller ones. In case our method introduces larger superpixels, it has the risk of merging two different objects into a superpixel, which will produce incorrect predictions. On the other hand, when our method utilizes superpixels at smaller sizes, it introduces less context information, and thus fails to improve a segmentation model a lot. Besides, it is also hard to establish region correspondences for small superpixels, because of potential errors of optical flow. Therefore, we only show clear improvements in RGB-D indoor scene scenarios, in which superpixels from additional depth information are much more successful than RGB inputs only.

Besides, even though providing additional modality, selecting a proper threshold for superpixels computation is still an issue. Generally, there are small objects as well as large regions in an image, as a result, it is hard to obtain suitable initial segments by applying only one threshold. This issue may be addressed or alleviated by introducing a superpixel hierarchy with multiple different thresholds, and then summarizing the multiscale contextual information of superpixels. For each pixel,

multiscale context information from a set of regions of various sizes are provided, which may be stronger representations for better segmentation.

Furthermore, in our current method, there is no consideration with respective to dependencies between different superpixels, even though we demonstrate improved semantic segmentation performance. Due to some rare appearances or challenging examples, it is possible to predict incorrectly, despite of providing perfect superpixels. To handle this issue, dependencies between superpixels are important information, which may refine segmentation results. Therefore, passing the information between superpixels is another possible solution to improve superpixels-based networks for segmentation, while we do not consider in this thesis.

3.5 CONCLUSION

In this chapter we studied the challenging task of semantic segmentation using partially annotated videos. We design a superpixel-based multi-view semantic segmentation network with spatio-temporal data-driven pooling which can receive multiple images and their correspondence as input. We propagate the information from multiple views to the target frame, and significantly improve the semantic segmentation performance on the target frame. Besides, our method can leverage large scale unlabeled images for training and test, and we show that using unlabeled data also benefits single image semantic segmentation.

4 LEARNING DILATION FACTORS FOR SEMANTIC SEGMENTATION

Contextual information is crucial for semantic segmentation. However, finding the optimal trade-off between keeping desired fine details and at the same time providing sufficiently large receptive fields is non trivial. This is even more so, when objects or classes present in an image significantly vary in size. Dilated convolutions have proven valuable for semantic segmentation, because they allow to increase the size of the receptive field without sacrificing image resolution. However, in current state-of-the-art methods, dilation parameters are hand-tuned and fixed. In this paper, we present an approach for learning dilation parameters adaptively per channel, consistently improving semantic segmentation results on street-scene datasets like Cityscapes and Camvid.

4.1 INTRODUCTION

Semantic segmentation is the task of predicting the semantic category for each pixel in an image, i.e. its class label from a given set of labels. It is considered a crucial step towards scene understanding and has a wide range of use-cases including autonomous driving and service robotics. The trade-off between local detail and global context is inherent in the task. The prediction of class labels requires sufficient contextual information, especially for semantic classes whose instances usually cover large portions of the image (e.g. *trucks*, *street*) or may lack local features (e.g. *sky*). At the same time, well localized detailed information is important for pixel-accurate prediction.

Recently, dilated convolutions have been proposed to improve semantic segmentation performance by providing larger receptive fields without sacrificing image resolution or adding network complexity (Yu and Koltun, 2016). While this principled idea has shown promise for recent architectures (Zhao *et al.*, 2017; Chen *et al.*, 2018b), the dilation parameters are not learned but hand-tuned and fixed. In contrast, we propose to learn the dilation parameters end-to-end thus generalizing the concept of dilated convolutions (Yu and Koltun, 2016). More specifically, we propose a fully trainable dilated convolution layer that allows to not only learn dilation parameters for each convolutional layer but for each channel individually. Thus, different features can be extracted and combined at different scales, rendering the network more flexible with respect to its receptive fields. We leverage the proposed layer to facilitate the learning of dilation parameters within three different network architectures for semantic segmentation, specifically Deeplab-LargeFOV (Chen *et al.*, 2018b), Deeplab-v2 (Chen *et al.*, 2018b) and PSPNet (Zhao *et al.*, 2017). For the task of

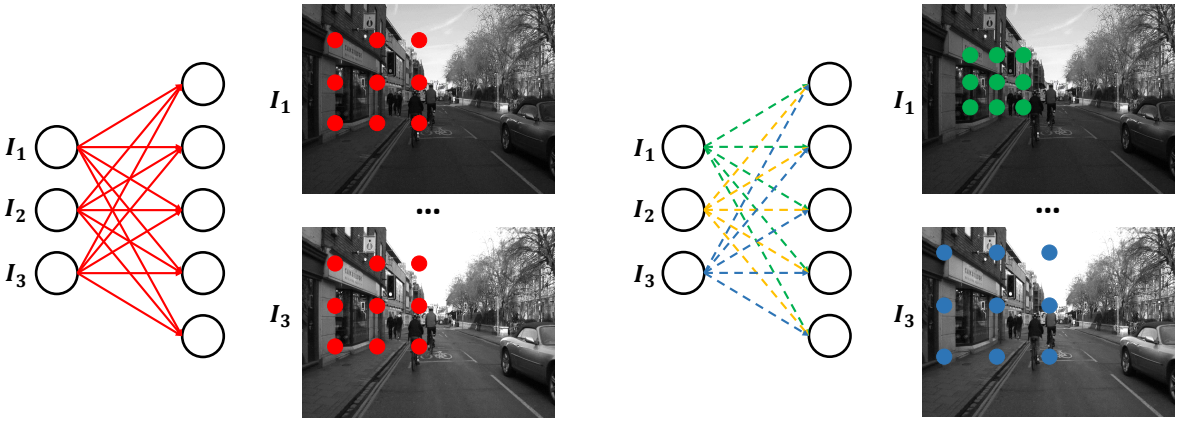


Figure 4.1: Illustration of standard dilated convolutions (left) and the proposed channel-wise learnable dilated convolutions (right). Standard dilated convolutions have a constant, manually set (solid lines) and integer valued dilation parameter for different channels. The proposed layer allows for channel-wise learning (dash lines) of dilation factors (encoded with different colors), which can take fractional values.

street scene segmentation, we show that the proposed method consistently improves results over the respective baselines.

4.2 LEARNABLE DILATED CONVOLUTIONS

In this section, we describe the concept and formulation of the channel based fractional dilated convolution layer as illustrated in Figure 4.1. Our proposed layer is a generalization of dilated convolutions as they were proposed in Yu and Koltun (2016) (Figure 4.1 (left)). Dilated convolutions provide a simple module facilitating to aggregate context information without pooling or downsampling the original image. It thus allows to preserve high spatial resolution. While previous dilated convolutions require manual tuning of an integer valued dilation parameter, the proposed method facilitates to learn dilation parameters from training data via back propagation. Thus, to allow for the definition of a gradient on the dilation parameter, these parameters can no longer be constrained to integer values but are relaxed to take a value in $\mathbb{R}^+$. To add further flexibility to the network w.r.t. the amount of context provided to each layer and each channel, we further allow for a channel-wise optimization of dilation parameters as shown in Figure 4.1 (right).

For a learned, fractional dilation factor, the output feature map of the dilated convolutions is computed using bilinear interpolation – inspired by spatial transformer networks (Jaderberg *et al.*, 2015). The proposed learnt dilated convolution layer is compatible with existing architectures, as it generalizes (and therefore can replace) convolutional and dilated convolutional layers in a given network architecture.

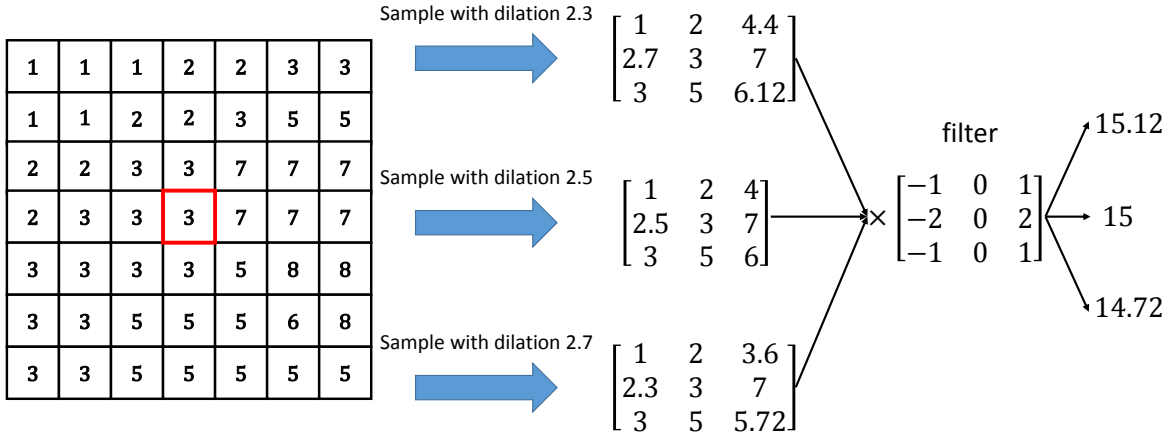


Figure 4.2: An example of the proposed dilated convolutions with a fractional dilation factor. With different dilation factors (i.e., 2.3, 2.5 and 2.7 in this figure), we obtain different input features, and then get different output activations for the red location. Assuming the current dilation factor is 2.5, we will get an output 15. With a training signal, which expects the output activation increased or decreased, we can modify the current dilation factor along the direction to 2.3 or 2.7.

4.2.1 Forward pass

We first give a brief recap on conventional dilated convolutions. With filter weights $\mathbf{W}$ and a bias term $\mathbf{b}$, the input feature $\mathbf{X}$ can be transformed to the output feature $\mathbf{Y}$ by

$$\begin{aligned} \mathbf{Y} &= \mathbf{W} * \mathbf{X} + \mathbf{b}, \qquad or \\ y_{m,n} &= \sum_c \sum_{i,j} w_{c,i,j} \cdot x_{c,m+i\cdot d,n+j\cdot d} + b, \end{aligned} \tag{4.1}$$

where d is the dilation factor, and must be an integer.

We extend the dilated convolution by introducing a dilation vector $d_c \in \mathbb{R}^{c+}$ for different channels, which can take fractional values. The forward pass of the proposed dilated convolutions can be formulated as

$$y_{m,n} = \sum_c \sum_{i,j} w_{c,i,j} \cdot x_{c,m+i\cdot d_c,n+j\cdot d_c} + b. \tag{4.2}$$

$x_{c,m+i\cdot d_c,n+j\cdot d_c}$ cannot directly be sampled from the input feature $\mathbf{X}$ for most d_c. We obtain the value of a fractional position by employing bilinear interpolation on its four neighboring integer positions as shown in Figure 4.2, specifically

$$\begin{aligned} & x_{c,m+i\cdot d_c,n+j\cdot d_c} = \\ & x_{c,\lfloor m+i\cdot d_c\rfloor,\lfloor n+j\cdot d_c\rfloor} \cdot (1-\Delta d)^2 + x_{c,\lfloor m+i\cdot d_c\rfloor,\lceil n+j\cdot d_c\rceil} \cdot (1-\Delta d)\cdot \Delta d + \\ & x_{c,\lceil m+i\cdot d_c\rceil,\lfloor n+j\cdot d_c\rfloor} \cdot \Delta d \cdot (1-\Delta d) + x_{c,\lceil m+i\cdot d_c\rceil,\lceil n+j\cdot d_c\rceil} \cdot (\Delta d)^2, \end{aligned} \tag{4.3}$$

where $\Delta d = m + i \cdot d_c - \lfloor m + i \cdot d_c \rfloor = n + j \cdot d_c - \lfloor n + j \cdot d_c \rfloor$, is the decimal part of the dilation factor d_c.

4.2.2 Backward pass

Chain rule and back propagation (Rumelhart *et al.*, 1988) are used to optimize a deep neural network model. For each layer, networks obtain the training signals (gradients) from the next connected layer, and use them to update the parameters in current layer. Then, the processed gradient is passed to the previous connected layer. Usually, those training signals are used to change the filter weights such that the output activation increases or decreases and the loss decreases. Besides changing filter weights, changing the dilation factor provides another way to optimize a convolution networks, as discussed in Figure 4.2.

The proposed dilated convolution layer based on bilinear interpolation is differentiable to dilation factors. Therefore, it allows us to train a full model end-to-end without any additional training signal for dilation factors. Because Δd in Eq. (4.3) is the decimal part of c-th channel's dilation factor d_c, the gradient for updating d_c can be formulated as

$$\begin{aligned}\frac{\partial L}{\partial d_c} &= \frac{\partial L}{\partial y_{m,n}} \cdot \frac{\partial y_{m,n}}{\partial d_c} = \frac{\partial L}{\partial y_{m,n}} \cdot \frac{\partial y_{m,n}}{\partial \Delta d} \\ &= \sum_{i,j} (x_{c,\lfloor m+i\cdot d_c \rfloor, \lfloor n+j\cdot d_c \rfloor} \cdot (2 \cdot \Delta d - 2) + x_{c,\lfloor m+i\cdot d_c \rfloor, \lceil n+j\cdot d_c \rceil} \cdot (1 - 2 \cdot \Delta d) + \\ &\qquad x_{c,\lceil m+i\cdot d_c \rceil, \lfloor n+j\cdot d_c \rfloor} \cdot (1 - 2 \cdot \Delta d) + x_{c,\lceil m+i\cdot d_c \rceil, \lceil n+j\cdot d_c \rceil} \cdot 2 \cdot \Delta d) \\ &\qquad \cdot w_{c,i,j} \cdot \frac{\partial L}{\partial y_{m,n}}.\end{aligned} \tag{4.4}$$

Besides the dilation factors, filter weights and bias term also require updating. The gradient for the filter weights $\mathbf{W}$ can be computed by

$$\frac{\partial L}{\partial w_{c,i,j}} = \frac{\partial L}{\partial y_{m,n}} \cdot \frac{\partial y_{m,n}}{\partial w_{c,i,j}} = \frac{\partial L}{\partial y_{m,n}} \cdot x_{c,m+i\cdot d_c, n+j\cdot d_c}, \tag{4.5}$$

where $x_{c,m+i\cdot d_c, n+j\cdot d_c}$ can be computed using Eq. (4.3). The gradient for the bias term $\mathbf{b}$ can be computed by

$$\frac{\partial L}{\partial b} = \frac{\partial L}{\partial y_{m,n}}. \tag{4.6}$$

To employ back propagation for optimizing all the layers, the gradient for the input feature at location (p, q) can be computed as the sum of the gradients from all the locations of output side, who sample the location (p, q) in forward pass. The gradient at the output side $\frac{\partial L}{\partial y_{m,n}}$ will affect the gradients for all sampled input

locations in bilinear interpolation. Specifically,

$$\begin{aligned}
\frac{\partial L}{\partial x_{c,\lfloor m+i\cdot d_c\rfloor,\lfloor n+j\cdot d_c\rfloor}} &= \frac{\partial L}{\partial y_{m,n}} \cdot w_{c,i,j} \cdot (1-\Delta d)^2, \\
\frac{\partial L}{\partial x_{c,\lfloor m+i\cdot d_c\rfloor,\lceil n+j\cdot d_c\rceil}} &= \frac{\partial L}{\partial y_{m,n}} \cdot w_{c,i,j} \cdot (1-\Delta d) \cdot \Delta d, \\
\frac{\partial L}{\partial x_{c,\lceil m+i\cdot d_c\rceil,\lfloor n+j\cdot d_c\rfloor}} &= \frac{\partial L}{\partial y_{m,n}} \cdot w_{c,i,j} \cdot \Delta d \cdot (1-\Delta d), \\
\frac{\partial L}{\partial x_{c,\lceil m+i\cdot d_c\rceil,\lceil n+j\cdot d_c\rceil}} &= \frac{\partial L}{\partial y_{m,n}} \cdot w_{c,i,j} \cdot (\Delta d)^2.
\end{aligned} \tag{4.7}$$

4.2.3 Network architectures

We select a set of state-of-the-art methods which use fixed and manually set dilation parameters. We employ our method and make the dilation parameters in those models learnable. Next, we describe the baseline models (Chen *et al.*, 2018b; Zhao *et al.*, 2017) in this paper and how we adopt them.

Deeplab-LargeFOV (Chen *et al.*, 2018b) is a VGG (Simonyan and Zisserman, 2014) based semantic segmentation model. It replaces conv5_1, conv5_2 and conv5_3 in original VGG network with dilation 2. And it has a convolution layer fc6 with 512 input feature channels, 1024 output channels and dilation 12. We make conv5_1, conv5_2, conv5_3 and fc6 learnable, and set the range of them as $[1,4]$, $[1,4]$, $[1,4]$ and $[4,20]$.

Deeplab-v2 (Chen *et al.*, 2018b) uses ResNet-101 (He *et al.*, 2016) to extract visual features. It modifies the original ResNet-101 with dilated convolutions. Before res5c, there are 23 layers with dilation 2, and 3 layers with dilation 4. There is an ASPP layer after res5c, which combines dilation 6, 12, 18 and 24 to recognize the class of each location. We set the range of the dilated convolution layer with dilation factor 2, 4, 6, 12, 18 and 24 to $[1,4]$, $[1,8]$, $[1,11]$, $[7,17]$, $[13,23]$ and $[19,29]$, respectively.

PSPNet (Zhao *et al.*, 2017) has a similar architecture to *Deeplab-v2*, and achieves state-of-the-art performance on various datasets. There are 23 layers with dilation 2, and 3 layers with dilation 4. Similarly, we set the range of the dilated convolution layer with dilation factor 2 and 4 to $[1,4]$ and $[1,8]$. We show that our method can boost *PSPNet*, and achieve new state-of-the-art performance on the challenging street view dataset Cityscapes (Cordts *et al.*, 2016).

4.2.4 Implementation details

We train our Deeplab-LargeFOV, Deeplab-v2 and PSPNet models with filter weights initialized from released, original models, which are trained on PASCAL VOC 2012 (Everingham *et al.*, 2010), MS-COCO (Lin *et al.*, 2014) and Cityscapes (Cordts *et al.*, 2016), respectively. We initialize the dilation factors with the manually set value of the original dilated convolutions. We apply batch SGD with momentum to optimize

Table 4.1: Hyperparameters in the experiments of this section.

Datasets	Networks	Batch size	Image patch size	Base learning rate	iterations
Cityscapes S	Deeplab-LargeFOV	10	321× 321	1×10^{-3}	20,000
	Deeplab-v2	10	321× 321	2.5×10^{-4}	20,000
	PSPNet	16	561× 561	1×10^{4}	20,000
CamVid	Deeplab-v2	10	321× 321	2.5×10^{-4}	15,000
	PSPNet	10	473× 473	1×10^{-4}	10,000

the models. The batch size is set to 10, the momentum is 0.9 and the weight decay is 0.0005. We use the "poly" learning rate policy where the current learning rate is equal to the base learning rate multiplying $(1 - \frac{iter}{iter_{max}})^{power}$, and $power = 0.9$ for all the experiments. The other hyperparameters are presented in Tab. 4.1. We use the respective training code released from the authors. We implement our method using the *Caffe* (Jia *et al.*, 2014) framework. In our experiments, our models need 3% to 8% additional time in inference. Training Deeplab-LargeFOV, Deeplab-v2 and PSPNet need only additional 10%, 15% and 15% computational time compared to the base models, respectively.

4.3 EXPERIMENTS

We evaluate the proposed method and baselines on the public benchmarks Cityscapes (Cordts *et al.*, 2016) and CamVid (Brostow *et al.*, 2008) using four evaluation metrics following previous work (Long *et al.*, 2015): pixel accuracy (*Pixel Acc.*), mean class accuracy (*Cls Acc.*), region intersection over union (*Mean IoU*), and frequency weighted intersection over union (*f.w. IoU*).

4.3.1 Cityscapes

Cityscapes (Cordts *et al.*, 2016) is a recently released street scene dataset, which is collected from diverse cities in different seasons. The image resolution in Cityscapes is 1024×2048 and the image quality is very high. It defines 19 semantic classes covering traffic, stuff and objects. There are 2975, 500 and 1525 carefully annotated images for training, validation and testing. Besides, there are also 20,000 coarsely annotated images provided for additional training data. Following previous work (Zhao *et al.*, 2017), we leverage those coarse annotations during training to obtain state-of-the-art performance.

Ablation study for Deeplab-LargeFOV. We first provide an ablation study on the Cityscapes validation set to show the effectiveness of learning dilation factors using the baseline model Deeplab-LargeFOV. We train models with different dilation configur-

ations (see Tab. 4.2), varying fixed dilations in conv5_1 to conv5_3 from 1 to 4, and learning parameters for conv5_1 to conv5_3 and fc6.

The first row in Tab. 4.2 shows the performance of the baseline method without dilated convolutions, which is only 58.91% mean IoU. Fixed integer valued dilated convolution parameters can improve the performance to up to 62.51% for a factor of 4 in conv5_1 to conv5_3.

By replacing the fixed dilation parameters with our learnable dilated convolutions using the same dilation factors as initialization, we get a further improvement to up to 63.31% mean IoU. Besides, we also use uniform distribution for the initialization of dilated convolutions, obtaining 62.92% mean IoU, which is comparable to constant value initialization, and better than the models with fixed dilation factors. We show the dilation distribution over input channels in Figure 4.3. We observe that the learned dilation distributions of using constant value and random noise, are very similar, which is clearly shown the stability of optimization. We observe that our dilation covers most values in the range of $[1, 4]$ for conv5_1 to conv5_3, which allows us capture *local* details and *wide* context at the same time. The second observation is that there are some peaks in the distribution, which make the current layer capture more local information or capture a wider context. For a specific dilated convolution layer in a network, it is very difficult to know whether the current convolution

Table 4.2: Ablation study on the Cityscapes validation set using the VGG based Deeplab-LargeFOV model. Black numbers for the convolutional layers indicate fixed dilation parameters, red numbers or ranges in our learnable dilated convolution layers indicate the initial values or distributions before training.

conv5_1	conv5_2	conv5_3	fc6	*Pixel Acc.*	*Cls Acc.*	*f.w. IoU*	*Mean IoU*	learned conv5_{1-3}	learned fc6
1	1	1	12	93.11	68.70	87.76	58.91		
2	2	2	12	93.49	71.76	88.36	61.44		
3	3	3	12	93.50	71.93	88.37	62.17		
4	4	4	12	93.49	72.35	88.38	62.51		
2.35	2.6	3.5	12	93.63	72.46	88.58	62.94		
2	2	2	12	93.50	73.25	88.41	62.31	✓	
2	2	2	12	93.65	72.64	88.61	62.86		✓
[1,4]	[1,4]	[1,4]	[4,20]	93.69	72.90	88.71	62.92	✓	✓
2	2	2	12	**93.72**	**73.38**	**88.77**	**63.31**	✓	✓

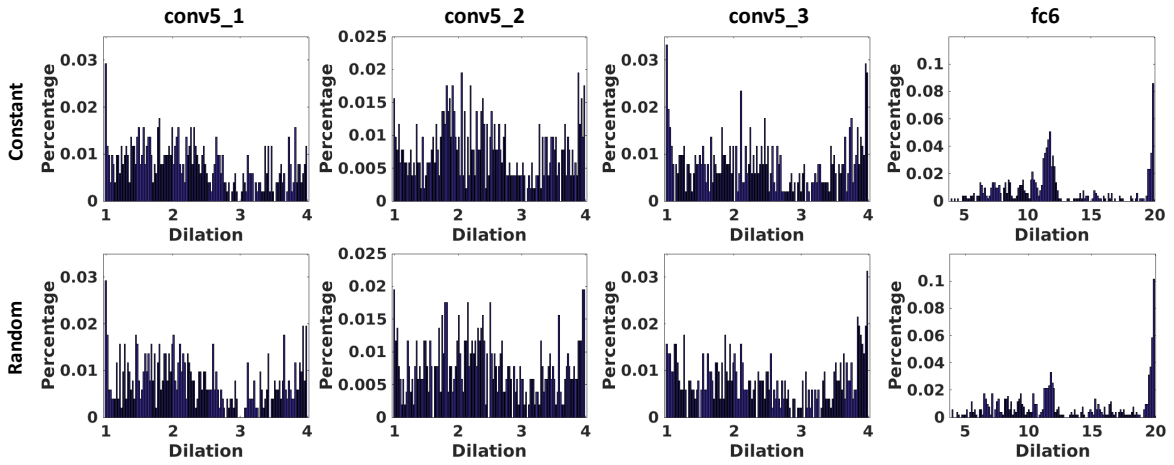


Figure 4.3: The learned dilation distribution in the Deeplab-LargeFOV model on Cityscapes dataset. The first row shows the distribution using constant value initialization, and the second row shows the distribution using random noise as presented in Tab. 4.2.

should capture local or global information, which is why the proposed learning based model achieves better performance than all fixed dilation settings. To verify this point, we train a further model with fixed dilated convolutions with factors 2.35, 2.6 and 3.5 which are the average values of the learned dilation distribution over all channels. This model achieves better results than all the other fixed settings but is worse than the channel-wise learned setting.

Comparisons on Deeplab-v2 and PSPNet. We choose Deeplab-v2 and PSPNet as our baselines, which leverage the powerful ResNet (He *et al.*, 2016) to build their models. We report the *Mean IoU* score. The comparison results on Cityscapes validation set can be found in Tab. 3. Compared our results to baselines, we observe that we got general improvement in most classes like "Fence", "Terrain", "Car" and "Bus". Particular, in some challenging and important (core role in traffic scenarios) classes like "Person" and "Rider", we got clearly improvements for Deeplab-v2 as well as PSPNet. Due to deformations and large appearance variances, "Person" and "Rider" are easily confused each other. By learning appropriate details and context, we boosted Deeplab-v2 and PSPNet to recognize "Person" (+0.6 percentage points (pp) for Deeplab-v2 and +1.3pp for PSPNet) and "Rider" (+0.8pp for Deeplab-v2 and +4.9pp for PSPNet). Figure 4.4 shows some qualitative results from baselines and our models. In the left two columns, Deeplab-v2 baseline recognizes the rider and the middle section of the train to classes "person" and "bus", which are confusing to ground truth. In the right two columns, PSPNet baseline fails to recognize the font part of the truck and the rider, while our method shows improved predictions.

Table 4.3: Comparison IoU scores on Cityscapes validation set.

Method	Road	Sidewalk	Building	Wall	Fence	Pole	Traf. Light	Traf. Sign	Vegetation	Terrain	Sky
Chen *et al.* (2018b)	97.2	78.7	90.2	49.3	48.8	52.5	**57.7**	69.7	90.8	59.4	92.7
+ Ours	97.2	**79.1**	**90.5**	**52.2**	**49.9**	**53.2**	57.4	**70.1**	**90.9**	**59.6**	**92.9**
Zhao *et al.* (2017)	98.3	86.4	**93.1**	**60.6**	65.9	**64.3**	72.0	81.1	92.6	64.7	94.9
+ Ours	98.3	86.4	93.0	59.1	**66.4**	64.0	**72.7**	**81.3**	92.6	**65.6**	94.9

Method	Person	Rider	Car	Truck	Bus	Train	Motorcycle	Bicycle	*Mean IoU*
Chen *et al.* (2018b)	76.6	53.3	92.6	**66.8**	78.1	**61.3**	**60.4**	71.9	70.9
+ Ours	**77.2**	**54.6**	**92.8**	62.8	**79.9**	58.9	58.9	**72.2**	**71.1**
Zhao *et al.* (2017)	82.5	61.5	95.3	81.4	89.7	**84.5**	62.8	78.7	79.4
+ Ours	**83.3**	**66.4**	**95.4**	**83.0**	**89.9**	80.6	**66.8**	**78.9**	**79.9**

4.3.2 CamVid

CamVid is a smaller street view dataset captured from onboard camera. We not only compare our networks to baseline methods, but also compare to previous state-of-the-art methods on Camvid. For the ease of comparison, we utilized the training and test setup from Sturgess *et al.* (2009), which has 11 semantic classes, 367

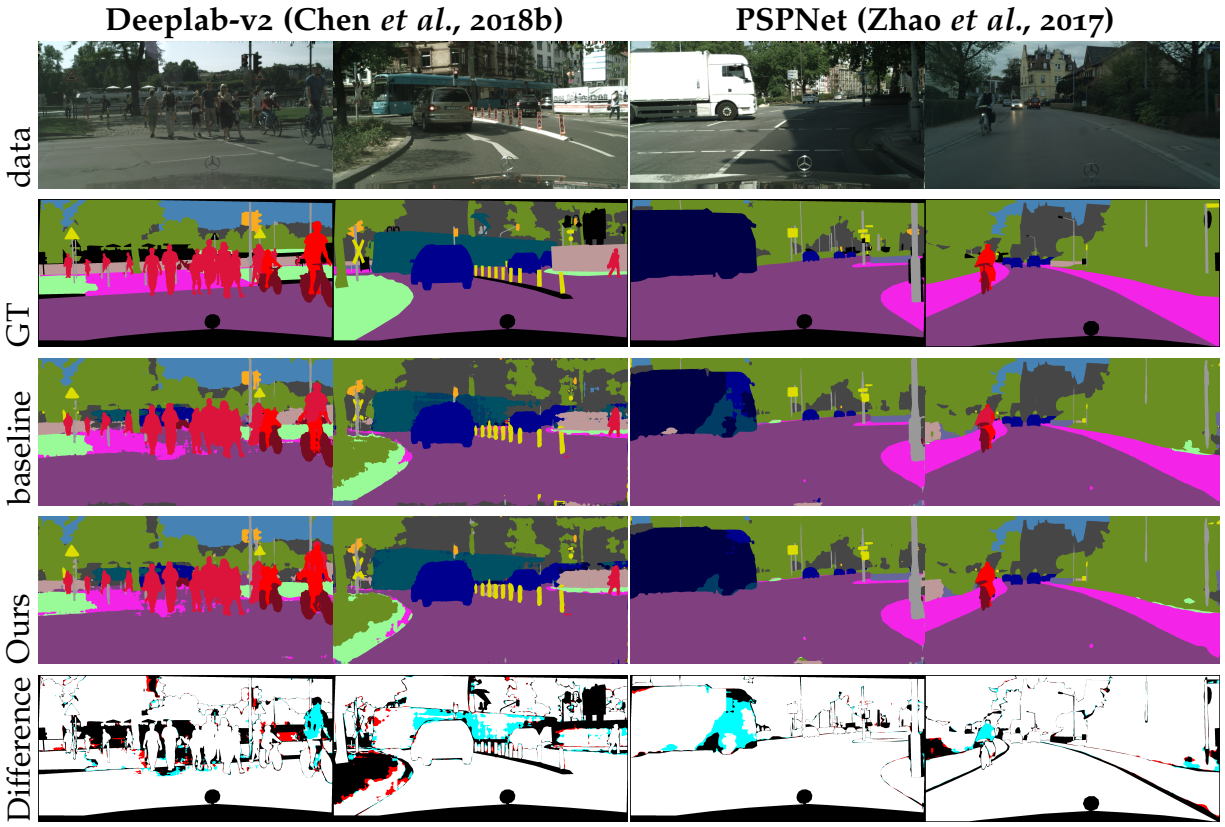


Figure 4.4: Qualitative results on the Cityscapes validation set for Deeplab-v2 (Chen *et al.*, 2018b) (left) and PSPNet (Zhao *et al.*, 2017) (right). The first four rows show the raw images, ground truth, baselines' predictions and our predictions. The last row is a visual comparison of correctly classified pixels. In white areas, both predictions are correct, in *red* areas, only the baseline prediction is correct and in *cyan* colored areas, the proposed predictions are correct, while the baseline prediction is erroneous.

training, 100 validation and 233 test images. The image resolution in our experiments is 640×480. The quantitative results are summarized in Tab. 4. We improve over Deeplab-v2 and PSPNet for 0.9 *pp* and 0.5 *pp*, respectively. For most classes, we obtain comparable performance. Particularly, in the classes of "Sign", "Fence" and "Bicyclist", our method achieves clear improvements over Deeplab-v2 as well as PSPNet. This shows the benefit of our learned dilation: State-of-the art methods can be improved to recognize a range of classes better.

Table 4.4: Quantitative results on CamVid dataset. With the proposed dilated convolutions, our method achieves better performance than two baselines, and we present new state-of-the-art performance on the CamVid dataset.

Method	Building	Tree	Sky	Car	Sign	Road	Pedestrian	Fence	Pole	Sidewalk	Bicyclist	*mean IoU*
Yu and Koltun (2016)	82.6	76.2	89.9	84.0	46.9	92.2	56.3	35.8	23.4	75.3	55.5	65.3
Kundu *et al.* (2016)	84	77.2	91.3	85.6	49.9	92.5	59.1	37.6	16.9	76.0	57.2	66.1
Richter *et al.* (2016)	84.4	77.5	91.1	84.9	51.3	94.5	59	44.9	29.5	82	58.4	68.9
Chen *et al.* (2018b)	83.8	**76.5**	90.9	**89.1**	46.0	**94.6**	57.0	28.4	19.6	**81.4**	50.1	65.2
+ Ours	**84.1**	76.3	90.9	88.8	**47.5**	94.4	**58.1**	**32.4**	**20.0**	80.8	**54.0**	**66.1**
Zhao *et al.* (2017)	87.8	**79.5**	**91.4**	91.4	57.7	96.5	66.7	58.6	**23.5**	87.8	66.9	73.4
+ Ours	**88.0**	79.3	91.3	**91.7**	**58.8**	96.5	**66.9**	**61.9**	23.0	87.8	**69.4**	**73.9**

Table 4.5: Quantitative results on CamVid dataset. With the proposed dilated convolutions, our method achieves better performance than two baselines, and we present new state-of-the-art performance on the CamVid dataset.

Dataset	Model	Baseline	Ours
Camvid	Deeplab-v2	65.2	66.1
	PSPNet	73.4	73.9
Cityscapes	Deeplab-v2	70.9	71.1
	PSPNet	79.4	79.9

4.4 DISCUSSION

Our approach provides a solution for learning dilation factors in convolutions, which are the key operations for CNNs. Instead of setting a dilation factor manually, we learn this parameter jointly with filter weights by propagating errors. The experimental results on a series of baseline models and datasets demonstrate that our approach learns different dilation factors for the convolutions at different layers of a network on various datasets. As a result, our models obtain consistent improvement over their counterparts with standard convolutions.

4.4.1 Application Scenarios

As an extension of the basic convolution operation in CNNs, our method is an alternative choice in building CNNs. Particularly, our approach is flexible to use and worth to apply in the architecture with dilated convolutions (Yu and Koltun, 2016), such as the applications of semantic segmentation, such as street scene understanding, indoor scene understanding, and medical diagnosis systems, etc.

Second, our learnable dilated convolutions have channel-wise dilation factors, which are able to capture local details as well as wider context of scenes. Therefore, our convolution operation with channel-wise dilation factors is a kind of multiscale module, extracting different levels of features. Consequently, in some other applications, such as depth prediction, it is also possible to leverage our dilated convolution.

Our approach also provide an optimization framework for updating the dilation factors. Except directly applying our approach, i.e. learning a dilation factor for each channel, it is possible to learn context adaptive dilation factors for different input images. In this case, dilation factors may be generated by another network, or integrate prior knowledge in computing reliable dilations.

4.4.2 Technical Limitations

As we can observe, the improvement of our models over baselines are not significant. However, when applying suitable fixed dilation factors via AutoML (Chen *et al.*, 2018a; Liu *et al.*, 2019), more improvements can be obtained. One of possible reason for limited improvements is about optimization. From figure **??**, we show loss function can be decreased by updating either dilation factors or filter weights. However, there is no guarantee on the decrease of loss values for the joint optimization, which may limit the optimization for a better local minimum solution.

Similarly, the idea of learning shape and scale for convolution kernels, have also been studied by other works (Jeon and Kim, 2017; Zhang *et al.*, 2017b). Such approaches and ours have similar optimization strategy with bilinear interpolation. However, those modules have not achieved great impact like dilated convolution for semantic segmentation, or inception modules and residual blocks for general visual recognition. The main reason is improvements of those modules are not significant, despite of extra computation and memory cost. Consequently, better optimization is necessary to learn more proper dilation factors.

4.5 CONCLUSION

In this chapter, we have presented learnable dilated convolutions, which is fully compatible with existing architectures and adds only little overhead. We have applied our novel convolutional layer to learn channel-based dilation factors in the semantic segmentation scenario. Thus, we were able to improve the performance of Deeplab-LargeFOV, Deeplab-v2 and PSPNet for the semantic segmentation of street

scenes consistently across two datasets. We showed that our method is able to obtain visually more convincing results, and improved quantitative performance. Besides, a series of ablation studies shows that learning the dilation parameter is helpful to design better semantic segmentation models in practice.

Part II

ON THE TRAINING DATA IN SEMANTIC SEGMENTATION AND BEYOND

Except network architectures as discussed in Part I, data are also crucial to train deep neural networks. Since dense annotations for semantic segmentation are quite expensive, synthetic data and generative modeling have drawn much attention in recent years, which aims to exploit cheap data or model data distributions for manipulating an image or sampling more images. For generative modeling, quality and diversity are two important measurements for generative models. First, a successful generator is supposed to synthesize data close to the real distribution. Besides, conditional generation is a natural one-to-many mapping task, that ideally produces multiple different outputs, and all of them match the conditional input properly. Further, except generating images, generating intermediate CNN features at decent quality and diversity are also interesting, which have several applications as we demonstrated in this thesis.

Consequently, in Chapter 5, we propose a stochastic regression model and a new loss function to improve the diversity of synthetic images, while maintaining comparable performance with respective to visual quality. We formulate conditional image generation as a regression task. To overcome the lack of diversity and sampling capability of a deterministic architecture, we introduce latent random variables into networks via dropout. Besides, to further improve the diversity, we extend multiple choice learning by sampling neighbors which have similar conditional inputs, and directly approximating the one-to-many mapping. In Chapter 6, we present our dense feature generator based on generative adversarial network, to produce intermediate CNN features for semantic segmentation. We design a specific network architecture for the generator and discriminator of a GAN model for generating spatial compressed features. We show that our synthetic features can be applied as data augmentation to train a semantic segmentation model for improved performance. In Chapter 7, we present the first membership inference attack system for black-box semantic segmentation models, pointing out information leakage of training data frequently happens for semantic segmentation models under various attack settings. To handle the information leakage, we leverage our synthetic features to perform prediction obfuscations on posteriors, which is able to reduce the confidence distribution gap between training and testing data.

5 DIVERSE CONDITIONAL IMAGE GENERATION BY STOCHASTIC REGRESSION WITH LATENT CODES

Recent advances in Deep Learning and probabilistic modeling have led to strong improvements in generative models for images. On the one hand, Generative Adversarial Networks (GANs) have contributed a highly effective adversarial learning procedure, but still suffer from stability issues. On the other hand, Conditional Variational Auto-Encoders (CVAE) models provide a sound way of conditional modeling but suffer from mode-mixing issues. Therefore, recent work has turned back to simple and stable regression models that are effective at generation but give up on the sampling mechanism and the latent code representation. We propose a novel and efficient stochastic regression approach with latent drop-out codes that combines the merits of both lines of research. In addition, a new training objective increases coverage of the training distribution leading to improvements over the state of the art in terms of accuracy as well as diversity.

5.1 INTRODUCTION

Many computer vision and graphics problems can be viewed as a conditional generation problem. For example, we can imagine the appearance of a human face when we only see the shape or the keypoints of the face. Typically, this generation process is not deterministic, as the conditioning information (e.g. keypoints) is insufficient to single out a particular face. Despite the diverse scope of applications for such models, these learning problems remain highly challenging, as an efficient sampling process is required that results in accurate and diverse samples that closely mimic the true conditional distribution.

In particular, *Generative Adversarial Networks (GANs)* (Goodfellow *et al.*, 2014) have contributed in recent years to the state-of-the-art in generative models of such high dimensional output spaces – as we are dealing with in the case of images. These methods allow for sampling by a random generated latent code and the adversarial training leads to highly accurate and realistic samples. The vanilla version of these models lacks the capability for conditional sampling and these methods are known to be notoriously difficult to train and often do not reproduce the full diversity of the training data.

Conditional Variational Autoencoders (CVAE) (Sohn *et al.*, 2015) have been introduced to model a latent code in dependence of the input and by a probabilistic formulation have shown an increased diversity in the generated samples. These models tend to be more stable to train, but still suffer from mode-mixing issues – limiting the success when conditioning data is weak. A series of *Conditional GAN (CGAN)* models

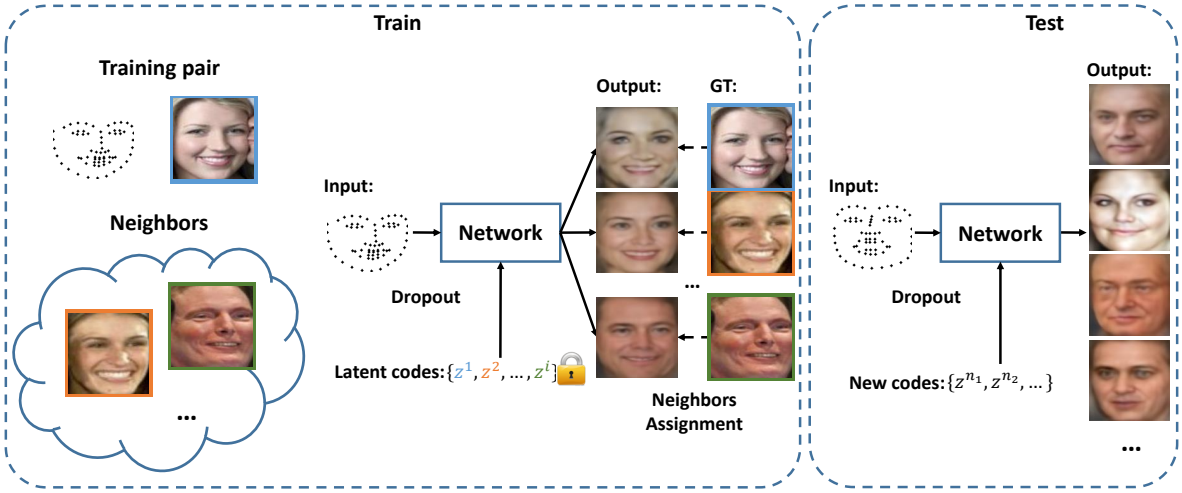


Figure 5.1: The pipeline of the proposed method. We present a stochastic regression with latent dropout codes for image generation, which are fixed during training. At test time, we are able to generate more examples by providing newly sampled codes. Besides, Diversity is further improved by sampling many neighbors considering the condition input when the data distribution is dense enough. With those neighbors, we directly learn the one-to-many mapping by assigning sampled neighbors to different network branches.

(Mirza and Osindero, 2014; Zhu *et al.*, 2017b; Isola *et al.*, 2017) have been proposed that combine ideas of GANs and CVAE (bicycleGAN (Zhu *et al.*, 2017b), pix2pix (Isola *et al.*, 2017)) and thereby achieve some of the increased diversity of CVAE, but yet suffer from some stability problems of GANs.

In order to address the stability issues, generation has been formulated as a regression task (Chen and Koltun, 2017) with *Multiple Choice Learning (MCL)* (Guzman-Rivera *et al.*, 2012). In essence, this re-phrases the conditional generation problem, as a regression task with a fixed number of output samples. This greatly improves the stability of the learning, but no additional samples can be drawn and there is no latent code that represents the samples.

We present a novel solution to the conditional image generation task that is stable to train, has a latent code representation, can be sampled from and results in accurate and diverse samples. We achieve this in a stochastic regression formulation where dropout patterns are conditioned on latent codes. A new training objective increases coverage of the training distribution – resulting in accurate and diverse samples at test time. Our experimental results on two datasets show improvements in efficiency, accuracy and diversity.

5.2 STOCHASTIC REGRESSION WITH LATENT DROP-OUT CODES

Despite the recent progress at the intersection of Generative Adversarial Networks and Conditional Variational Autoencoders, improving stability and increasing the generated diversity are topics of ongoing research. Recent work has shown strong results for conditional image generation in a regression framework that greatly improves stability, but comes with the caveat of no latent code representation, no sampling mechanism and limited diversity in the output (Chen and Koltun, 2017). We seek a model that produces accurate and diverse samples, which is stable to train and provides a sampling mechanism with a latent code representation.
Model: We propose an image generation system that is based on stochastic regression with latent drop-out codes as shown in Figure 5.1. While we are using stable codes to train regression formulation, we are not limiting ourselves to a fixed number of samples due to a fixed number of branches. We rather generate an arbitrary number of branches via dropout patterns derived from a random vector z. In turn, z is characteristic for each sample and acts as a latent code representation.
Training: We sample a set of latent codes – each corresponding to a branch with different dropout pattern that generates one sample. We minimize our new "Neighbors enhanced loss function" which increases the coverage of the training set by those generated samples.
Test: Our model can use both the above training latent codes and newly generated latent codes to produce an arbitrary number of new images. As each images is associated with a latent code, this also allows for additional manipulations like interpolation.

5.2.1 Stochastic Regression with Latent Drop-out Codes

Formally, given an input X, our stochastic regression model produces multiple outputs $\{Y^i\}$ with different dropout patterns controlled by a set of latent codes $\{z^i\}$ which are drawn from a random distribution. Figure 5.2 shows our stochastic model, where $f_1(\cdot)$ and $f_2(\cdot)$ are composed of several convolution, nonlinear, pooling or up-sampling operations. Particularly, $d(\cdot)$ is a function transferring latent codes into binary dropout patterns for selecting features in our network.

Given a latent code z^i, our model can be formulated as

$$Y^i = f_2(f_1(X), d(z^i)). \tag{5.1}$$

Hence, f_1 is shared among all branches and samples, while f_2 depends on the randomized dropout pattern. During training, we fix the latent codes z^i and perform regression on the training set. Yet, in addition to using the training latent codes, we can also use a new code to generate more examples at test time.

Traditional dropout (Srivastava *et al.*, 2014) randomly selects activations of a feature map for all channels and locations. This, however, does not lead to separated branches in a network. Therefore, we propose *channel-wise dropout* that, different to traditional dropout, selects the same features for all the locations. Different channels

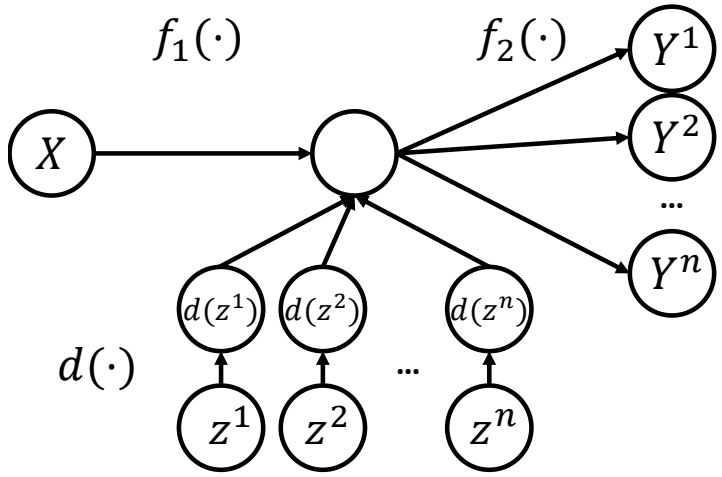


Figure 5.2: Latent codes based stochastic multiple branch model.

usually encode different kinds of visual cues like color, parts or objects as revealed in the research of understanding neural networks (Zhou *et al.*, 2015; Bau *et al.*, 2017). Therefore, our model selects different visual cues for generating multiple outputs with diverse visual properties.

Consider a feature map $I \in R^{C\times H\times W}$ having C channels with size of $H \times W$, the feed-forward operation of channel-wise dropout for the c-th channel at location (h, w) of the i-th latent code can be described as

$$\begin{aligned} & z_c^i \sim U(0,1), \\ & d(z_c^i) = \begin{cases} 1, & \text{if } z_c^i > r \\ 0, & \text{otherwise} \end{cases} \\ & O(c,h,w) = \frac{I(c,h,w) \times d(z_c^i)}{1-r}, \end{aligned} \tag{5.2}$$

where z_c^i is a scalar for the latent code of c-th channel, $r \in (0, 1)$ is the dropout ratio, and $O \in R^{C\times H\times W}$ is the output of our dropout.

Interestingly, we are able to perform interpolation between two generation images in the test as shown in Figure 5.8, and the interpolated output can be described as

$$Y_{ij}(a) = f_2(f_1(X), d(a \cdot z^i + (1-a) \cdot z^j)). \tag{5.3}$$

5.2.2 Neighbors Enhanced Loss Function

In order to improve diversity, we propose a new *Neighbors enhanced loss function* that samples neighbors with respect to a condition input as shown in Figure 5.1. We leverage the sampled neighbors to encourage diversity during training. We update multiple branches for a network by assigning sampled neighbors to different

branches. We first describe a simpler loss function based on the best neighbor (similar to MCL (Guzman-Rivera *et al.*, 2012)) and then our Neighbors enhanced loss function for increased diversity.

Best neighbor loss. With our stochastic model formulated in Eq. 5.1, the scheme generates n hypotheses $\{f_2(f_1(X), d(z^i))|i = 1, 2, ..., n\}$ from the same input X. A simple version would only update the best branch which has the smallest loss. Formally, such a loss function for a batch $\{(X_m, Y_m)|m = 1, 2, ..., M\}$ with size M is defined as

$$L = \sum_{m=1}^{M} \min_i l(f_i(X_m), Y_m), \tag{5.4}$$

where $l(\cdot)$ is a $L1$-based perceptual loss in this paper, defined in Eq. 5.6, and $f_i(X_m) = f_2(f_1(X_m), d(z^i))$ for short.

Neighbors enhanced loss. Given a training pair (X^0, Y^0), we first sample several data pairs $\{(X^i, Y^i)|i = 1, ..., N\}$ satisfying the inputs $\{X^i\}$ are close enough to X^0, i.e., $\{dis(X^i, X^0) < \theta|i = 1, ..., N\}$. We directly approximate the conditional distribution $P(Y|X^0)$ at X^0 by $\{(X^0, Y^i)|i = 0, 1, .., N\}$.

With data pairs $\{(X^0, Y^i)|i = 0, 1, .., N\}$ and n network output hypotheses $\{f_i(X^0)|i = 1, ..., n\}$. We design a neighbors assignment procedure to give a sampled images to one of branches as a ground truth. We assign those sampled neighbors iteratively. Ideally, we hope to assign every samples Y^i to its best hypothesis $f_{best}(X^0)$ where the loss $l(f_{best}(X^0), Y^i)$ is smaller than any other hypothesis. However, there might be more than one sample assigned to the same hypothesis with this condition. To address this issue, we design several assigning rules. First, Y^0 is supposed to assign to its best branch, as (X^0, Y^0) is a well-aligned data pair while others are approximations. Second, because each branch is able to has only one ground truth, we assign the sample with smallest loss if there are more than one sample for the same hypothesis. The matching is proceeded iteratively until all the sampled neighbors are assigned, and output a matching set $S_0 = \{(f_i(X^0), Y^j)\}$.

After applying neighbors assignment, the neural network is optimized with standard back propagation (Rumelhart *et al.*, 1988). We formulate the neighbors enhanced loss function on a batch $\{(X_m, Y_m)|m = 1, 2, ..., M\}$ with the neighbors $\{Y_m^j\}$ as

$$L = \sum_{m=1}^{M} \sum_{(f_i(X_m), Y_m^j) \in S_m} l(f_i(X_m), Y_m^j), \tag{5.5}$$

where S_m is the matching set for m-th example in a batch. Particularly, we utilize $L1$-base perceptual loss (Johnson *et al.*, 2016; Dosovitskiy and Brox, 2016) for optimization in this paper, that is

$$l(X, Y) = \sum_i \lambda_i |\Phi_i(X) - \Phi_i(Y)|, \tag{5.6}$$

where λ_i is the loss weights and Φ_i is the i-th representation form a network Φ.

5.2.3 Architectures and Parameter-Sharing

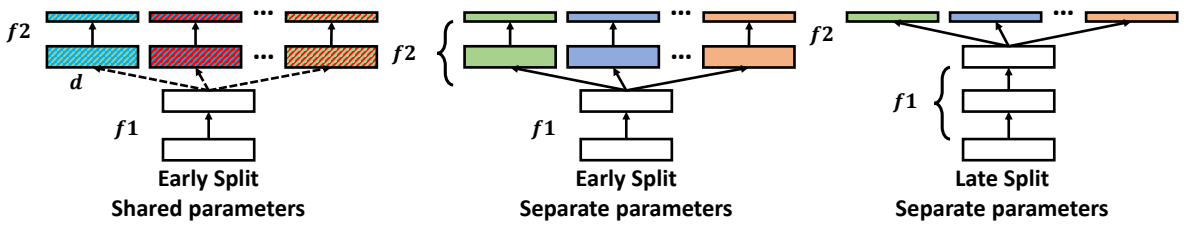


Figure 5.3: The illustration of comparison between different network architectures to produce multiple outputs. Solid lines are deterministic modules and dash lines are stochastic modules.

Our regression model applies channel-wise dropout to split the network into a multiple branches architecture. Because we apply our channel-wise dropout to select different feature maps for multiple branches, we do not have to learn separate parameters for different branches, which will not increase any model size when we increase the number of network branches. Secondly, we embed channel-wise dropout earlier instead of end of a network, which we call "early split" strategy. "Early split" is able to evolute information in a network earlier on a higher dimension, which benefits the generation of diverse examples than the "late split" as shown in Figure 5.3.

When removing our channel-wise dropout, we learn separate parameters for different branches, and thus our full model degenerates to the "early split with separate parameters" setting. In that case we can only generate a fixed number of outputs and need more parameters to represent a model. Comparing to the "late split" setting, used in previous work (Chen and Koltun, 2017), it still has the merit that already intermediate layers contribute to generate different samples, which we show experimentally to be important to generate more diverse images.

Furthermore, our neighbors enhanced loss function can be applied to all architectures in Figure 5.3. Also, CRN (Chen and Koltun, 2017) is a "degernated" case of our approach with deterministic late split network architecture and no neighbors considered.

5.2.4 Discussion

Comparison to Multiple Choice Learning (MCL (Guzman-Rivera et al., 2012)). Our approach generalizes MCL scheme with sampled neighbors and stochastic regression. MCL is a kind of ensemble learning that produces multiple outputs of high quality. In MCL, only one branch gets gradients to update the model. It is therefore not efficient to learn a model due to limited parameter sharing, especially in the case of a large number of network branches to produce highly diverse examples. It also lacks the latent code representation and sampling capability.

5.3 EXPERIMENTS

We compare our approach for conditional image generation to state of the art methods (Isola *et al.*, 2017; Zhu *et al.*, 2017b; Bansal *et al.*, 2018; Chen and Koltun, 2017) on established datasets for generating human faces from facial landmarks and animal heads from normal maps. We systematically compare three different architecture choices in Figure 5.3 under different loss functions (CRN (Chen and Koltun, 2017), "Separate" model and latent codes based "Shared" model) along two dimensions (accuracy and diversity). We implement the proposed network using the *Caffe* framework (Jia *et al.*, 2014).

5.3.1 Datasets and Implementation Details

Oxford-IIIT Pet (Parkhi *et al.*, 2012): It contains 3,868 images of cats and dogs with bounding boxes of animal heads. In our experiments, we use 3,000 images to train a model and 686 images to test the model, which follows previous work (Bansal *et al.*, 2018). We first use bounding boxes to crop the animals' heads and resize them into 96 $\times$ 96 pixels. Finally, we utilize PixelNet (Bansal *et al.*, 2017) to estimate the normals.

LFW (Learned-Miller *et al.*, 2016): We utilize the deep funneling aligned LFW dataset, and apply the peopleDevTrain/peopleDevTest split containing 4,038 and 1,711 images to train and evaluate the performance. For each image, we first employ the MTCNN (Zhang *et al.*, 2016) face detection model to extract faces. Next, we employ the TCDCN (Zhang *et al.*, 2014) extracting 68 facial landmarks for each face, and use the heat map of key points as the input of the network. For all the faces, we resize the bounding box regions into 128$\times$128 pixels, and thus we generate 128$\times$128 color faces from the input with size 128 $\times$ 128 $\times$ 68.

Implementation details. We implement our channel-wise dropout and networks with the Caffe (Jia *et al.*, 2014) deep learning framework. We set dropout ratio as $r = 0.5$ for all the channel-wise dropout in our experiments. For all models, we apply Adam (Kingma and Ba, 2015) to optimize our models and use the "poly" learning rate policy that current learning rate is $lr_{init} \times (1 - \frac{iter}{iter_{max}})^{power}$. And we set power as 0.9 and initial learning rate lr_{init} as 1×10^{-4}. We set max iterations number as 110,000 and 90,000 for animal head generation and face generation tasks, respectively. To have a fair comparison of separate parameters networks and shared parameters networks, we split feature representation or introduce channel-wise dropout at the same location. For CRN and our models, we generate 96$\times$ 96 images for head animal generation task, and generate 128$\times$ 128 images for face generation.

5.3.2 Animal Head Generation from Normal Map

In this experiment, we test three kinds of architectures: For CRN and "Separate" model, we learn a set of models with 2, 4, 8, 20, 30, 40, 50, 72 branches. For our "Shared" models, we learn 4, 8, 20, 72 branches to test.
Quantitative results and analysis. In terms of accuracy, we apply root mean square error (RMSE), SSIM (Wang *et al.*, 2004) and FSIM (Zhang *et al.*, 2011) as the measurements for appearance similarity. Besides, we also evaluate the consistency of predicted normals from generated images and real images with 6 evaluation criteria following (Bansal *et al.*, 2018; Wang *et al.*, 2015). We report the performance against strong competing methods pix2pix (Isola *et al.*, 2017), BicycleGAN (Zhu *et al.*, 2017b), PixelNN (Bansal *et al.*, 2018) and CRN (Chen and Koltun, 2017) on the accuracy of normal prediction with generated images. For all those methods (Chen and Koltun, 2017; Isola *et al.*, 2017; Bansal *et al.*, 2018; Zhu *et al.*, 2017b), we choose the best example among all 72 generated heads and compare results in Table 5.1. We also report the performance of our final "Shared" models with different branches. We can observe that our model with 72 branches achieves best performance . And our 4 branches model also achieves better results than pix2pix (Isola *et al.*, 2017), BicycleGAN (Zhu *et al.*, 2017b) and PixelNN (Bansal *et al.*, 2018), and only 6% worse than CRN (Chen and Koltun, 2017).

Table 5.1: Comparison of predicted normals of best generated animal heads.

Method	Mean	Median	RMSE	11.25°	22.5°	30°
pix2pix (Isola *et al.*, 2017)	13.2	11.4	15.7	49.2	87.1	95.3
BicycleGAN (Zhu *et al.*, 2017b)	21.6	19.3	24.9	24.3	60.2	77.5
PixelNN (Bansal *et al.*, 2018)	13.8	11.9	16.6	46.9	84.9	94.1
CRN (Chen and Koltun, 2017)	11.8	10.3	13.9	56.3	91.4	97.6
Ours-4	12.4	10.9	14.5	52.9	90.0	97.0
Ours-8	12.4	10.8	14.4	53.6	90.2	97.1
Ours-20	12.0	10.5	14.1	55.2	91.1	97.6
Ours-72	**11.7**	**10.2**	**13.7**	**56.7**	**91.9**	**97.8**

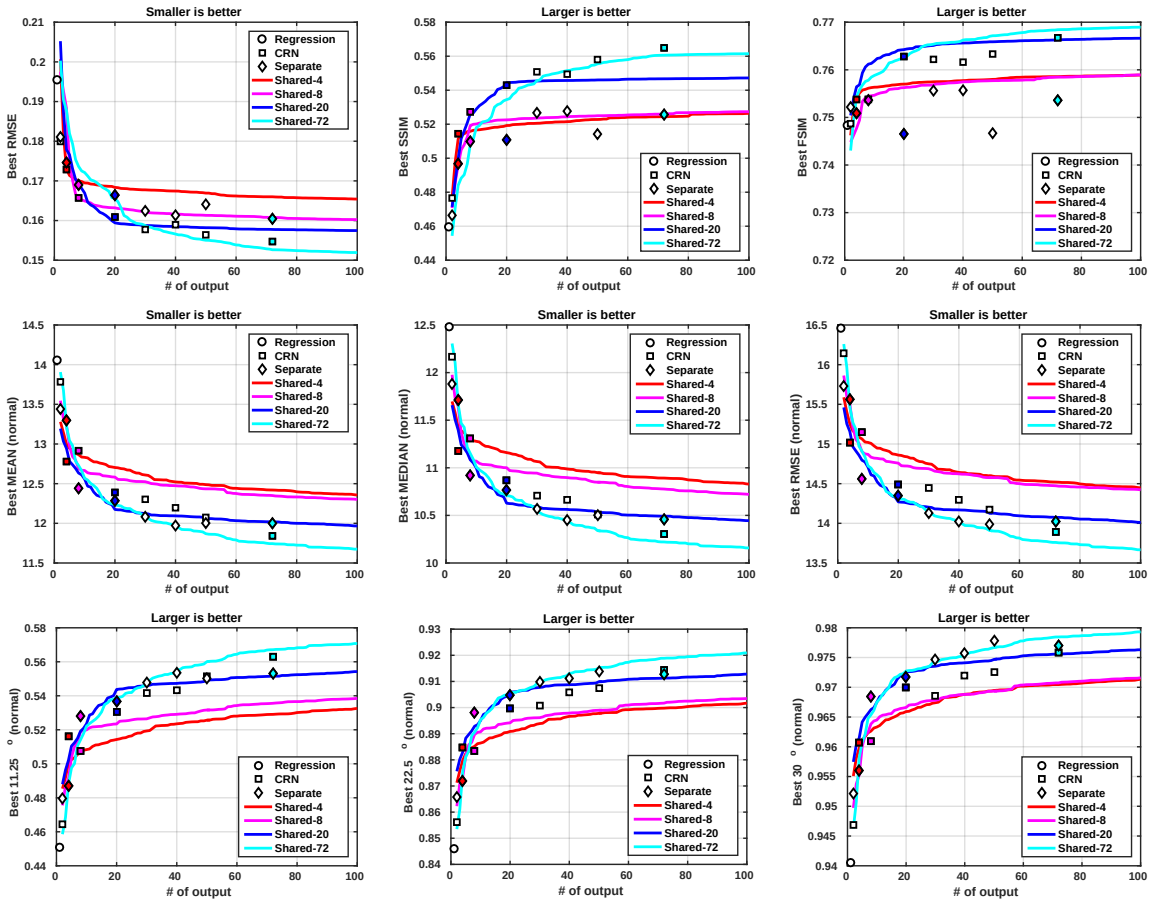


Figure 5.4: Evaluation of best example on Oxford-IIIT Pet dataset. The first row draws the evaluation plot for appearance similarity. The bottom two rows show the evaluation plots for the consistency of predicted normals between generated images and real images. “Regression" model produces one output image. For CRN and “Separate" models, we learned eight models (2, 4, 8, 20, 30, 40, 50, 72 branches) to test them. For “Shared” model, we learned four models(4, 8, 20, 72 branches) and samples totally 100 outputs during test time. Best viewed in color.

Next, we provide comparison plots for different architectures including CRN, our “Separate” model and our “Shared” model with channel-wise dropout as shown in Figure 5.4. In this figure, we show the best performance for all 9 evaluation metrics used in our experiments. For our “Shared" models, we use both the training latent codes and newly sampled codes to generate 100 outputs. Differently colored curves show the performance of sampling different number of examples. First, we observe that the performance of the best example gradually increases for all the architectures when the number of branches is increased. Second, we can see that the performance of our “Shared” model is better than other two when the number of branches is large, which means our models generate better results with a large number of branches even without sampling more outputs. Finally, we observe that the red, pink, blue and cyan lines always become better. This means our latent codes based regression

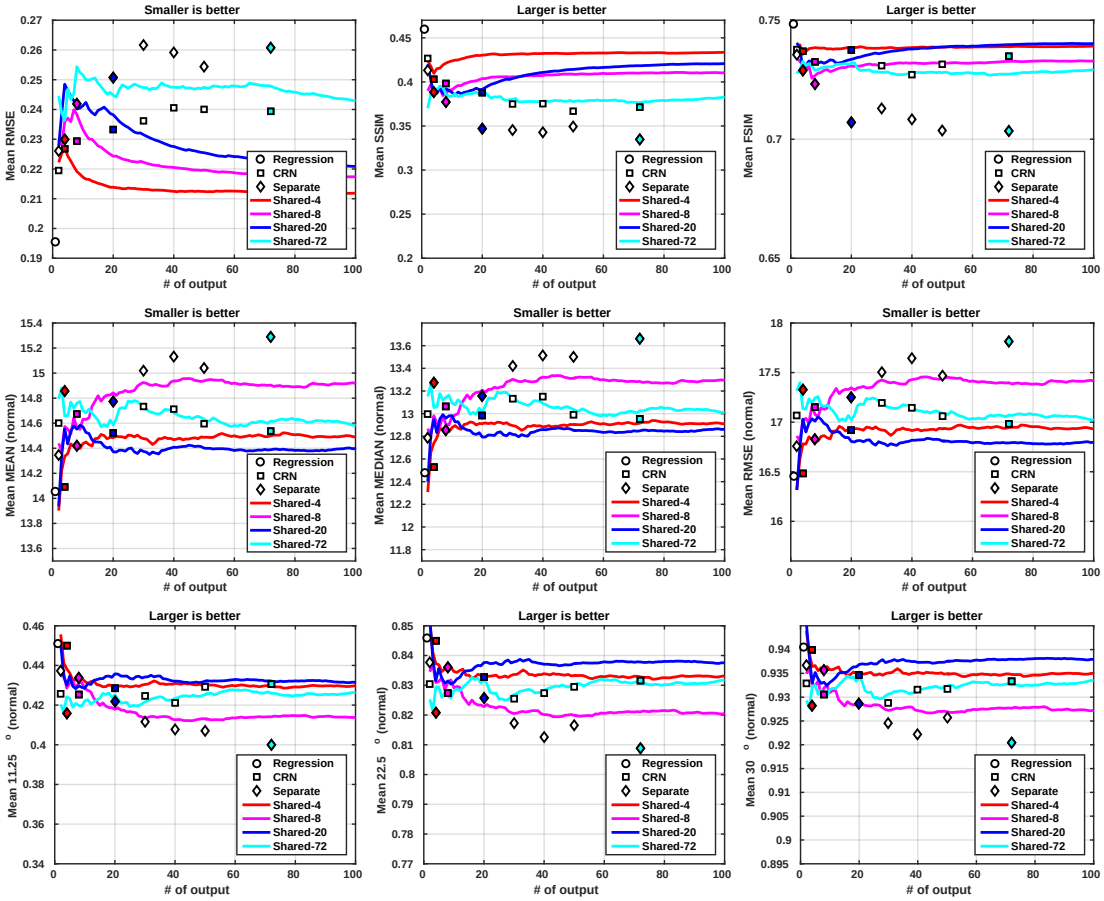


Figure 5.5: Evaluation of average performance of all the generated images on Oxford-IIIT Pet dataset. We utilize the same metrics to Figure 5.4.

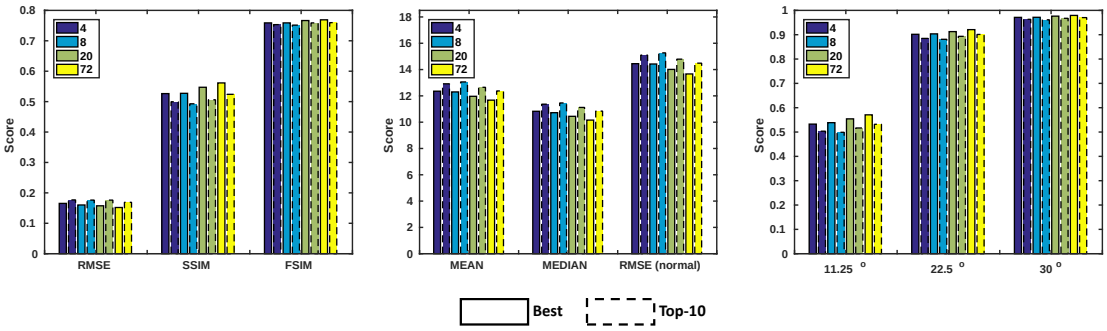


Figure 5.6: Comparison of best example and top-10 examples on Oxford-IIIT Pet dataset. For RMSE, MEAN and MEDIAN, smaller is better. For other measurements, larger is better.

models generate better examples with new codes, even though they have never been used during training.

To further demonstrate the overall quality of samples drawn by our models, we

also analyze three plots of average performance in Figure 5.5 on the appearance similarity and the consistency of normal prediction. We observe that all of our "Shared" models maintain their performance comparing to the case of using training latent codes. Besides, we also report the performance of top-10 examples comparing to the best one in Figure 5.6. This figure shows that the average performance of top-10 examples is very similar to the best number, which is another evidence for the effective sampling of our approach.

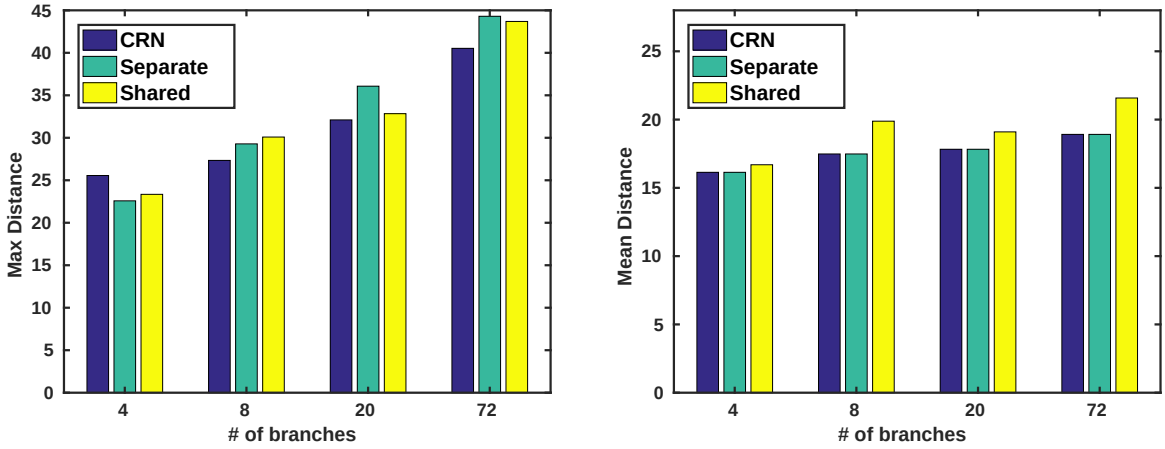


Figure 5.7: Evaluation of diversity on Oxford-IIIT Pet dataset. It reports the max distance and mean distance to the average of generated images.

Diversity analysis. To evaluate the diversity quantitatively, we compute the max distance and mean distance of all the generated images to their center, and present the comparison in the Figure 5.7. The results show that an early split improves diversity over the late split, which is used in CRN (Chen and Koltun, 2017). Besides, this figure also shows that our "Shared" model performs comparably to our "Separate" model, even it has smaller model size and is able to generate an arbitrary number of examples at test time. Beyond sampling, we also show in Figure 5.8 interpolation results using Eq. 5.3. The smooth transition between the samples corresponding to z_1 and z_2 gives evidence that the latent codes indeed serves as a meaningful representation.

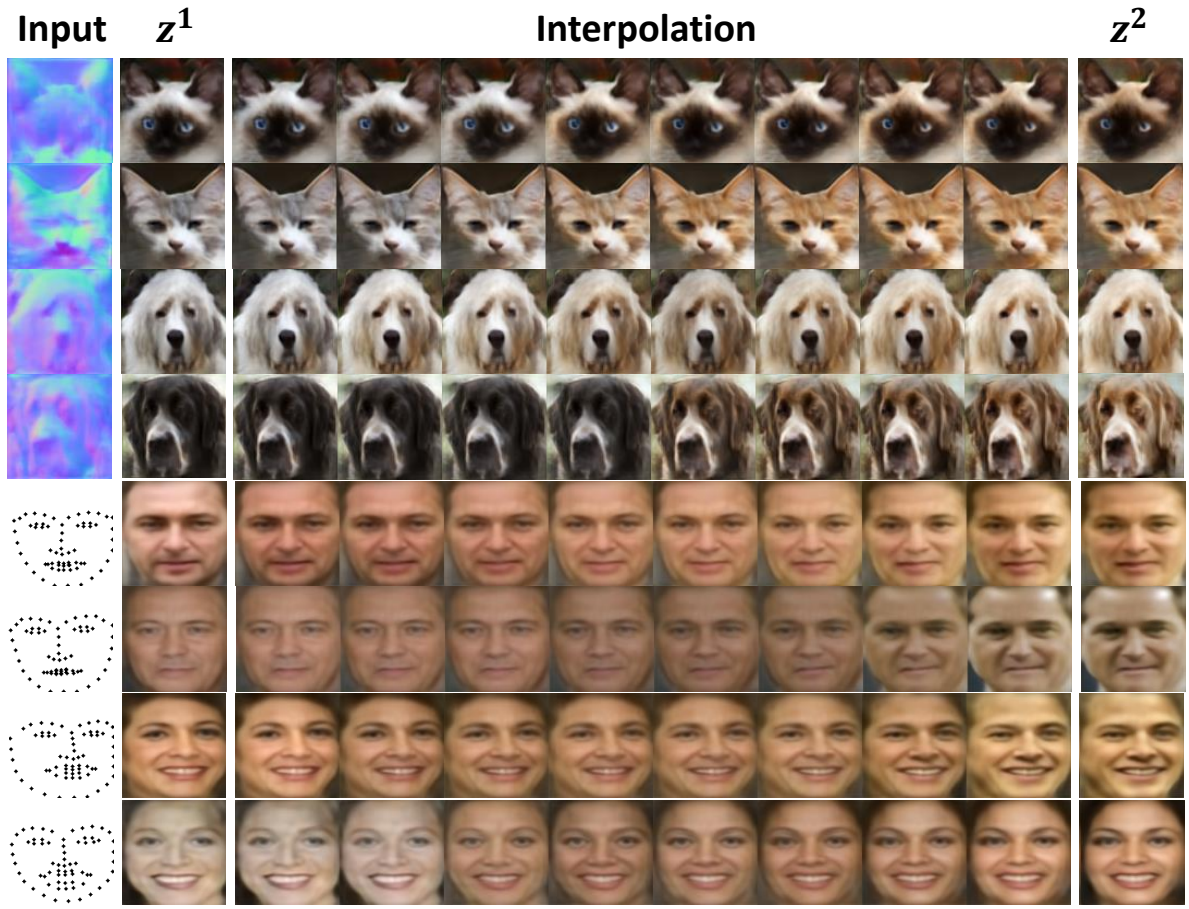


Figure 5.8: Interpolation results from the proposed architecture using channel-wise dropout. Images below z^i correspond to the respective latent code.

Channel-wise dropout vs. classic dropout. Traditional dropout (Srivastava *et al.*, 2014) can be also used in our architecture. However, it cannot select or reject a whole feature channel like our channel-wise dropout. We provide a comparison between our channel-wise dropout and traditional dropout in the proposed model. Figure 5.9 shows some visualization results, which clearly demonstrate that channel-wise dropout successfully selects different features and then generates animals with different color, while dropout generate more similar animal heads.

Figure 5.9: Channel-wise dropout vs. dropout. In each block, top row shows the results from the network with channel-wise dropout, and bottom row shows the results applying dropout.

Figure 5.10: The example and comparison with competing methods (Isola *et al.*, 2017; Zhu *et al.*, 2017b; Chen and Koltun, 2017) on the task of face generation from landmarks. Best viewed in color.

5.3.3 Face Generation from Facial Landmarks

In this task, we generate human faces from 68 facial landmarks. For both models, we generate 10 output faces from the networks. Except evaluating on the architectures, we also test our neighbor assignment strategy for improving diversity. After acquiring neighbors, we use kmeans on HSV color space to cluster the neighbors. For each cluster, we randomly select an example as the sampled neighbor. We use $L1$ distance between the coordinates of two landmarks.

Qualitative comparison. We show generated faces in Figure 5.10. For each of the six blocks, the left two images show the facial landmarks and the corresponding image. Next to them results of pix2pix (Isola *et al.*, 2017), BicycleGAN (Zhu *et al.*, 2017b), CRN (Chen and Koltun, 2017) and our "Shared" model are shown row by row. Top two blocks show "normal" case, middle blocks show landmarks of immediate difficulty and bottom blocks show the hard examples. In the top two blocks, we observe that our model is not only able to generate faces with different skin, but also to generate faces from different identity or gender, or with different local details. While the baseline CRN (Chen and Koltun, 2017) just generates faces with very similar structure, even different colors of skin are also covered. Second, comparing to pix2pix (Zhu *et al.*, 2017b) and BicycleGAN (Zhu *et al.*, 2017b), which try to apply GAN to generate realistic faces, we also generate faces with better quality. Although BicycleGAN can generate many visually realistic faces, it also generates some very

Table 5.2: Accuracy comparison of face image generation on LFW dataset. We report the average as well as the best performance (Mean / Best) on four measurements.

Method	Nbs.	RMSE	SSIM	FSIM	Landmarks
pix2pix (Isola *et al.*, 2017)		9.952/9.649	0.696/0.708	0.749/0.755	1.913/1.645
BicycleGAN (Zhu *et al.*, 2017b)		9.948/6.856	0.621/0.715	0.725/0.758	1.704/2.197
CRN (Chen and Koltun, 2017)		9.107/5.414	**0.705**/**0.767**	**0.760**/**0.779**	**1.697**/1.452
Our CRN	✓	**8.612**/5.309	0.687/0.763	0.754/0.777	1.863/1.440
Ours (Separate)		8.871/5.528	0.704/0.761	0.755/**0.779**	1.718/**1.308**
Ours (Separate)	✓	9.034/5.768	0.678/0.757	0.748/0.774	1.915/1.442
Ours (Shared)		8.908/**5.269**	0.701/0.762	0.757/**0.779**	1.728/1.365
Ours (Shared)	✓	8.956/5.679	0.686/0.764	0.753/**0.779**	1.864/1.453

poor results due to its unstable training. Finally, observing the last row, we even generate plausible faces in hard cases.

Accuracy analysis. We evaluate our method and compare to pix2pix (Isola *et al.*, 2017) and CRN (Chen and Koltun, 2017). For pix2pix (Isola *et al.*, 2017) and BicycleGAN (Zhu *et al.*, 2017b), we run the code provided by authors with default settings except increase the training epochs from 200 to 300. For CRN, we use the same number of feature maps and branches to ours.

For accuracy, we apply root RMSE, SSIM (Wang *et al.*, 2004) and FSIM (Zhang *et al.*, 2011) to evaluate the appearance similarity between generated faces and ground truth faces. Besides, we also run the landmark detector (Zhang *et al.*, 2014) on the generated faces and compare the accuracy of detected landmarks and ground truth landmarks. The accuracy of landmarks is measured by the sum of distance of each predicted points to ground truth normalized by the width of the images. We report the best performance as well as average performance among 10 outputs as summarized in Table 5.2.

From Table 5.2, we can observe that our models and CRN achieves better performance than pix2pix and BicycleGAN in all four metrics on best performance and mean performance, benefiting from the cascaded refinement network architecture and the effectiveness of perceptual loss. Comparing to CRN, the overall performance of our “Shared" and “Separate" models are comparable in four metrics. Even the quantitative performance decreases a little in SSIM and landmarks accuracy after applying neighbors enhanced loss function, the decrease is just a little and we found that our approach can generate better faces visually as shown in Figure 5.10.

Diversity analysis. To quantify the measurement of diversity for multiple output from the same landmarks, we compute standard deviations of different levels of face representation from Wen *et al.* (2016) and an identity embedding for face recognition from Oh *et al.* (2017). Table 5.3 lists the scores for competing approaches. Clearly, our “Separate" and “Shared” models achieves better diversities than CRN (Chen and Koltun, 2017). Besides, we also observe that the standard deviations get consistent and significant improvements for CRN, “Separate" and “Shared” models, when we

Table 5.3: Standard deviation of the convolutional features from (Wen *et al.*, 2016) and the identity embedding from (Oh *et al.*, 2017). For all the values, larger is better.

Method	Nbs.	pool1	pool2	pool3	pool4	fc5	identity
pix2pix (Isola *et al.*, 2017)		0.134	0.359	0.364	0.193	0.370	0.660
BicycleGAN (Zhu *et al.*, 2017b)		**0.428**	**0.817**	**0.716**	0.350	0. 696	**2.055**
CRN (Chen and Koltun, 2017)		0.198	0.337	0.350	0.186	0.389	1.098
Our CRN	✓	0.272	0.549	0.591	0.322	0.669	1.299
Ours (Separate)		0.225	0.491	0.551	0.302	0.610	1.235
Ours (Separate)	✓	0.276	0.596	0.640	**0.353**	**0.733**	1.427
Ours (Shared)		0.210	0.422	0.471	0.259	0.523	1.198
Ours (Shared)	✓	0.310	0.591	0.617	0.322	0.670	1.370

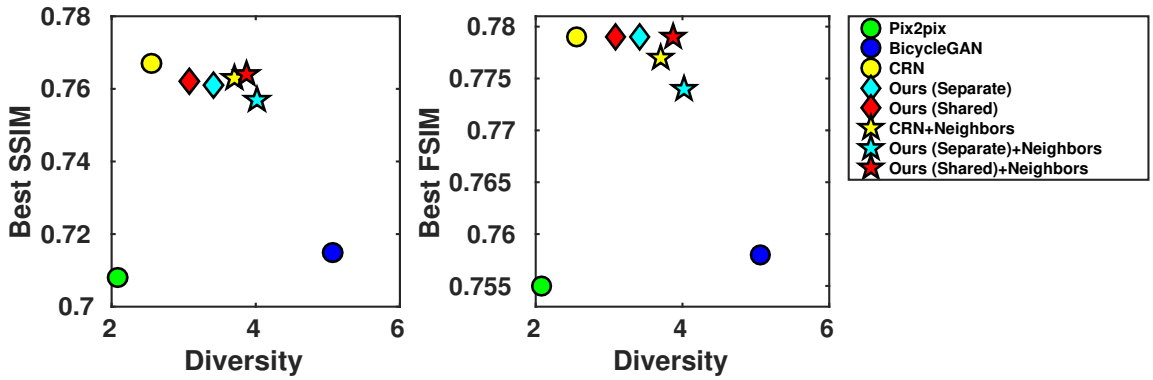


Figure 5.11: Accuracy-Diversity plots on LFW dataset. The diversity score is the sum of stand deviation of all the representations in Table 5.3.

apply our neighbor enhanced loss function. It clearly shows that the effectiveness of assigning sampled neighbors. Particular, recent proposed BicycleGAN model has the best score on $pool_1$, $pool_2$, $pool_3$ and *identity*, but the accuracy of BicycleGAN is not comparable to our generation results. More importantly, diversity and accuracy are often a trade-off. To observe this relationship clearly, we plot a accuracy-diversity scatter plot in Figure 5.11, whose top-right corner is best. It shows that our stochastic regression model and neighbors enhanced loss function both contribute to generating more diverse example while maintain a comparable accuracy.

5.4 DISCUSSION

Our method presents a stochastic model for generating multiple diverse examples from a same conditional input, by formulating image generation as a regression problem, which is quite stable in training. Second, we show our network allows to sample numerous novel examples with high quality. Final, our experimental results

also show that the proposed neighbor enhanced loss function is helpful to generate examples with stronger diversity, such as generating various genders, skin colors, facial expressions etc.

5.4.1 Application Scenarios

The main advantage of our model over baseline deterministic architecture is to sample multiple novel examples during inference time, which is controlled by a vector of random variables. Our method formulates conditional image generation as a regression task, and the regression based generation pipelines are normally combined with popular generative adversarial networks (Goodfellow *et al.*, 2014) to synthesize realistic images.

Besides, we also show that our neighbor enhanced loss function allows a network to learn one-to-many mapping directly when enough neighbors are acquired, and thus generate more diverse images. The learning framework with neighbors could be applied in several application scenarios, such as human body generation from key-points or layouts (Ma *et al.*, 2017), 3D shape generation from class labels (Dosovitskiy *et al.*, 2015), in which similar conditional inputs can be searched.

5.4.2 Technical Limitations

Recently, disentangled representations are popular, since they provide explanation of learned representations, and allow for more interesting applications, such as image editing, or generating images with an reference image. However, our approach is not able to fulfill such tasks, because our randomly sampled dropout patterns and their corresponding features are not disentangled.

Besides, taking face generation from landmarks as an example, even though our method is able to produce high quality faces, we cannot guarantee our model produces stable faces of a same person with continuous landmark inputs. Therefore, for more practical applications on videos, we need to adjust our model with the consideration of temporary consistency.

5.5 CONCLUSION

In this paper, we have presented a novel image generation approach, which learns to produce multiple diverse examples from a single conditional input. We have tested our method on the tasks of generating human faces from facial landmarks and animal heads from normal maps. Based on a series of ablation studies and comparisons with state-of-the-art image generation frameworks, we demonstrate the effectiveness of enforcing diversity by sampling neighboring examples, and efficiency of our network architectures by introduce channel-wise dropout to randomly select feature maps to generate accurate and diverse images of various styles and structures.

6 SYNTHETIC DENSE FEATURES FOR IMPROVING SEMANTIC SEGMENTATION

Recently, learning-based image synthesis has enabled to generate high resolution images, either applying popular adversarial training or a powerful perceptual loss. However, it remains challenging to successfully leverage synthetic images for training of semantic segmentation. Therefore, we argue to generate instead intermediate feature representations and propose the first synthesis approach that is catered to such dense intermediate representations. This allows us to generate new features from label masks and include them successfully into the training procedure. Experimental results and analysis on two challenging datasets *Cityscapes* and *ADE20K* show that our generated feature improves performance on segmentation tasks.

6.1 INTRODUCTION

Semantic image segmentation is a fundamental problem in computer vision, and has many applications in scene understanding, perception, robotics and in the medical area. To achieve robust performance, models usually are trained with data augmentation like flipping and re-scaling to make full use of expensively annotated data.

Recent work leverages synthetic images for data augmentation for computer vision tasks benefiting from capable graphic engines and development of generative modeling (Goodfellow *et al.*, 2014), e.g. for gaze estimation (Shrivastava *et al.*, 2017) and hand pose estimation (Mueller *et al.*, 2018). However, using synthesized images for semantic segmentation remains challenging, because of the complexity of scenes, and exponential combinations of different elements. Previous work on semantic segmentation with synthetic data (Hoffman *et al.*, 2016; Sankaranarayanan *et al.*, 2018; Tsai *et al.*, 2018; Saleh *et al.*, 2018; Wu *et al.*, 2018) focuses on domain adaptation problems that aim to reduce the distribution gap between synthetic images and real images, instead of improving a segmentation model trained with fully annotated real data. Besides, even though high resolution realistic generated images (Chen and Koltun, 2017; Qi *et al.*, 2018; Wang *et al.*, 2018b) are acquired, they do not show better segmentation results of training with those generated images, comparing to training with real images. When inspecting these generated images visually (Wang *et al.*, 2018b; Qi *et al.*, 2018), there are still some visual artifacts, which affect the low-level convolutional layers significantly. Learning with those regions, low-level representations are probably degenerated, and thus high-level representations are also hard to effectively build on top of them. Therefore, training with such images

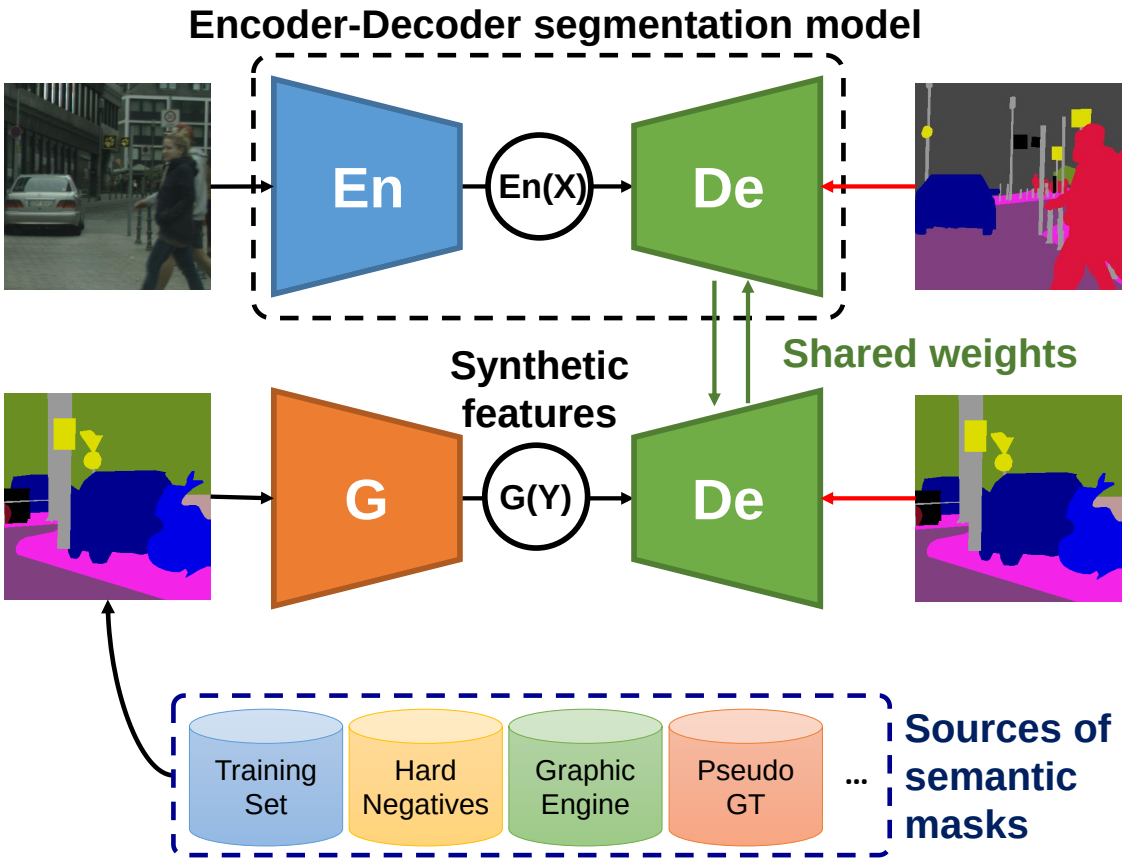


Figure 6.1: Our pipeline for semantic image segmentation. We learn a generator to synthesize convolution features for data augmentation in semantic segmentation. A semantic segmentation model is learned with synthetic features as well as real images. Real images are used to update the entire network, and synthetic features are used to update the decoder of the network.

might lead to decreased segmentation performance.

Because of the difficulty of image synthesis, we question if we really need to synthesize images for boosting the training of semantic segmentation models. Instead, we present a feature synthesis-based data augmentation approach for semantic segmentation, as shown in Figure 6.1. We aim to learn a semantic segmentation model with a mixture of real images and synthetic data from a generator, allowing to sample paired data from semantic layout masks, which assign categories for each pixel. Modern semantic image segmentation models built on fully convolutional architectures, as a result, we can extract features at different locations. Our goal is to find a suitable location, whose output feature is easily synthesized and able to used as training data to improve semantic segmentation.

Different to image synthesis, the output of synthesized features have lower spatial dimension but a larger number of channels. Consequently, it is hard to apply existing image synthesis architectures to the feature generation task, and thus a new effective

architecture is needed. The capacity and quality of data is the key to success for data augmentation. Hence, a good feature synthesis architecture enables dense feature generation with the following requirements. It allows us sample multiple diverse features from one semantic mask input, and thus provides us numerous training examples. Besides, those synthetic features should follow a similar distribution as extracted features from real images. In other words, the synthetic features are able to be segmented by a trained model, with comparable performance to real features. The final challenge is that raw images contain many detailed information which are compressed in the feature domain. Our architecture should be powerful enough to represent those important details. To achieve this goal, we design a generator under the framework of multi-modal translation (Zhu *et al.*, 2017b) with a network architecture catered to the dense feature synthesis task.

The main contributions of this work are: (1) We propose to synthesize convolution features for data augmentation for semantic image segmentation, leading to improved results; (2) We present an effective feature generative model, whose effectiveness is shown according to a series of ablation studies; (3) Several techniques are proposed to leverage the synthetic features, including online hard negative mining, generation from additional masks, and label smoothing regularization.

6.2 DATA AUGMENTATION WITH SYNTHETIC FEATURES

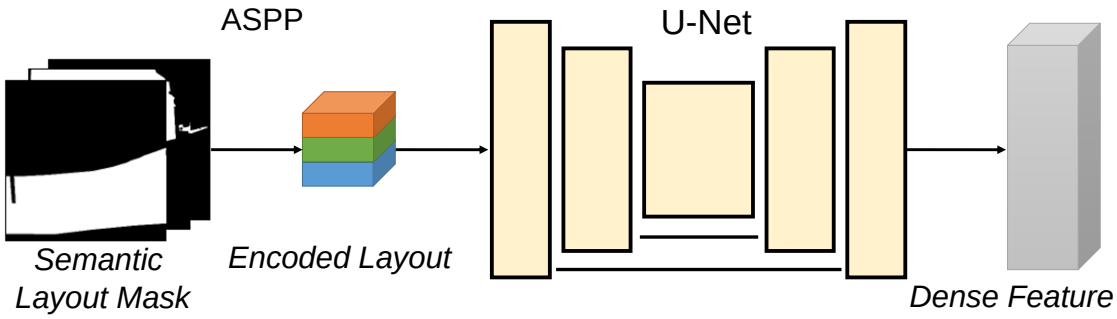
Generation and classification are reverse problems, which translate between images and labels each other. With a paired training set $\mathcal{T} = \{(X^i, Y^i)\}_{i=1}^n$, we can learn a segmentation model $Y^i = S(X^i)$ as well as an image generator $X^i = G_{img}(Y^i)$. Naturally, it is able to train a segmentation model with the augmented dataset $\mathcal{T} \cup \{G_{img}(Y^i), Y^i\}_{i=1}^n$. However, it is hard to guarantee improved performance due to quality of generated images.

Instead, we generate convolutional features for providing more data to segmentation models, and our pipeline is presented in Figure 6.1. Semantic segmentation model S is learned from $\mathcal{T}$, and consist of encoder En and decoder De, as a result, we can extract features for an image by $En(X)$ and segment the image by $De(En(X))$. Hence, our goal is: (1) to learn a generator G_{feat} which is able to produce realistic features, formally $p(En(X)|Y) \sim p(G(Y)|Y)$; (2) to learn the parameters for the decoder $De(\cdot)$ with synthetic training pairs $\{G_{feat}(Y^i), Y^i\}_{i=1}^n$ as well as real pairs $\{En(X^i), Y^i\}_{i=1}^n$.

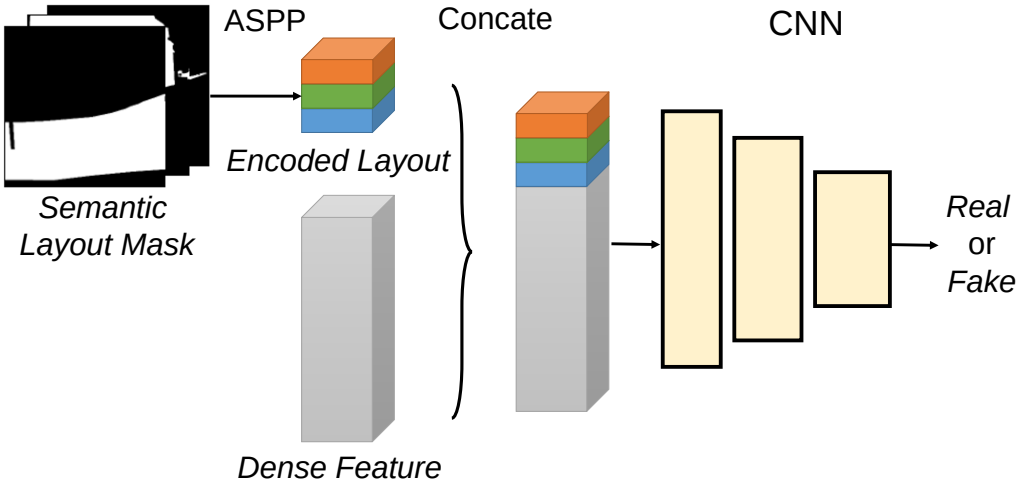
We train a whole model with mixture of synthetic features and real images. The encoder, which captures low-level detailed features, is updated with real images only. The decoder is shared by synthetic and real features, and both branches contribute to the updating of parameters in decoder. Formally, the loss function is

$$\mathcal{L} = \mathbb{E}(-\log De(En(X))) + \mathbb{E}(-\log De(G_{feat}(Y))), \tag{6.1}$$

where the output of De is per class probabilities for each pixel normalized with softmax, and the feature generator G_{feat} is already successfully trained with $\mathcal{T}$ and $En(\cdot)$.



(a) Generator.



(b) Discriminator.

Figure 6.2: The illustration of network architectures in our dense feature generative adversarial networks (DFGAN).

6.2.1 DFGAN: Dense Feature Generator

We present our model for effective feature synthesis, which is crucial for our goal, as discussed in section 6.1. Generating multiple diverse features is challenging, because they encode information of large areas as well as details, which cannot be ignored. Also, the synthetic features should follow a similar distribution as extracted real features. We formulate our generator leverage the recently proposed BicycleGAN (Zhu *et al.*, 2017b) objective, which is shown to be successful in one-to-many image translation tasks. In the objective, the generation is driven by a conditional input and a random latent vector, allowing us to sample multiple different examples. Reconstructions on latent vectors and input features guarantee the quality of sampled features. Finally, an adversarial loss helps to generate features with useful details.

Architecture.. It turns out that previous synthesis approaches are not directly applicable to feature synthesis, as they emit an output with the same dimension of the input. Dense features are compressed from images, with smaller spatial dimension, but much larger channel number, encoding location information and many useful

details (Ghiasi and Fowlkes, 2016). Hence, representing such information correctly plays an important role to successful feature synthesis. As shown in Figure 6.2, our generator takes a high resolution semantic layout mask input to produce low resolution feature maps. Our discriminator takes a layout/feature pair as input, to judge if the feature is compatible with the layout or not.

Preserving resolution by atrous pooling.. Atrous spatial pyramid pooling (ASPP) is an effective module in semantic segmentation used to aggregate multi-scale context information (Chen *et al.*, 2018b). Here, we take advantages of ASPP in multi-scale representation capability, and effectively encode a high resolution semantic layout input. In our ASPP module, there are three convolution layers with dilation 1,2, and 4, to capture neighboring information and wider context. Stride operation is followed after ASPP, leading to downsampled resolution. After applying several ASPP modules (depends on the resolution of synthesized features), the encoded semantic layout reaches to the same spatial dimension as the features. We feed encoded semantic layout into a U-Net (Ronneberger *et al.*, 2015) and output final dense features.

In our discriminator for adversarial training, we also apply ASPP module to encode high resolution semantic layout mask. We concatenate the encoded layout and its corresponding real/fake feature together to classify the feature/mask pair as real or fake.

6.2.2 Regularization for using synthetic features

Label smoothing regularization (LSR) has been shown to reduce the influence of noisy labels and improve generalization (Szegedy *et al.*, 2016). Because not all sampled synthetic features are perfect, we apply LSR to train a model with synthetic features, as shown in Figure 6.1.

In Eq. (6.1), the per class probabilities for X are expressively described as $p_i(k|X) = \frac{\exp(r_i^k)}{\sum_{k=1}^{K}\exp(r_i^k)}$ for each label $k \in \{1, \cdots, K\}$, where r_i^k is the unnormalized log probabilities for k-th class, indexing at i-th location. Similarly, the per class probabilities for synthetic feature $G_{feat}(Y)$ are $p_i(k|G_{feat}(Y)) = \frac{\exp(s_i^k)}{\sum_{k=1}^{K}\exp(s_i^k)}$. The negative log likelihood in Eq. (6.1) can be rewritten as

$$\begin{aligned}\mathcal{L} = \mathbb{E}(-\sum_i \log p_i(k|X) q_{real}(k)) \\ + \mathbb{E}(-\sum_i \log p_i(k|G_{feat}(Y)) q_{syn}(k))\end{aligned} \tag{6.2}$$

with weighting functions $q_{real}(k)$ and $q_{syn}(k)$ for the branches using real images and synthetic features, respectively. For cross entropy loss, it only takes the probability for designed label; for the version with LSR, it takes all the probabilities to compute

a loss. They can be formulated in an unified formulation, i.e.,

$$q_\epsilon(k) = \begin{cases} 1 - \frac{K-1}{K}\epsilon, & k = y \\ \frac{\epsilon}{K}, & k \neq y, \end{cases} \tag{6.3}$$

where ϵ is a small value in the range of (0,1) for label smoothing regularization. It will become cross entropy when $\epsilon = 0$. As a result, we set $q_{real} = q_0$ and $q_{syn} = q_\epsilon$ in Eq. (6.2).

6.2.3 Online hard negative mining

Except generating features by selecting layouts randomly, we can search hard examples during training, which have a large loss value. We do online hard negative mining and feature generation alternatively. We randomly sample some image patches and compute their loss value, the top ranking patches are used to generate the features for the next several training iterations.

6.2.4 Additional semantic masks

Since we generate paired data from semantic layout masks, it is possible to acquire more data than augmenting a training set only, by providing novel mask. For example, in traffic scenarios, the environment is fixed, but everyday road users are different. It is interesting to know if segmentation model can be further improved by seeing more combination of road users and still objects. We present more semantic masks from different sources in Table 6.4 of section 6.3.2.

6.3 EXPERIMENTS

6.3.1 Experimental settings

We evaluate our data augmentation scheme using PSPNet (Zhao *et al.*, 2017) on the *Cityscapes* (Cordts *et al.*, 2016) and *ADE20K* (Zhou *et al.*, 2017) datasets. *Cityscapes* captures traffic scenes in various cities under different weather and illumination conditions containing 2975 training image pairs with detailed annotations, and 19998 extra images with coarse annotations. Except still frames, it also provides a short video for each frame. *ADE20K* has 20210 training images at different image resolutions including a variety of indoor and outdoor scenes. We evaluate our approach with widely used measurements for semantic segmentation for all the datasets including pixel accuracy (PixelAcc), class accuracy (ClassAcc), mean intersection over union (mIoU) and frequent weighted intersection over union (fwIoU).

Implementation details. We implement our generator with the modification of (Zhu *et al.*, 2017b) using the PyTorch framework. We implement our segmentation

Table 6.1: Comparison of utilizing different synthetic data on the *Cityscapes* validation set.

Models	Mask	PixelAcc	ClassAcc	mIoU	fwIoU
Baseline		96.34	86.34	79.73	93.15
Images (Zhu *et al.*, 2017b)	train	95.84	82.54	76.55	92.17
Images (Wang *et al.*, 2018b)	train	96.21	85.61	79.52	93.07
Images (Qi *et al.*, 2018)	val	96.33	85.99	79.60	93.11
Ours	train	96.40	87.29	80.30	93.27
Ours	train+val	96.40	87.47	80.33	93.29

model with synthetic features under the official PSPNet implementation (Zhao *et al.*, 2017), and apply released ResNet-101 and ResNet-50 based PSPNet as our baselines for *Cityscapes* and *ADE20K*. We extract 3000 and 40000 patches on for the conv4_12 and conv4_3 layers to train the generator for *Cityscapes* and *ADE20K*, respectively. We set 60 and 20 epochs for those datasets. With a learned generator, we finetune the segmentation model from our baseline. All models are learned using SGD with momentum, and the batch size is 16. Initial learning rates are set to 10^-6 and 10^-7 for *Cityscapes* and *ADE20K*, and we use the "poly" learning rate policy where current learning rate is the initial one multiplied by $(1 - \frac{iter}{max\_iter})^{power}$, and we set power to 0.9. Momentum and weight decay are set to 0.9 and .0005 respectively. Besides, the parameter ϵ in Eq. (6.3) for label smoothing regularization is set to 0.0001 and 0.1 for *Cityscapes* and *ADE20K*.

6.3.2 Results on Cityscapes

Training with synthetic data. Table 6.1 compares models using different sources of synthetic data. During training of listed models, each batch contains 70% real images and 30% synthetic data.

First, we train a segmentation model (Zhao *et al.*, 2017), with using synthetic images from previous state-of-the-art generation approaches (Zhu *et al.*, 2017b; Wang *et al.*, 2018b; Qi *et al.*, 2018). For Zhu *et al.* (2017b), we can see the performance is significantly decreased comparing to baseline model (Zhao *et al.*, 2017). For Wang *et al.* (2018b), we utilize the training set masks to generate images, which reduces performance across all four metrics comparing the baseline model as well. In addition, we test another state-of-the-art image generator (Qi *et al.*, 2018). To know if it is possible to improve semantic segmentation, we even utilize validation set masks. Despite of providing additional layouts from validation, the performance still decreases at validation set. Both experiments demonstrate that generated images often do not lead to improved performance.

In contrast, applying our synthetic features successfully improves results by

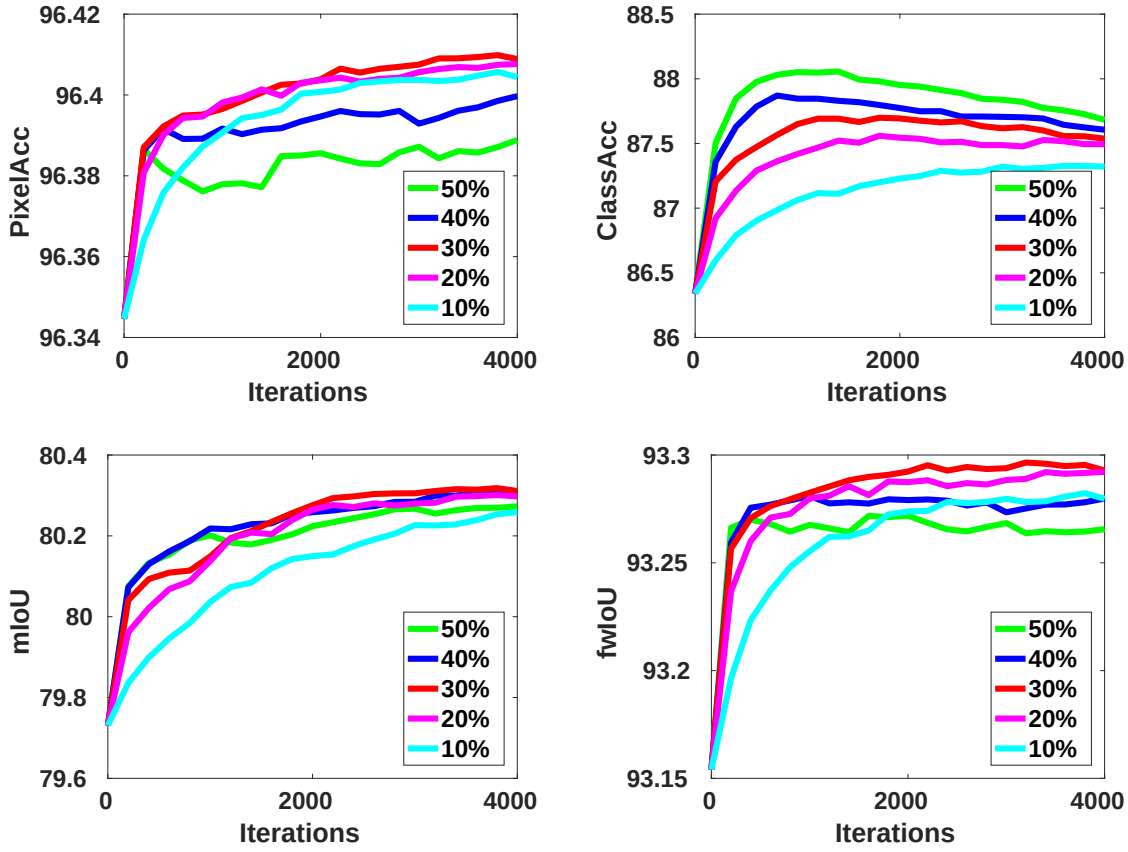


Figure 6.3: Evaluation of ClassAcc and mIoU at varying training iterations w.r.t. different percentages of our synthetic features on *Cityscapes*.

Table 6.2: Comparison of utilizing different model architectures.

Evaluation	Ours	PixelAcc	ClassAcc	mIoU	fwIoU
PSPNet-score		93.82	66.85	63.40	88.63
PSPNet-score	✓	96.44	80.05	74.33	93.44
Improvement		+0	+1.44	-0.52	+0.04
Improvement	✓	+0.06	+1.17	+0.55	+0.13

applying different semantic masks. Particularly, our feature synthesis has the same learning objective to image synthesis (Zhu *et al.*, 2017b), however, the performance is able to be improved clearly with training set only as well as additional validation set. The results demonstrate the effectiveness of utilizing our synthetic features for data augmentation in semantic segmentation, while synthetic images are hard to use.

In Figure 6.3, we provide a study on using different percentages of synthetic

Table 6.3: Statistics on different stages of features. The PSPNet-score for the real is 86.15.

	Images	*conv1_3*	*conv2_3*	*conv3_4*	*conv4_6*	*conv4_12*	*conv4_18*	*conv4_23*
Channels	3	128	256	512	1024	1024	1024	1024
Resolution	1	1/2	1/4	1/8	1/8	1/8	1/8	1/8
Entropy	4.8290	3.0987	3.1584	4.0767	3.3799	3.3659	3.2535	3.4065
mIoU of hist.	0.5596	0.5257	0.3171	0.3431	0.4295	0.4087	0.3651	0.2802
PSPNet-score	2.22	9.76	26.97	56.77	70.66	74.33	61.54	83.78
ΔClassAcc	-3.80	+0.02	+0.65	+0.81	+0.99	+1.17	+1.75	+0.08
ΔmIoU	-3.18	-1.09	-0.37	-0.1	+0.11	+0.55	+0.13	+0.05

features in a batch. First, it clearly shows that incorporating our synthetic features with different percentages brings improvements. Besides, we observe that incorporating more synthetic features will gain more in ClassAcc, which further shows the effectiveness of our approach. On the other hand, better fitting on synthetic data might lead to less improvement on other metrics, as a result, we mix 70% real images and 30% synthetic features in each batch for the rest of the experiments.

Importance of generator architectures. To explore the reason of improved results, we analyze the features from different network architectures: (1) baseline architectures in Zhu *et al.* (2017b), whose output size is same to input; (2) our architectures described in section 6.2.1.

To begin with, we provide a comparison of PSPNet-scores for those two architectures. We sample 3000 patches from *Cityscapes* training set to compute PixelAcc, ClassAcc, mIoU and fwIoU, as PSPNer-scores. Second, we train a segmentation model with synthetic features from different architectures. A good generator is suppose to have more close number to real features, and achieves larger improvements. Table 6.2 lists those numbers, showing features from our architecture lead to higher PSPNet-scores in four metrics. For improvements, even baseline has larger ClassAcc, the improvement for PixelAcc, fwIoU is less, and mIoU is quickly decreased. While, using features from our architectures achieves consistent improvements. As a result, we remarks that the effectiveness of our proposed architecture is a key to synthesizing successful features.

To understand more about synthetic features from difference architectures, we visualize them as well as the extracted features from real images in Figure 6.4. Even though both architectures output activations for different channels at similar locations as real features, our architecture with ASPP is able to produce better details. For example, activations in 49-th channel are synthesized for the pole class, which is very small, while the baseline cannot do. In addition, we feed the synthetic features

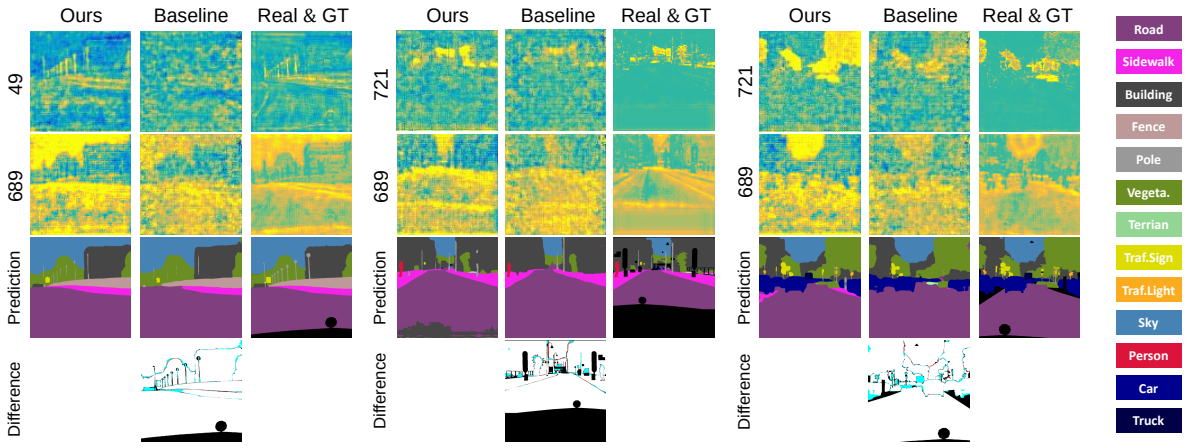


Figure 6.4: Visualization of synthetic features conv4_12. In difference maps, cyan color indicates our architecture is better, while red color means baseline is better. Best viewed in color.

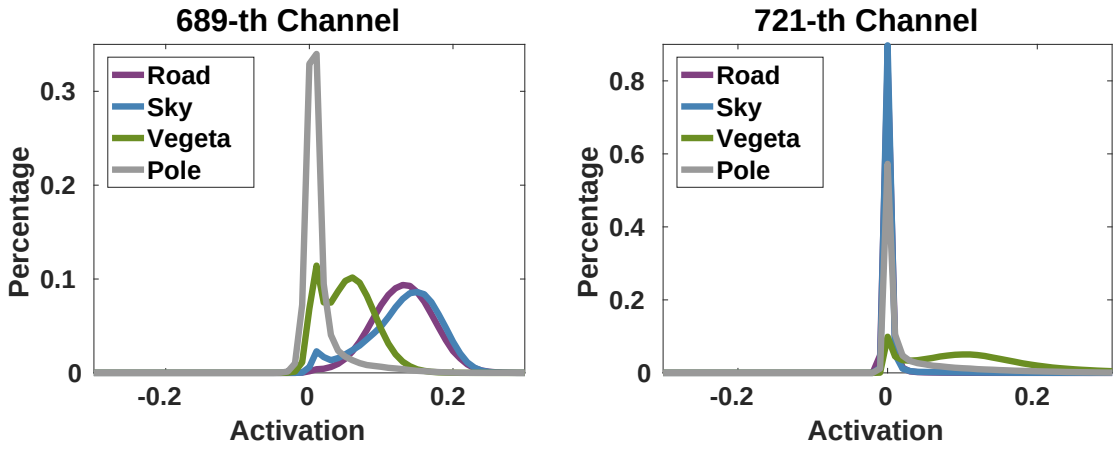


Figure 6.5: Distribution of selected channels for real conv4_12 of PSPNet on *Cityscapes*.

to PSPNet and compare the prediction difference. Observing all the difference maps, we further conclude that our model with high resolution input is stronger in generating features with accurate boundaries and details.

Synthesizing different stages of CNN features. In this paragraph, we perform synthesizing different stages of CNN features. The earliest stage is the image itself, which has shown to be hard to use for data augmentation in Table 6.1. To understand the synthesis of good features for semantic segmentation, we extract features (before ReLU layers) at different stages and provide statistics for them from PSPNet, as listed in Table 6.3. Due to the architecture of PSPNet, the latest stage of features we

extract is conv4_23, which is the input of the pyramid spatial pooling module.

The statistics are collected from 3000 randomly cropped image patches. First, we compute the entropy per class and channel, and report the average number in the table. To eliminate the distribution inconsistency for different stages, we normalize all patches to a same norm ball (L_2 norm 100) for computing the entropy. A good feature which is easier to generate is suppose to have smaller entropy. As we can see, the statistic is consistent to our requirement, that high-level features usually has smaller entropy.

Second, we compute the similarity of activation distributions across classes and channels. We use intersection of union between two histograms to measure the similarity. As shown in the 4-th rows in Table 6.3, similarities for higher level features are less , revealing stronger discrimination and ease to generate. Particularly, color pixels have the largest similarity, which further confirms our motivation and is consistent to difficulty of synthesizing qualified images to augment semantic segmentation, as shown in Table 6.1.

Third, we directly feed the synthetic features to PSPNet to test if they can be recognized by PSPNet. Good synthetic features should be recognized well. We use mIou of PSPNet as a measurement, as reported in the 5-th row of the table. As shown, scores are gradually increased from low-level features to high-level features.

Fourth, we applied different stages of synthetic features as training set, and report the improvements on ClassAcc and mIou on *Cityscapes* validation set. We observe that improvements on two metrics are obtained by using high level features conv4_6 to conv4_23. On the contrary, it is hard to achieve clear improvements with low-level synthetic features conv1_3 to conv3_4. Applying synthetic images leads to the worst performance. We also observe that the largest improvement comes from conv4_12 instead of conv4_23. Even conv4_23 has the highest PSPNet-score, it can boost less layers of a segmentation model, comparing to conv4_12. Finally, we choose to apply synthetic conv4_12 for *Cityscapes* in the rest of this section.

Last, we plot the histograms for pole, vegetation, sky and road at 689- and 721-th channels in Figure 6.5. The distribution differences for various classes and channels are large, which means they are discriminative. Besides, the features are very informative. The histogram indicates 90% pixels of vegetation have an activation at 721-th channel. 689-th channel is correspondent to road as well as sky. For our synthetic features, they produce activations at road and sky regions in 689-th channel at Figure 6.4.

Effectiveness of online hard negative mining. We compare the models with random sampled masks and mined hard negative during training in Figure 6.6. From this plot, we point out (1) simply adding synthetic features is already helpful to improve the performance; (2) Generating features for hard negatives can help us leverage the generator more effectively and obtains higher boosted performance.

Additional semantic layout masks. Since our approach generates a paired data from a semantic mask, we are not only able to augment a training set, but also to leverage new masks to generate more novel examples. To know if more masks are

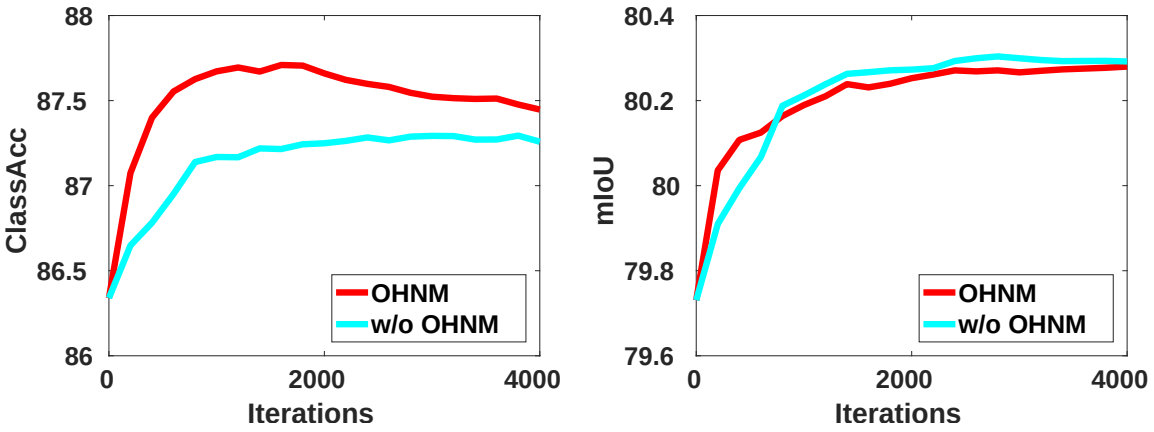


Figure 6.6: ClassAcc and mIoU on *Cityscapes* w.r.t. applying online hard negative mining (OHNM). Red curve is the model with OHNM.

beneficial to boost the performance, we seek for additional masks as listed in the following

- Validation GT masks. It is very easy to acquire and quality of masks is very high. It is used to test if providing more masks with the same distribution is helpful.
- Rendering system. It provides us large amounts of data with very low cost. In our experiments, we apply recent released *Synscapes* dataset (Wrenninge and Unger, 2018). Finally, it provides us 25000 extra semantic masks.
- Pseudo GT masks. We regard prediction from unlabeled images as pseudo ground truth. Although the prediction is not perfect, we generate paired data from a mask, as a result, it still generates features aligned with the input mask. To alleviate the unsmooth prediction, we only save the prediction with posterior larger than 0.7, and impaint the holes with nearest neighbor interpolation. To provide novel scenes, we leverage the unlabeled video frames and coarse annotation frames in *Cityscapes*, leading to 29823 extra masks.

Table 6.4 presents the performance with single scale predicton and multi-scale prediction. To have a fair comparison, we set the ratio between training masks and additional masks for all the additional mask choices. That is $3:1$, in each sampled batch for synthesizing. Besides, the ratio between real images and synthetic data follows previous experiments $7:3$, and online hard negative mining is applied for various versions of our models.

We observe that after adding validation mask, the performance is further improved than pure augmentation (second row). Besides, applying pseudo GT also achieves better performance than training set only. Interestingly, when we apply the semantic masks from *Synscapes* (Wrenninge and Unger, 2018), the improvement is

Table 6.4: Comparison of PSPNet and our approach on Citycapes validation set. The top block is single scale prediction and the bottom is multi-scale prediction.

Models	Additional Mask	PixelAcc	ClassAcc	mIoU	fwIoU
Baseline	–	96.34	86.34	79.73	93.15
Ours	No	96.40	87.29	80.30	93.27
Ours	Val	96.40	87.47	80.33	93.29
Ours	Synscapes	96.39	87.55	80.03	93.27
Ours	Pseudo GT	96.40	87.49	80.31	93.28
Baseline	–	96.59	87.23	80.89	93.58
Ours	No	96.64	88.16	81.48	93.69
Ours	Val	96.64	88.46	81.52	93.71
Ours	Synscapes	96.64	88.43	81.34	93.70
Ours	Pseudo GT	96.64	88.34	81.51	93.70

less, the reason is the distribution gap of layouts between *Cityscapes* and *Synscapes*. As a result, to leverage synthetic data from game engine better, not only similar appearance, but also similar semantic layouts are needed, such as shapes, ratios between different objects, etc.

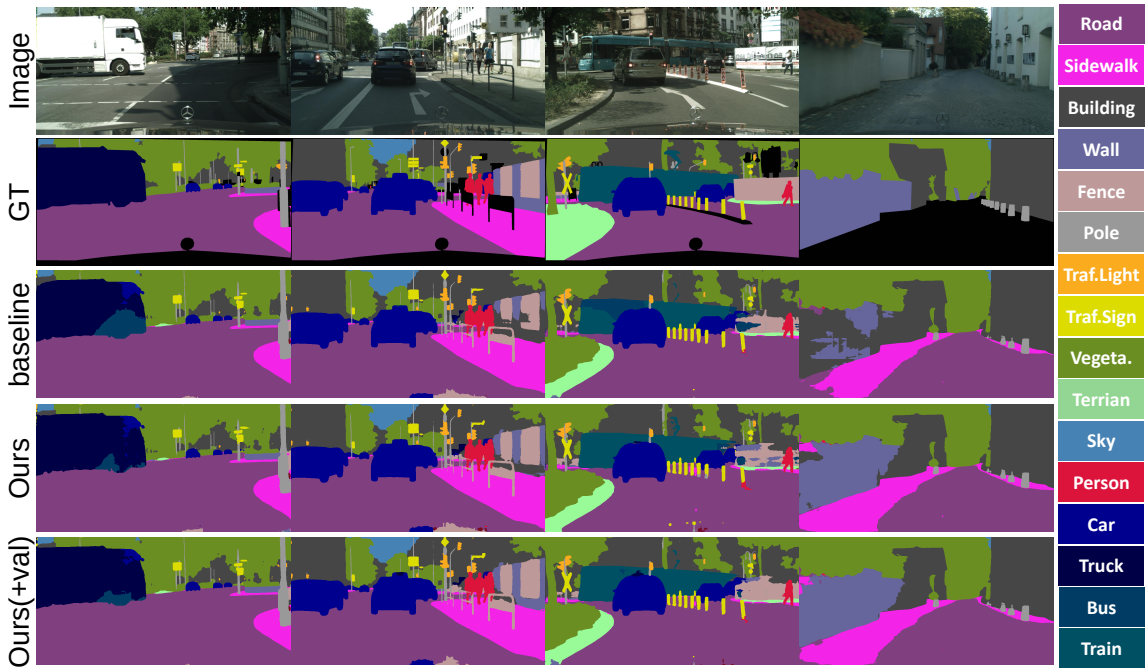


Figure 6.7: Qualitative results on the Cityscapes validation set. We show the augmentation results as well as using additional masks from validation set. Best viewed in color.

Qualitative comparison. We present some visualization results comparing to our baseline PSPNet (Zhao *et al.*, 2017) in Figure 6.7. First, our model predicts more smooth results. In the second examples, our model successfully recognize the fence and wall along the street, and accurately segment the boundary between them. Besides, for some large regions, i.e., the truck, the train and the wall in the first, third and fourth examples, our approach achieves clear improvement and segment the whole objects, while baseline predicts unsmooth results. Particularly, we notice that our model with training masks only is also able to achieve an clear improvement over baseline and very similar results to our model with additional validation masks, which means our data augmentation is very effective.

6.3.3 Results on ADE20K

To understand if our approach works well, we also test our approach on *ADE20K* dataset. We present a quantitative comparison in Table 6.5 with single scale prediction as well as multi-scale prediction. We emphasize that our model achieves 2.08/2.43 and 0.34/0.57 improvements on ClassAcc and mIou for single scale prediction and multiple scale prediction, respectively. Besides, we show several qualitative comparisons in Figure 6.8, that our predictions are more smooth and accurate. We are able to distinguish ambiguous classes, such as house/building, mountain/rock. As a result, our model recognize the entire region for the objects, instead of generating multiple cracked regions.

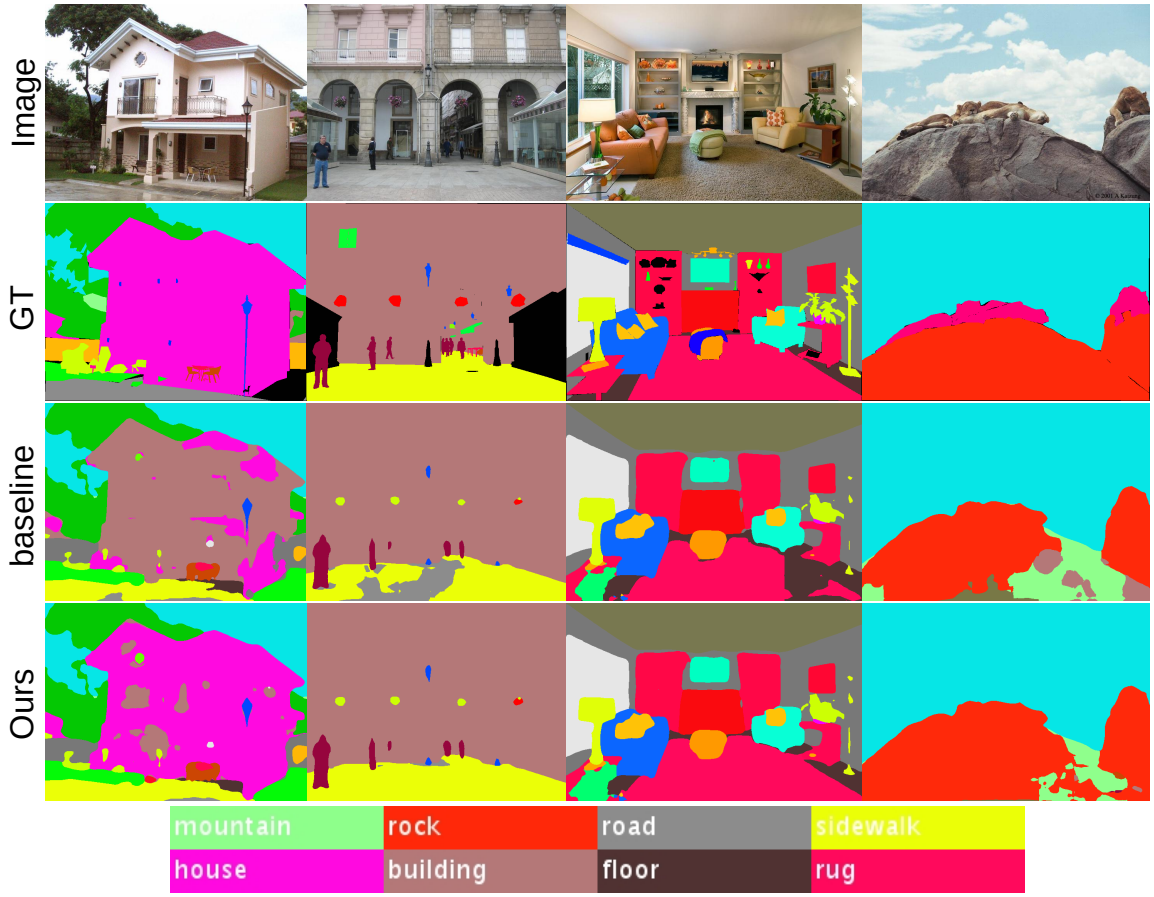


Figure 6.8: Qualitative results on the *ADE20K* validation set. It is clearly observed our data augmentation approach is more robust and produce smoother segmentation results.

Table 6.5: Comparison results on *ADE20K* validation set. The top block shows the results of single scale prediction and the bottom is multi-scale prediction.

Models	PixelAcc	ClassAcc	mIoU	fwIoU
Baseline	80.04	51.75	41.68	67.46
Ours	80.00	53.83	42.02	68.19
Baseline	80.76	52.27	42.78	68.75
Ours	80.81	54.70	43.35	69.17

6.4 DISCUSSION

Our method provides a conditional generative adversarial network for learning CNN intermediate features from semantic layout masks. Similar to generating images, we also aim to model the data distribution correctly for CNN features. The experimental results show that our synthetic features can be used as data augmentation strategy during training, to improve the semantic segmentation performance. Besides, our results also show our generator is able to take a novel layout mask, and produces a corresponding feature map. With additional mask, our method further improve the performance of a segmentation model.

Besides, our dense feature generator paves the way for further applications presented in Chapter 7. Unlike generating images, that people are interested in synthesizing visually realistic images, synthetic features provide potential usage for stronger machine learning models. Chapter 7 leverages our synthetic features in inference, which aims for data privacy purposes. To summarize, we present a very early study on CNN feature generation, while most people try to synthesize visually appealing images. Except creating visually realistic images, we show generative modelling can be further explored for different goals, with respective to better leverage of data.

6.5 CONCLUSION

In this work, we propose synthesizing dense features for improving semantic segmentation. Our pipeline is simple but surprisingly effective to reach stronger segmentation results. A powerful architecture for dense feature synthesis is presented, enabling us to synthesize realistic features with rich details. Also, several techniques are presented to leverage synthetic features more effectively.

7 MEMBERSHIP INFERENCE ATTACKS ON BLACK-BOX SEMANTIC SEGMENTATION MODELS

Today's success of state of the art methods for semantic segmentation is to a large extend driven by large datasets. Collection and annotation of such datasets comes at significant efforts and associated cost, therefore data is considered an important assets that need to be protected. In addition, visual data might contain private or sensitive data, that makes it equally unsuited for public release. Unfortunately, recent work on membership inference in the broader area of adversarial machine learning and inference attacks on machine learning models has shown that even black box classifiers leak information on the dataset that they were trained on. We provide the first investigation if complex, state of the art models for semantic segmentation are equally effected by such attack vectors. We provide the first method that is able to determine if a particular image was in the training set or not. In order to mitigate the associated risks, we also provide the first defenses against such new attack vectors, by leveraging prior work on classification, but also proposing a completely novel paradigm based on features synthesis that offer better protection. Finally, we extensively evaluate our attacks and defenses in a range of highly relevant real-world datasets *Cityscapes*, *BDD100K* and *Mapillary Vistas*.

7.1 INTRODUCTION

From the early breakthroughs in image classification fueled by the availability of large datasets (e.g. ImageNet), to the steady increase across different tasks that seems to correlated well with the increasingly large dataset, e.g. in semantic segmentation (Cordts *et al.*, 2016; Neuhold *et al.*, 2017; Yu *et al.*, 2018), VQA (Antol *et al.*, 2015), data is playing a key role in today's state of the art computer vision methods. Even on a slightly longer time horizon, today's methodology seems to scale well with the availability of data even for very large datasets (Sun *et al.*, 2017).

Hence, researcher and industry alike have recognized the importance of large and high quality datasets in order to push the state of the art in computer vision. However, data collection and in particular annotation and curation of large datasets comes at a substantial cost. While there are sizable efforts from the research community (Cordts *et al.*, 2016; Yu *et al.*, 2018; Geiger *et al.*, 2012), also industry has picked up the task of collection (Neuhold *et al.*, 2017) as well as providing annotation services such as Amazon MTurk platform, which in turn can be monetized and constitutes important assets to companies.

As a consequence, such assets need protection e.g. as part of intellectual property and it should be controlled which parts are made public (e.g. for research purposes)

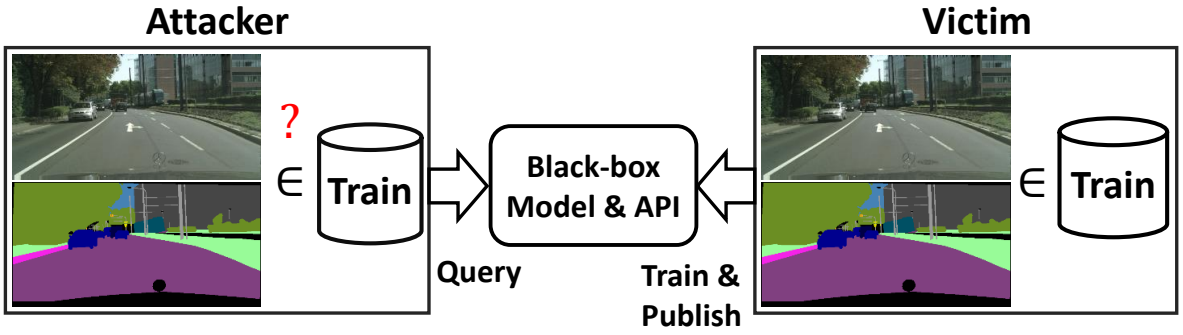


Figure 7.1: Membership inference attacks for black-box semantic segmentation models. The membership of training data is able to be attacked according to observing returned predictions.

and which part remain private. Based on these datasets, high performing models for semantic segmentation are trained which then are made public (e.g. as black box models) via an API or as part of a product. One might assume that the information of the training set remains contained within the trained parameters of the model and therefore remains private.

Beyond the aspect of intellectual property, data might also include private information of people that were captured as part of the data collection process. Either the legal basis for the capturing process can assume that information on the images is not leaked or consent of the depicted individuals not always includes dissemination of visual information beyond the purpose of training a machine learning model.

Unfortunately, recent work on inference attacks (Shokri *et al.*, 2017; Salem *et al.*, 2019) has shown that even a black box machine learning model leaks information of the training data. Current membership inference attacks try to infer if a particular sample was used as part of the training or not. Such approaches have shown high success rates on a range of classification tasks and have equally proven to be hard to fully prevent. While this constitutes a potential threat to the machine learning model, it can also potentially be used as a forensics technique to detect a potentially unauthorized used of data.

Currently, we are missing even a basic understanding to what extend these attack vectors of membership inference extend to state of the art models of semantic segmentation. Hence, we conduct the first investigation of membership inference attacks for semantic segmentation as well as propose different defense mechanisms. Specifically, we attack a semantic segmentation model for street scene understanding with two adversaries, that cover attack settings with constraints on the model and data selection, as well as unknown target model case. On the other hand, we study defenses of membership privacy to deal with information leakage in semantic segmentation. In addition, we propose a novel and effective defense mechanism based on generative modeling on the intermediate feature representation of a semantic segmentation neural network. A feature generator is learned from

segmentation trained with a dataset to be protected, which is utilized to perform prediction obfuscation for returning segmentation results to users. We show our proposed synthetic feature based defense is significantly effective to reduce the statistical distribution gap between membership data and others during test, and thus make the classification of membership difficult.

We highlight three aspects of our work in the following:

- We present the first work on membership inference attacks against semantic segmentation models.
- We present defenses against membership inference for semantic segmentation, and propose a novel feature synthesizing based approach, which significantly mitigate leakage of information on training data.
- A set of ablation studies and extended analysis is provided in order to shed light on the core challenges of membership inference attacks in the context of models for semantic segmentation.

7.2 MEMBERSHIP INFERENCE ATTACKS AGAINST SEGMENTATION MODELS

Membership inference attacks against machine learning models attempt to infer if a data pair (x, y) belongs to the training set of a given black-box segmentation model or not, as shown in Figure 7.2. Intuitively, membership inference attacks exploit that machine learning models tend to fit better on the training data and tend to produces more confident responses than on unseen examples. Based on this idea, we are proposing a membership inference attack on semantic segmentation models.

We now describe our attack and different roles of membership inference attacks on black-box semantic segmentation models in more detail. In order to represent different knowledge the attacker might have, we setup two adversaries to attack a segmentation model for which we give details later. In each adversary, there is a shadow model and an attack model, which is explained in the following.

Attack pipeline. Modern semantic segmentation models are normally trained with image crops from a larger one, to handle varying image resolutions and huge memory cost. As a result, our attacks reverse this for patch-level attack and image-level attack. Even though we are interested in knowing the membership of an entire image, the attacking system is built on patch-level prediction, which indicates the probability of use of an image patch for training. Image-level attacks are achieved by predicting each patch via sliding windows or random locations. And we use an average operation to fuse them all.

Victim V. This model is trained on a labeled dataset $D_V = \{(x, y)_i\}$, and provides posteriors of a prediction from a query image x as $\hat{y} = \mathbf{V}(x)$. It is deployed with sophisticated designs for providing high quality services, but a black-box model

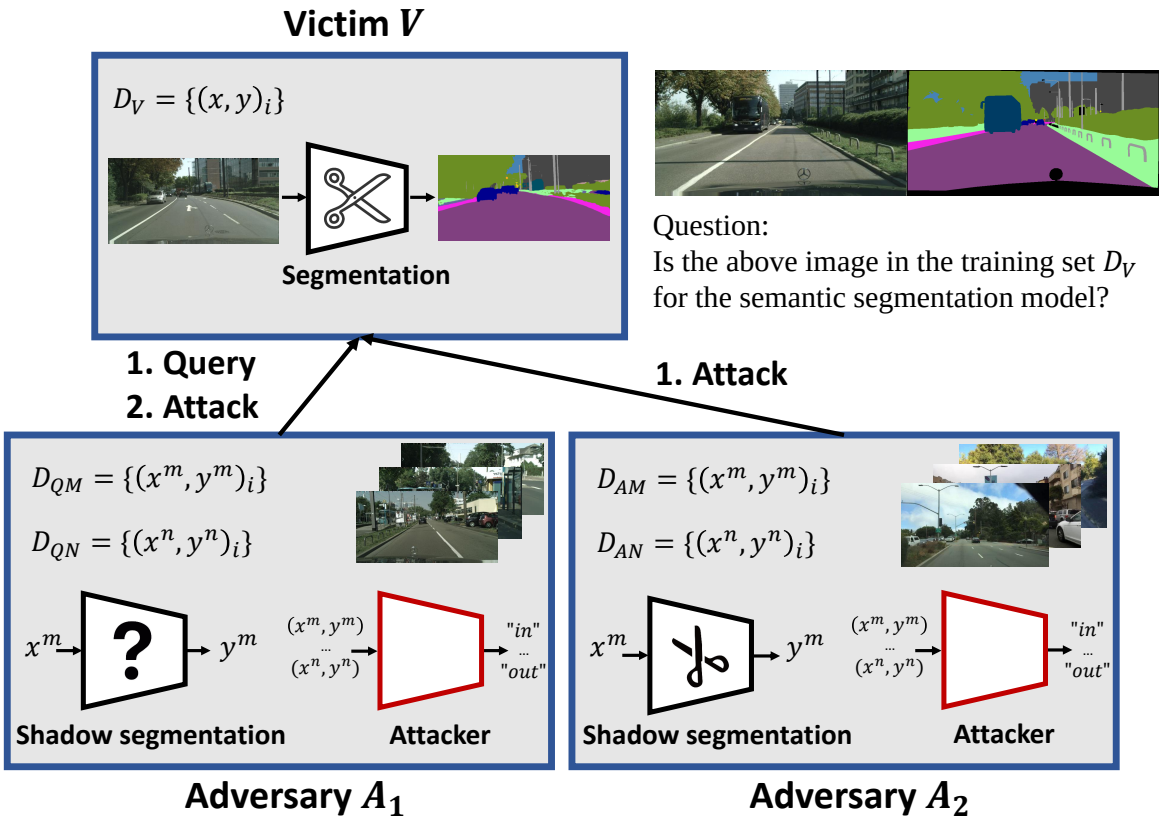


Figure 7.2: Problem statement. We consider two types of adversaries: model and data dependent attacks ($\mathcal{A}_1$), and independent attacks ($\mathcal{A}_2$), allowing for attacks under various conditions in real world.

for users. In other words, reported posteriors $\mathbf{V}(x)$ are the only information that attackers can acquire by interacting with the model.

Shadow model S. It is a semantic segmentation model with the same label space as a specific victim. It is trained by the attacker and thus provides the membership of images and their prediction posteriors of a semantic segmentation model for training an attacker. It produces prediction posteriors from an input image x as $\hat{y} = \mathbf{S}(x)$.

Attack model A. It is a binary classifier, whose input is a pair of prediction posteriors of an image and its corresponding ground truth labels. It is trained with paired data $(\mathbf{S}(x), y)$ and $\mathbf{L}$, where $\mathbf{L}$ refers to "member" or "nonmember" class of the pair, and objective is $\mathbf{L} = \mathbf{A}(\mathbf{S}(x), y)$. Once it is trained, membership inference attacks can be performed on victim $\mathbf{V}$ of (x, y) with $\mathbf{A}(\mathbf{V}(x), y)$.

Adversary $\mathcal{A}_1$: Model and data dependent attacks. . This attack assumes that the victims model can be queried at training time of the attack and that the model architecture of the victim is known. This allows the attacker to train a shadow model

with the same architecture to the victim, i.e. $\mathbf{S} = \mathbf{V}$.In this case, the first step is to construct a training set $D_{QM} = \{(x^m, y^m)_i\}$ for training $\mathbf{S}$, in order to mimic the behavior of the victim's model $\mathbf{V}$. Based on the assumption that models is slightly overconfident on the training data, we select the images with stronger confidences among all query images as the training set D_{QM}, while others are D_{QN}, which provides nonmember data of $\mathbf{S}$ for training $\mathbf{A}$.

Adversary $\mathcal{A}_2$: Model and data independent attacks.. For this adversary, we only know the victim model's functionality and a defined label space. There is no query process for constructing training set for $\mathbf{S}$, instead, $\mathbf{S}$ is able to be trained with a dataset on complete different domain as shown in Figure 7.2, which is a cheaper and more practical attack. The goal of the shadow model is to capture the membership status for each example, and provide training data for attack model $\mathbf{A}$. Except data distribution, there is no additional knowledge provided, i.e. $\mathbf{S} \neq \mathbf{V}$. Particularly, we highlight the severity of information leakage in this simplified attack. Model and data distribution are completely different to victims, even there is no query process, which might be detected on the server.

7.3 DENFENSES

Segmentation models suffer from attacks due to overfitting on a training set w.r.t. appearances and layouts. The success of attacks comes from the fact that models predict training data with higher confidences. To deal with this phenomenon, we develop a defense mechanism for protecting membership information of a semantic segmentation modelWe will investigate solutions that are inspired from defenses against membership inference on classifiers as well as present out novel solution that learns a feature generator to synthesize visual features as another source of predictions to build a safer black-box model via prediction obfuscations, as shown in Figure 7.3.

It illustrates the components and workflow of our membership privacy defense for black box models in semantic segmentation. When we have a dataset $\{(x, y)\}$, a neural network segmentation model can be trained with an encoder-decoder architecture, and predict a layout mask as $\hat{y} = \mathbf{D}(\mathbf{E}(x))$. Instead of directly reporting the posterior of an input image, we obfuscate the original prediction by a synthetic feature from a generator $\mathbf{G}$, which takes a conditional layout mask y as the input. In this way, statistical distribution on the training data bias to synthetic data, and the appearance distribution of novel examples are also from the feature generator, as shown in Figure 7.4. Even though there is also distribution gap of layouts in semantic segmentation, the information leakage is still effectively alleviated.

Hence, we construct a model to serve users segmentation in the following way: (1) collect training data and train a semantic segmentation model; (2) extract visual features and their corresponding layouts, and train a feature generator; (3) perform prediction obfuscations for query images. Notably, our approach is simple since we do not change objective functions of training individual modules. Taking advantages

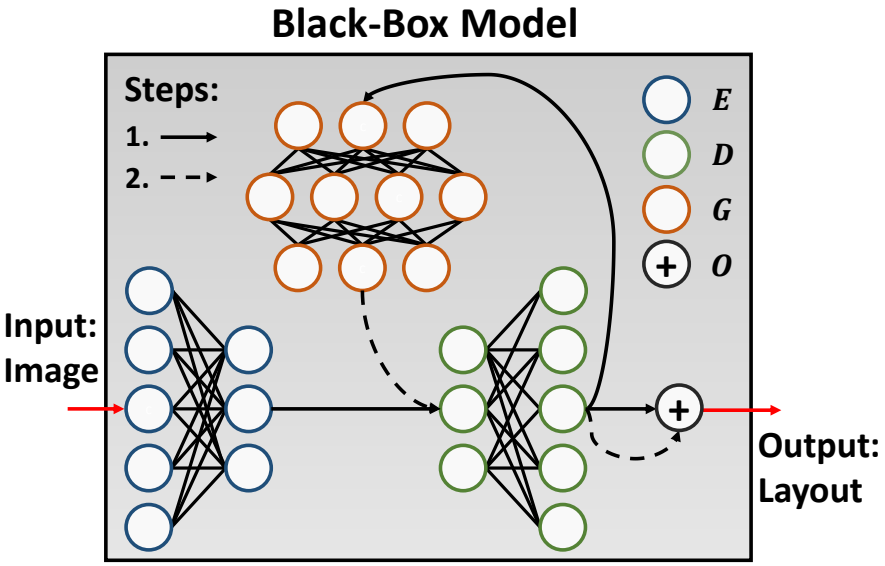


Figure 7.3: Components of a black-box model with defense. It is comprised of the encoder (E) and the decoder (D) of a segmentation model, a feature generator (G) and an obfuscation operation (O). Final protected predictions are obtained with combining two steps.

of generative adversarial training (Goodfellow *et al.*, 2014), we are able to learn a generator for synthesizing high quality and realistic visual features. We regard the feature generation as an image translation problem (Isola *et al.*, 2017; Zhu *et al.*, 2017b), and apply BicycleGAN (Zhu *et al.*, 2017b) to learn the generator, which is able to model multimodal distributions. The goal of feature generator $\mathbf{G}(y)$ aims to match the distribution of $\mathbf{E}(x)$, formally $\mathbf{E}(x) \sim \mathbf{G}(y)$. As a result, synthetic features are also able to produce predictions according to forward of the decoder, i.e., $\mathbf{D}(\mathbf{G}(y))$.

Specifically, our prediction obfuscation strategy has two steps: (1) direct prediction from an input image $\mathbf{D}(\mathbf{E}(x))$; (2) obfuscation with a synthetic feature. We utilize the prediction layout from first step as condition to generate a feature, which is used to obfuscate the first prediction, as shown in Figure 7.3. Therefore, the synthetic feature provides another prediction $\mathbf{G}(\mathbf{D}(\mathbf{E}(x)))$, and our obfuscation is achieved by combining $\mathbf{D}(\mathbf{E}(x))$ and $\mathbf{G}(\mathbf{D}(\mathbf{E}(x)))$. Particularly, to provide more information from synthetic features as much as possible, we set a mask, assigning the pixels with different predicted labels from two sources. We take the posteriors outside the mask from synthetic features completely, and the linear combinations inside the mask. Eventually, the posterior $\hat{y}$ from an input image x is formulated as

$$\begin{aligned}\hat{y} =& M \odot ((1-d) \times \mathbf{D}(\mathbf{E}(x)) + d \times \mathbf{D}(\mathbf{G}(\mathbf{D}(\mathbf{E}(x))))) \\ &+ (1-M) \odot (\mathbf{D}(\mathbf{G}(\mathbf{D}(\mathbf{E}(x))))),\end{aligned} \tag{7.1}$$

where M is a mask indicating the locations where posteriors from synthetic and real

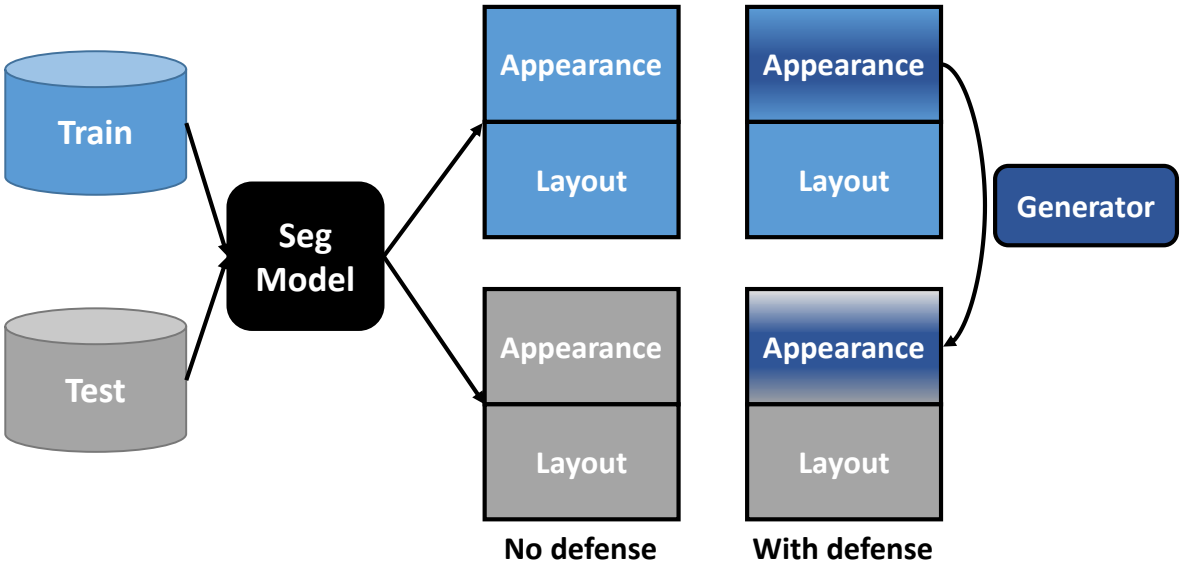


Figure 7.4: Prediction Obfuscation with feature generation. Appearances for both training and testing data bias to the generator, which is learned from training data.

data refer to different labels, and d is a defense factor in the range of $[0, 1]$ to control the importance of security considerations.

7.4 EXPERIMENTS

We conduct the experiments on street scene semantic segmentation task between various datasets, including *Cityscapes* (Cordts *et al.*, 2016), *BDD100K* (Yu *et al.*, 2018) and *Mapillary Vistas* (Neuhold *et al.*, 2017), which are captured in different countries under diverse weather conditions and image qualities, providing different domains of street scenes. We attack victim models with two adversaries as described in section 7.2: (1) model and data dependent attacks $\mathcal{A}_1$; (2) model and data independent attacks $\mathcal{A}_2$, where we utilize different datasets and semantic segmentation models for the adversary comparing to a victim model. In all attacks, we train a ResNet-50 (He *et al.*, 2016) from scratch as our attack model $\mathbf{A}$ of section 7.2, which allows us to visualize the regions contributing to the recognition of membership for a test example by class activation mapping (Zhou *et al.*, 2016).

7.4.1 Attacks

7.4.1.1 *Model and data dependent attacks*

In this setting, we leverage ResNet-101 based PSPNet to mimic the black-box model of a service provider, and use *Cityscapes* to conduct the experiments. The PSPNet is

pretrained using the data with coarse annotation in *Cityscapes* before train a victim or shadow model. Hence, we utilize all 3475 images with fine annotation in *Cityscapes* (2975 from training set and 500 from validation set) and split them into two parts without intersection for the victim and our shadow model in adversary. For the victim model, we use 1448 images from original training set to learn a segmentation model, and use 952 images as nonmembership to evaluate attacks. Therefore, we use the left images for training a shadow model and an attack model. 555 images are selected to train the shadow model via interactions with the victim, and select the examples with highest confidence scores. Finally, left 520 images are regarded as nonmembership to train an attack model.

To have a better evaluation and understanding on membership inference attacks, we train attack models with the shadow models at 15000 and 30000 training iterations, because we assume it is hard to know the training iterations of the victim. We compute the precision and recall of attacks with varying threshold, and precision-recall curves are drawn in Figure 7.5. The patch-level attacks are conducted at the patch size of 713 and image-level attacks are done with 10 patches at random locations, that 6 patches can cover an whole image of *Cityscapes*. In addition, we also plot the ROC curves in Figure 7.6. From those figures, we clearly observe the information leakage, that we achieve 0.75 and 0.83 precision when 0.8 recall is provided, at 15000 and 30000 iterations, respectively.

7.4.1.2 *Model and data independent attacks*

In this setting, we assume the only knowledge of a victim is street scene segmentation with 19 classes. We train semantic segmentation models with different datasets and models, as summarized in Table 7.1, that we attack PSPNet on *Cityscapes* with Deeplab-v3+ and DPC. We highlight that *Cityscapes*, *BDD100K* and *Mapillary Vistas* have image resolution of 1024×2048, 720×1280 and varying sizes whose heights are from hundreds to thousands. Besides, *Mapillary Vistas* does not have the same label space to *Cityscapes*, which further increases the challenge of attacks. To handle this situation, we pick up 25 classes from 65 classes, and set others as ignored regions. Some conceptual similar classes are merged and a label space with 19 categories is created, which is compatible with *Cityscapes*. Our merged label space of *Mapillary Vistas* can be found in the Supplementary Materials.

In our experiments, we use official released ResNet-101 based PSPNet as our victim model, and we do not touch it when we train an adversary. In each adversary, we train the shadow model by ourselves. Specifically, we train Deeplab-v3+ with initial learning rate 0.01, batch size 7 and cropped patch size 705, and DPC with initial learning rate 0.02, batch size 9 and cropped patch size 721. Both models apply momentum 0.9, weight decay 0.00004, and “poly" learning rate adjust strategy with factor 0.9, which are widely used in semantic segmentation.

Surprisingly, we observe information leakage in both attacks from *BDD100K* and *Mapillary Vistas* in Figure 7.5, 7.6. Particularly, the adversary using *BDD100K* achieves only a bit lower performance in patch-level prediction than our first adversary with

Table 7.1: Experimental descriptions of model and data independent attacks. We regard training set or a part of them as membership data (M) and left as nonmembership data (¬M). For segmentation models, we employ either ResNet-101 (He *et al.*, 2016) or Xception-71 (Szegedy *et al.*, 2016) backbones.

Dataset	Model	Backbone	M / ¬M
Cityscapes	PSPNet (Zhao *et al.*, 2017)	ResNet-101	2975 / 500
BDD100K	Deeplab-v3+ (Chen *et al.*, 2018c)	Xception-71	4k / (3k+1k)
Mapillary Vistas	DPC (Chen *et al.*, 2018a)	Xception-71	10k / (8k+2k)

querying images whose shadow model are trained with 30000 iterations.

7.4.2 Defenses

To mitigate the risk of leaking information, we try a set of defense techniques for semantic segmentation models in the following:

Argmax. It only returns predicted labels of an image, instead of posteriors. It tries to provide less information to protect a victim model.

Dropout. It is used to avoid overfitting, which helps a network reaching higher performance. Besides, it also regularizes networks to emit less confident predictions for training data, which makes membership inference attacks harder. In our study, the victim model (PSPNet) is trained with one dropout layer of ratio 0.1. Besides, we enable dropout during test with ratio 0.1 and 0.9 to make predictions blurry and confuse attackers under different levels of strength.

Gaussian. We add Gaussian noises with different variances (0.5, 1.0) to the posteriors of predicted labels for each location, and then normalize them with sum of 1. Particular, to ensure posteriors have positive values, we set 0.0001 as truncation value to bound the value before normalization.

SynFeat. It is presented in section 7.3, and we learn a feature generator at conv4_12 of PSPNet for protecting victims, and apply different defense factors 0.75 and 0.95 in Eq. (7.1) for a comprehensive analysis.

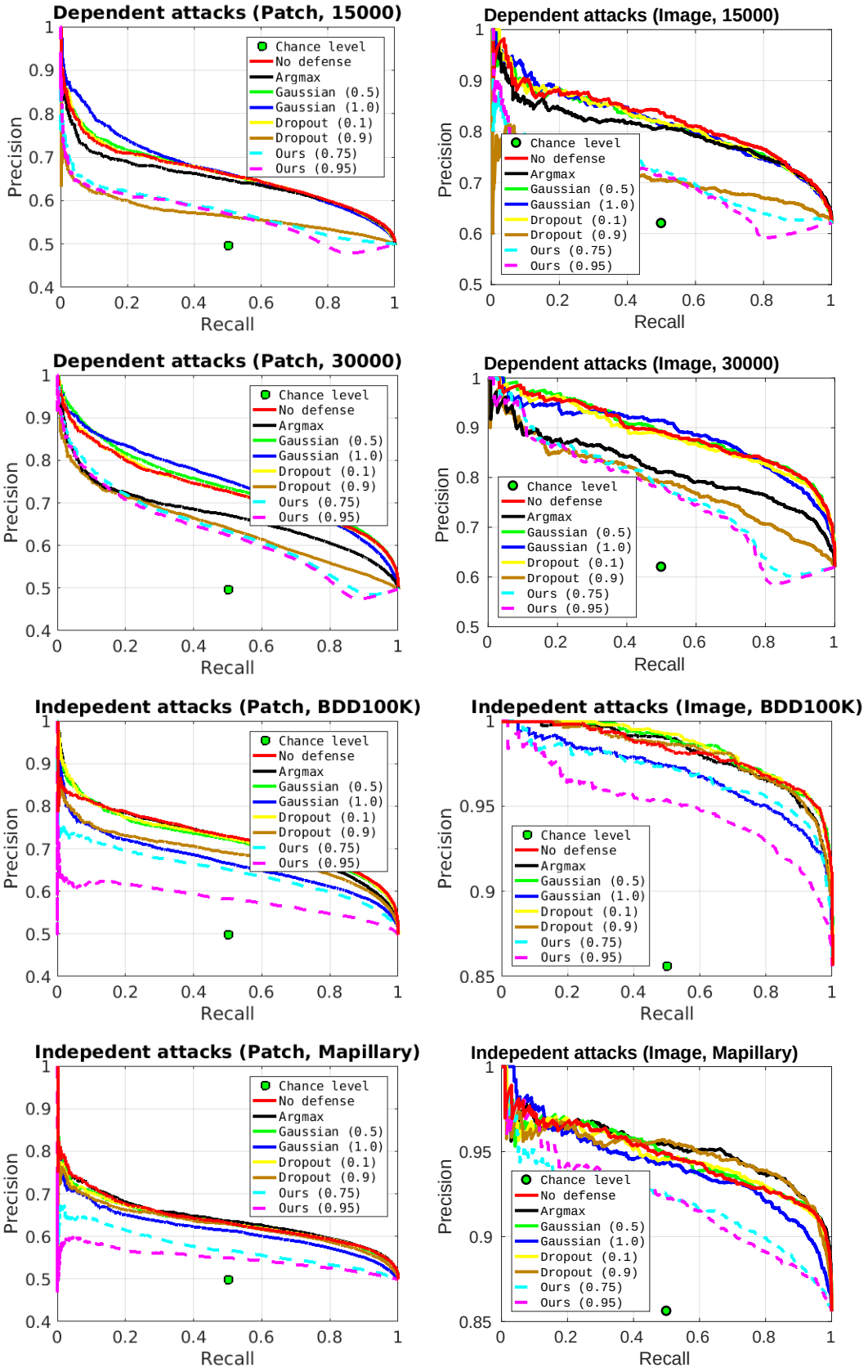


Figure 7.5: Precision-recall curves of attacks and defenses. Patch- and image-level attacks are drawn. Dependent and independent attacks are presented from up to down.

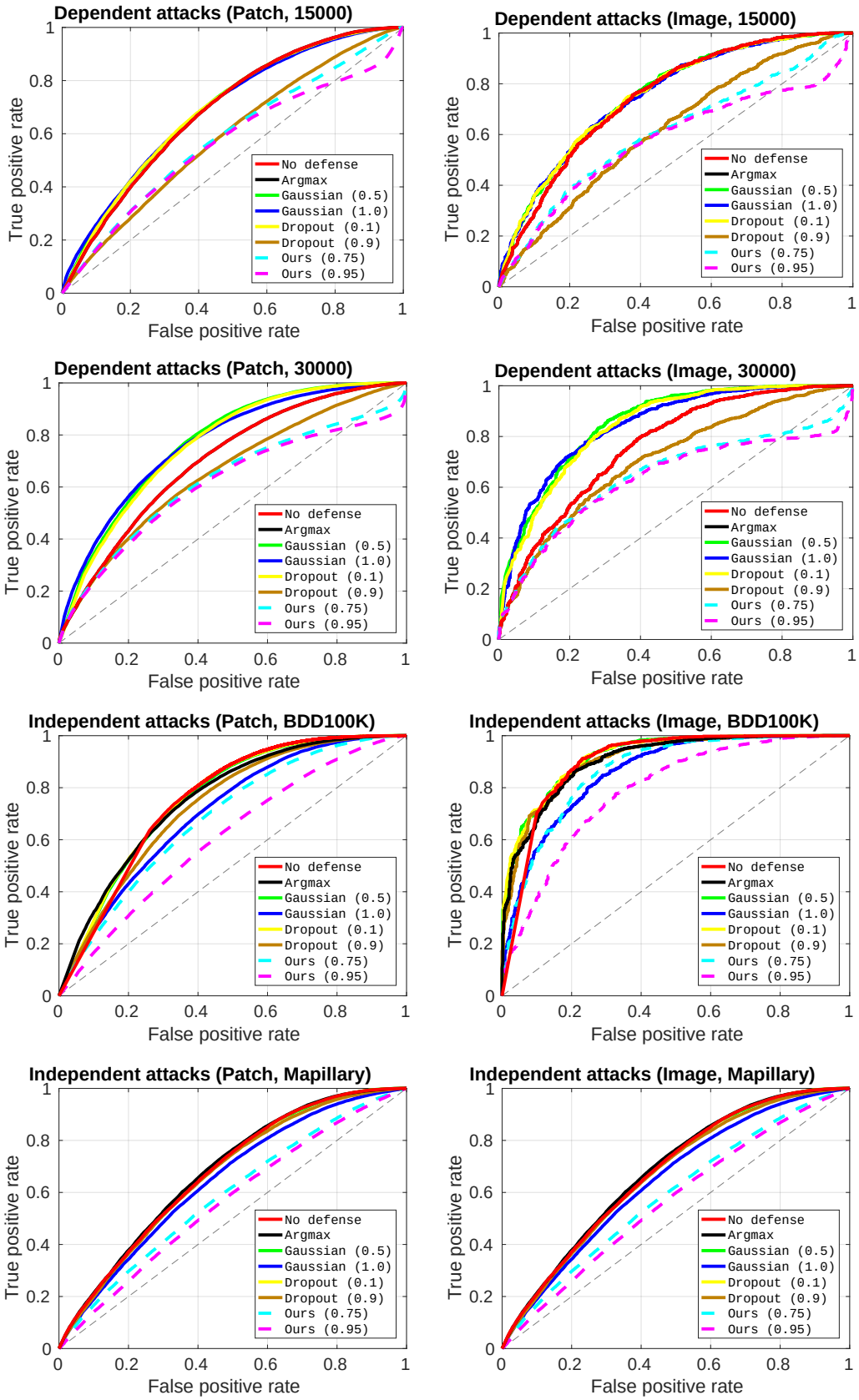


Figure 7.6: ROC curves of attacks and defenses. Patch- and image-level attacks are drawn. Dependent and independent attacks are presented from up to down.

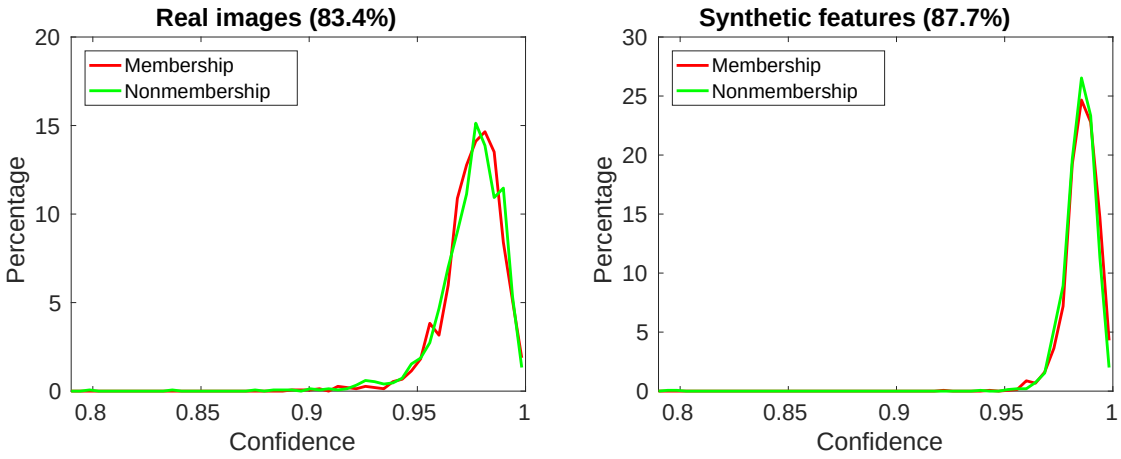


Figure 7.7: Comparisons of confidence scores (posteriors of predicted classes) from segmentation models.

7.4.2.1 *Results*

In figure 7.5 and 7.6, we present precision-recall curves and ROC curves for various defenses and compare them to the model without any defense. We first observe **Argmax** performs similar to the situation without any defense in independent attacks, but shows some promising defense results on the dependent setting, but not strong enough for handling all kinds of attacks. **Dropout** has similar behaviors to **Argmax**, that also performs well in dependent attacks but not clear defense results for the independent setting. In contrast to above two defenses, **Gaussian** protects victims in independent setting, but does not work well in dependent one. We conclude that adding Gaussian noises cannot eliminate or reduce the distribution gap between training and testing data, as a result, it is hard to achieve high quality security guarantee with querying images.

Finally, we can see that **SynFeat** achieves better performance than all the simple strategies for both settings. It is clearly shown that attacks have been alleviated and get close to chance level accuracy. To understand this results, we visualize the distributions of posteriors for predictions from real images and ones combined with our synthetic features at different defense factors in Figure 7.7. We also quantify the similarity between membership and nonmembership data, which is expected to higher scores for successful defenses. Obviously, with increasing defense factors, higher similarities are achieved. Besides, we also visualize the distribution of logits before softmax of the attack model trained with *BDD100K* for predicting as membership in Figure 7.8. It is apparently shown the posteriors from our models are more challenging to distinguish that the attack's output logits are around 0 for both membership and nonmembership data.

We also provide interpretations on the attacks with class activation mapping (CAM) maps (Zhou *et al.*, 2016), as shown in Figure 7.9. It highlights the regions for

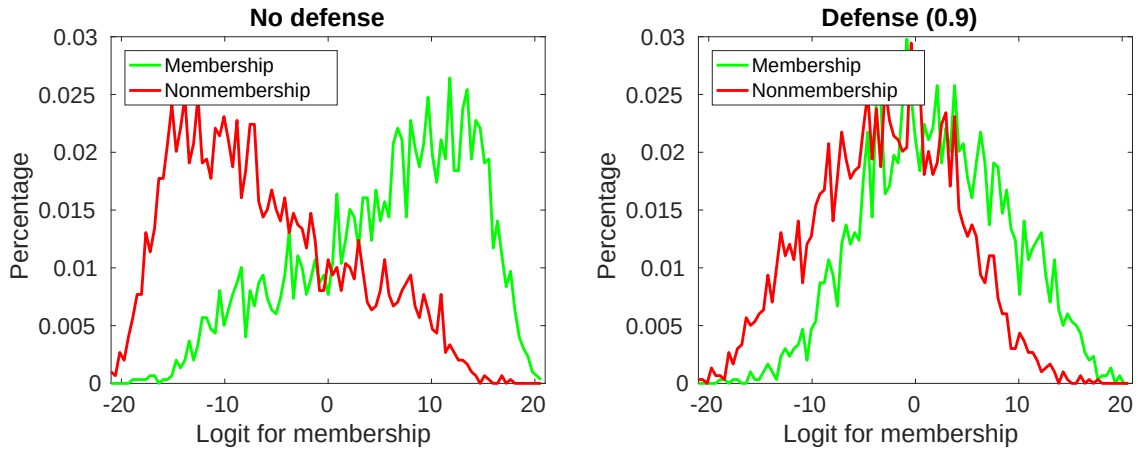


Figure 7.8: Comparisons of logits from attackers.

recognizing query images are in the training set of *Cityscapes*, where they actually are. We can see **SynFeat**'s highlight regions are drastically changed comparing to the model without defense, while **Dropout** has similar ones. Besides, **Gaussian** also changes CAM maps, and these observations are compatible with precison-recall and ROC curves.

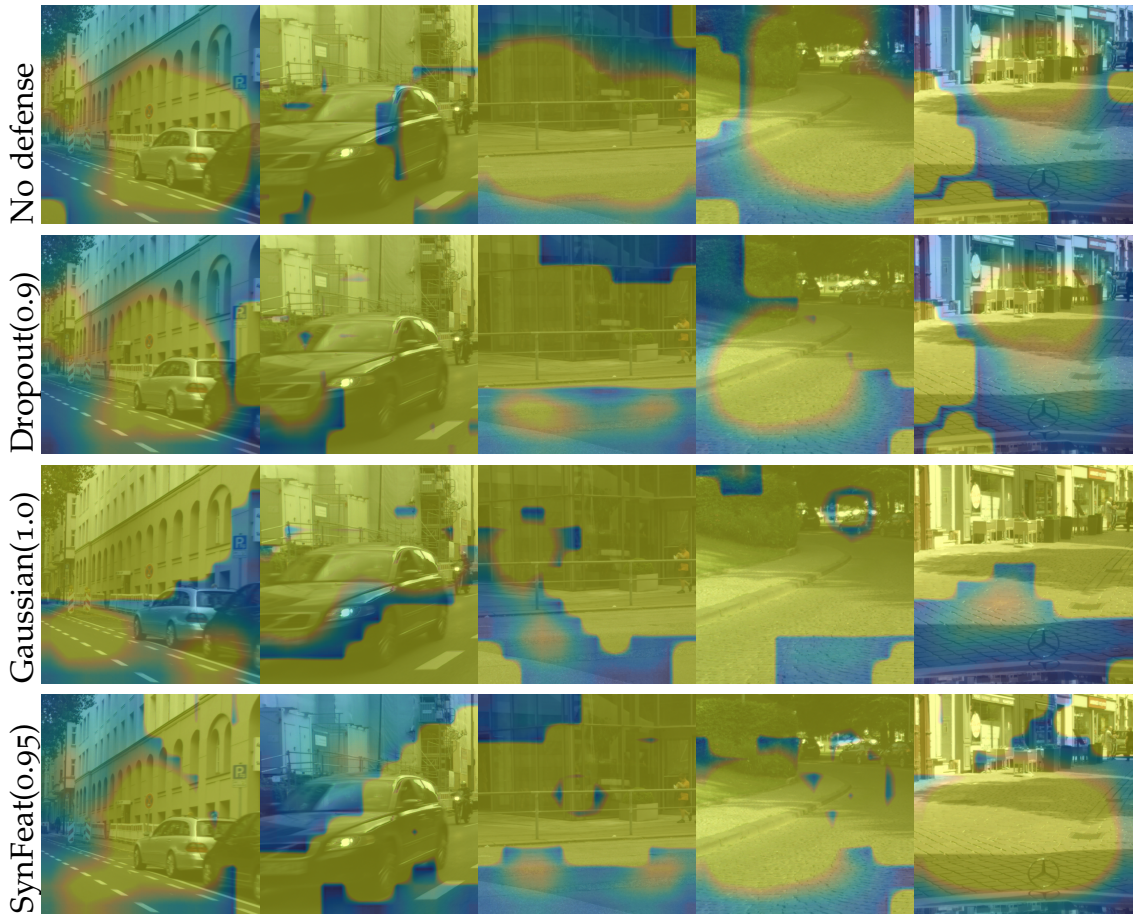


Figure 7.9: Class activation mapping (Zhou *et al.*, 2016) regions from attackers trained on *BDD100K* in patch-level prediction, contributing to recognition as a part of training data. All those examples are from training set of *Cityscapes*.

7.4.2.2 *Preserving utility*

Except protecting membership from stealing by attackers, a decent model ideally provides high quality recognition performance as well. As a result, we further explore and compare the segmentation results with different defense strategies and strength. In Table 7.2, we report the quantitative results of four widely used metrics for semantic segmentation (i.e., pixel accuracy (PixAcc), class accuracy (ClsAcc), mean intersection of union (mIoU) and frequent weighted intersection of union (fwIoU)). Because **Argmax** will not change the predictions, we do not show this defense in the table. To begin with, we observe that **Gaussian** lose performance drastically when increase the variance , while it is easier to leak the training data than our approach. In addition, **Dropout** is able to segment images at good quality. Finally, the segmentation performance of our method only decreases a little, even applying a large defense factor.

In Figure 7.10, we show a segmentation example. Similar to Table 7.2, Gaussian

Table 7.2: Quantitative comparison of segmentation performance with different defenses on *Cityscapes*.

Defense	Split	PixAcc	ClsAcc	mIoU	fwIoU
No		97.8	93.5	88.6	95.9
Dropout(0.1)		97.8	93.2	88.7	95.8
Dropout(0.9)		97.1	91.6	85.7	94.9
Gaussian(0.5)	Train	92.2	85.9	73.1	86.4
Gaussian(1.0)		81.3	76.6	52.8	70.4
SynFeat (0.75)		97.1	91.7	87.0	94.9
SynFeat (0.95)		96.6	89.8	82.8	93.8
No		96.3	86.3	79.7	93.2
Dropout(0.1)		96.3	86.3	79.7	93.1
Dropout(0.9)		95.8	84.1	76.6	92.2
Gaussian(0.5)	Val	90.8	79.4	65.7	84.1
Gaussian(1.0)		80.2	71.3	49.2	69.6
SynFeat (0.75)		95.7	80.5	74.0	92.0
SynFeat (0.95)		95.0	75.0	67.3	91.2

is significantly affected and returns poor results. With variance 1.0, it produces an extreme noisy mask with many dots and inconsistent regions. In spite of the similar quantitative performance of **Dropout** to ours, some processing can be recognized with this defense, as it produces more noises in boundaries, because dropout operations with large ratio loss too much information, and then hard to emit good results on rich detailed regions. In contrast, there are no obvious processing signs in our predictions for different defense factors. Even though **SynFeat(0.75)** and **SynFeat(0.95)** have comparable and worse quantitative performance comparing to **Dropout(0.95)**, we achieve more smooth and natural predictions at stronger security guarantee, as shown in Figure 7.5, 7.6, 7.10.

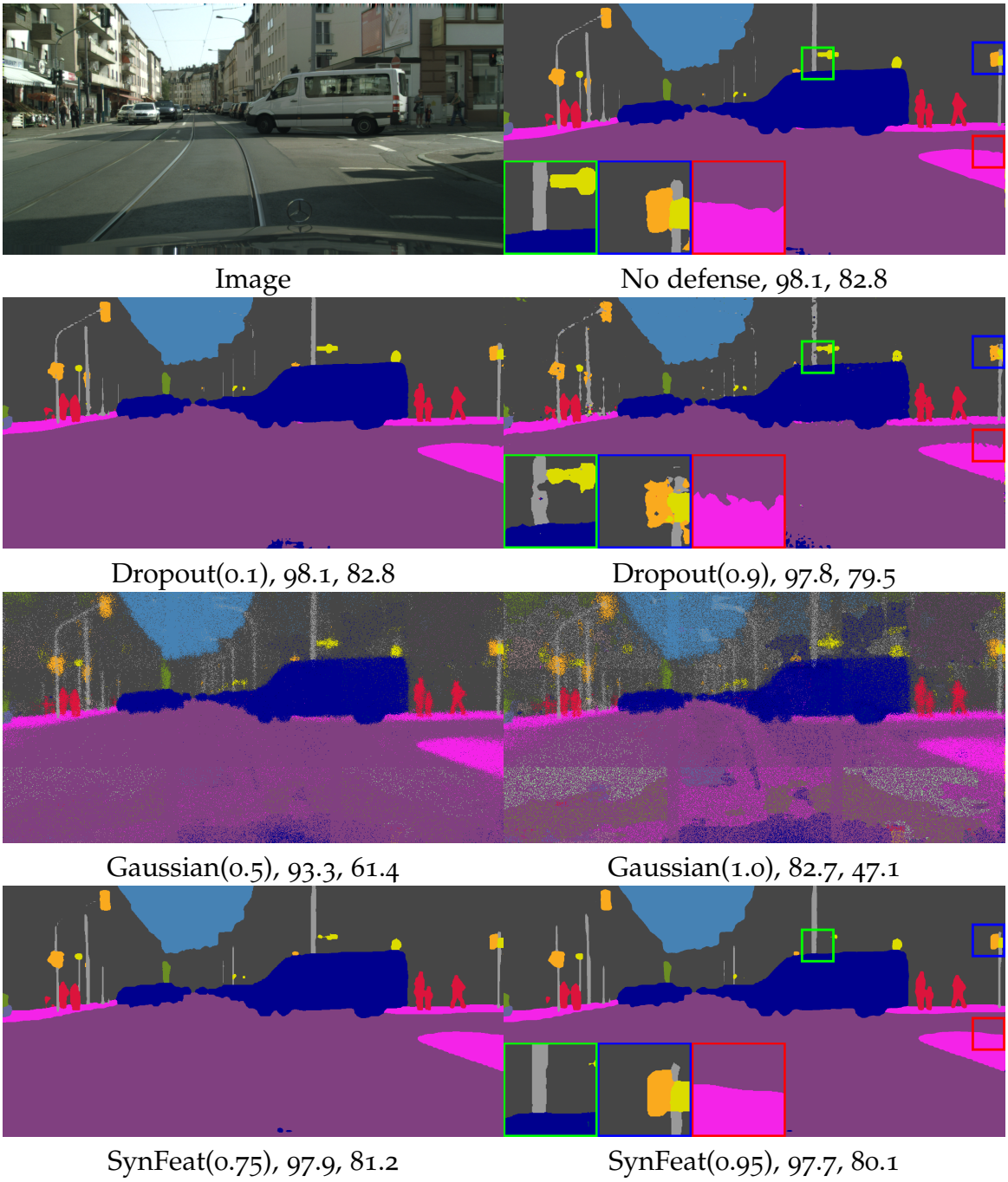


Figure 7.10: Qualitative comparison of segmentation with defenses. PixelAcc and mIoU of segmentation results are presented, respectively.

Furthermore, to have a better understanding and observation on the properties of individual defense methods, we test them with more defense strengths and plot joint attack-segmentation plots in Figure 7.11 and 7.12, which give intuitive observations and comparisons. In each figure, X-axis indicates the attack performance, that

we use area under ROC curves (AUC) to quantify it, and higher X-values refer to more successful membership inference attacks are achieved. Y-axis indicates the segmentation performance that we use mIoU from segmentation performance to draw those figures, and then higher Y-values are expected to maintain better segmentation. Except all the defenses, we also plot the optimal models and black-box models without any defense. Optimal models preserve segmentation performances completely and have the chance level accuracy in ROC curves, therefore, they locate at the left-top corners of all the plots.

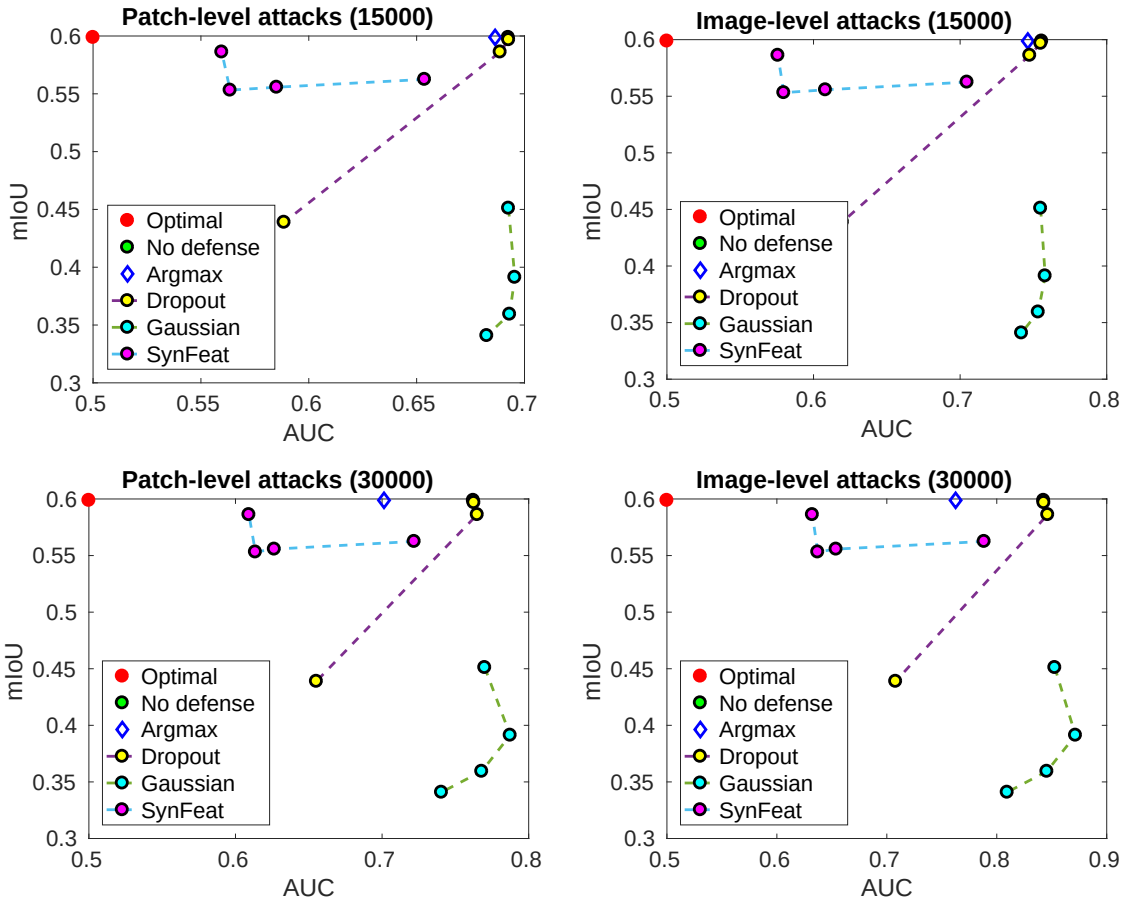


Figure 7.11: Joint attack-segmentation plots for dependent attacks.

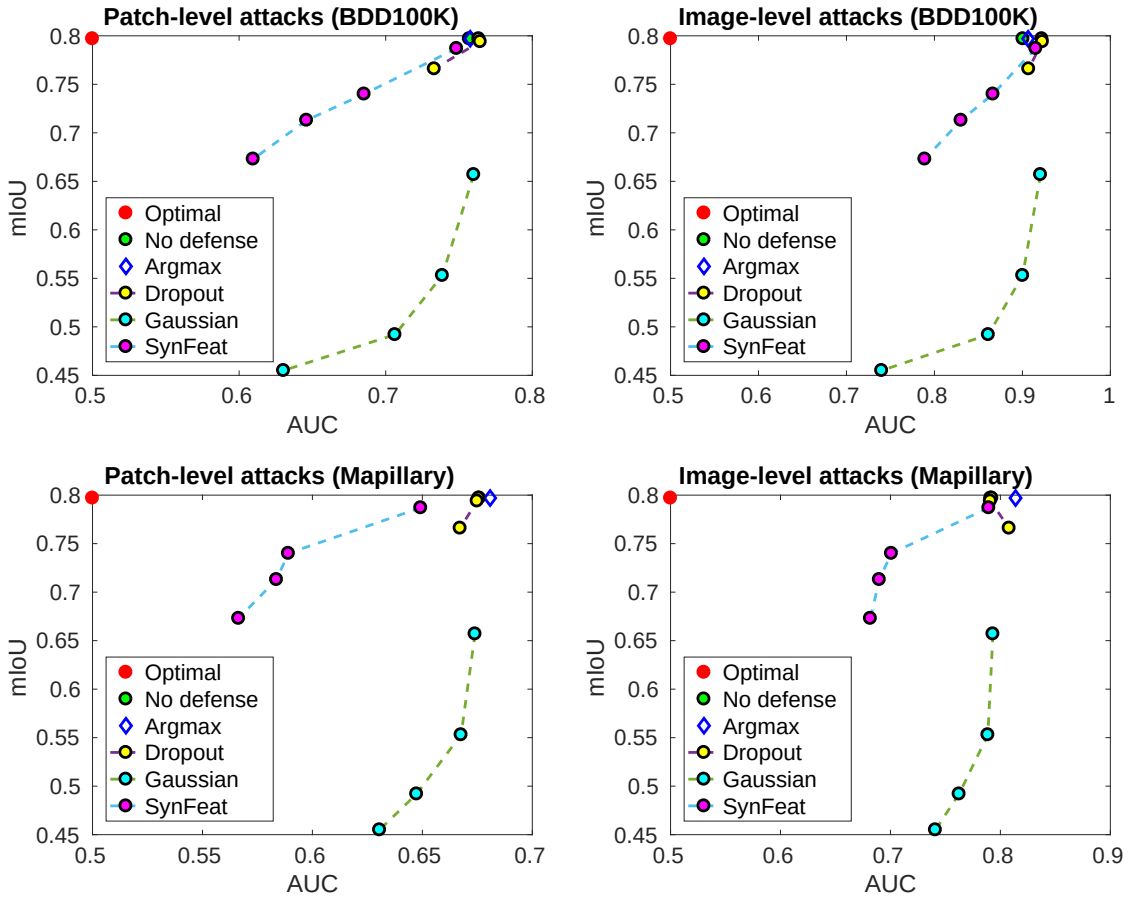


Figure 7.12: Joint attack-segmentation plots for independent attacks.

7.5 DISCUSSION

Our study provides the first membership inference attack system against black-box semantic segmentation models. We show the risk of information leakage on training data for semantic segmentation models with dense structural outputs, The experimental results show that training data leakage happens under data/model dependent and independent settings. The security risk potentially threatens the safety usage of segmentation models, which have broad applications, such as autonomous driving vehicles and indoor navigation robotics, as discussed in this thesis. Besides, we also present a set of comparisons on different defenses for protecting membership privacy, including our proposed method based on synthetic feature and posterior obfuscation, and we demonstrate our proposed method perform well in protecting membership privacy while preserving segmentation performance.

7.5.1 Application Scenarios

The applications of our study in this chapter are mainly two aspects, with respective to attacks and defenses.

First, the attack technique is useful as a third-party examiner, with suitable management. Even though we show the information leakage is a risk for service provider, it allows consumers to obtain evidences for forensics purposes. In some applications, such as medical diagnosis systems, information of patients are very sensitive, because of privacy issues. Without patients' agreements, a model should not be trained with their data, therefore, attacking techniques allows patients to judge if their scans are used by service providers or not. Besides consumers' privacy, ownership is another important issue, which is highly relevant to the model builders' profits. By analysing the distribution of posteriors, the ownership of a released model can be determined.

Second, the defense techniques are very important for protecting the profit of service providers and their users who already agree the usage of their data. From our study, the information leakage happens easily with querying examples according to released API or models. This attack may be detected by servers if frequent queries happened from a same group of users. However, our independent attack shows that a segmentation model is probably attacked even without any query. A malicious adversary only needs to know the label space of a victim, the attacks can be implemented, which is extremely hard to detect and avoid. Therefore, our study on defenses is important, which has been shown effective mitigation of information leakage, such as applying dropout during test time, or our synthetic features based obfuscation. For a real-world application, it is also possible to combine multiple effective methods for the final defense, to avoid attacks.

7.5.2 Technical Limitations

In our study, we need images with dense annotations to determine if current paired sample was used during training or not. Therefore, the requirement of large-scale labeled images limits the practise of attacks, even though we show the second adversaries without data distribution and model assumptions. Moreover, an interesting question is naturally asked: can we only provide image samples to train an adversary, or use limited annotation such as point level annotations (Bansal *et al.*, 2017; Bearman *et al.*, 2016). Point annotations are much easier to acquire compared to dense annotations, and need less precise boundaries.

Furthermore, in our independent attacks, we might apply different network architectures and training data to achieve attacks, according to the general fact that confidence scores for training data are higher than testing data. However, their overall performance or confidences have a distribution difference for different segmentation models. A stronger segmentation model normally has higher confidences than a weaker one for all the data points. Therefore, the overall distribution difference is a potential challenge we need to deal with, for more effective attacks for model/data independent adversary.

7.6 CONCLUSION

We have provided the first study of membership inference attacks for semantic segmentation models by proposing the first attacks and defenses in this scenario. We show significant risks of information leakage on the training set of a black-box model, under various practical attacking conditions. In order to deal with this privacy and security risk, we investigate various defenses, and propose a novel defense mechanism based on synthetic features. The proposed defense is able to reduce the distribution gap between training data and unseen samples, which effectively reduce the power of the attack. Finally, extensive experiments on real world images and state of the art models for street scene segmentation provide a realistic assessment of membership inference attacks and defenses for semantic segmentation models.

8 CONCLUSIONS AND FUTURE PROSPECTS

Deep learning has reshaped the research of computer vision with GPU parallel computing and large-scale databases. Semantic image segmentation, which stands at the core place of computer vision and machine perception, has been studied and boosted with fully convolutional networks (FCN) (Long *et al.*, 2015) recently. In this thesis, we focus on this fundamental task with FCN, and present improved methods and analysis on several issues of semantic image segmentation. Model and data play important roles in achieving successful machine learning models, therefore, we explore the research of improving semantic segmentation systems in terms of neural architectures/modules and more effective leverage of data with privacy considerations. In details, we conclude this thesis with the following two points: (1) designing data-driven neural modules which are compatible with existing convolution architectures and (2) exploring issues and analyzing the effectiveness of training data for more accurate segmentation and stronger guarantee on data privacy. To begin with, we show flexibility and complexity are still important in data-driven models, whose parameters are learned from data. Even though convolutional networks perform surprisingly well in various computer vision tasks, our models are able to further boost the existing basic architectures by incorporating data priors or learning more adaptive convolution operations, which bring more flexibility into a network. Besides, we point out the connections between model design and leverage of training data. In a summary, suitable architectures are indispensable for better utilization of training data with different levels of supervisions in terms of better segmentation performances and data privacy guarantees, which shows a promising direction of future works.

To begin with, we develop data-driven modules to introduce data priors into neural networks and learn more flexible knowledge with deep neural networks, for improving the performance of semantic segmentation. Even though convolutional neural networks are already data-driven models, our approaches allows a network to perform adaptive context aggregation instead of fixed receptive field for all pixels, which is crucial for semantic segmentation. In Chapter 3, we introduce superpixel correspondences into a network, which allows us to leverage large-scale unlabeled images from video sequences. Besides, we also show promising results of applying superpixels in neural networks to aggregate contextual information, which can be further explored with stronger superpixel computation. In Chapter 4, we investigate dilated convolution operations, which are widely used in semantic segmentation. We avoid to set dilation factors for individual layers, and instead learn those factors from data coupling with filter weights. We test our learnable dilated convolution with different segmentation baselines on several public datasets, and

achieve consistent improvements. In conclusion, our approaches allow us to improve semantic segmentation at different data accesses, i.e., we conduct semantic image segmentation from video sequences and still images. Besides, we study semantic segmentation tasks under different data formats and scene types, including indoor scenes with RGB-D sequences, outdoor scenes and mixture of them with RGB images. Accordingly, a set of analysis and comparisons to previous state-of-the-art models demonstrate the effectiveness of our proposed methods. Regarding to the future research of semantic segmentation, data-driven approaches and adaptive context modelling are still open questions and critical issues. In natural images, there are large objects as well as small ones in a same scene, therefore, a success model is not supposed to predict their labels with same contextual information. Besides, learning hyperparameters become increasingly popular in recent years, such as AutoML (Zoph and Le, 2016). However, AutoML needs huge computation resources, therefore, learning hyperparameters with differential backpropagation is interesting and a promising solution, which is possible able to combine with AutoML and easier to adapt a learned network to new data distribution.

In addition, we explore the research on leverage training data effectively and safely. Data annotations for semantic segmentation, which need polygon regions, are quite expensive. Hence, we design a data augmentation method in chapter 6, building on the generative modeling of CNN features. We present a solution based on generative adversarial networks, which takes a semantic mask as an input, and produces the corresponding features. The synthetic features have been shown effective, that we mix them with real images for training a segmentation model, and achieve consistent improvement over several datasets and settings. Besides, we also study the diversity of synthetic data in chapter 5 from a stochastic regression-based conditional generative model. We improve the diversity by sampling neighbors with similar conditional inputs by enforcing a network to learn one-to-many mapping. Furthermore, a real-world application is not only interested in pursuing the limitation of segmentation performance, but also providing safe and clean services in terms of data security and privacy of users. Since membership privacy of classification models can be inferred as pointed out by Shokri *et al.* (2017), it is also important to study the security issue for semantic segmentation, and has been studied in Chapter 7. As a result, we present the first membership inference attack system in semantic segmentation, as well as several protection regimes, including our synthetic features based prediction obfuscation method. As a summary, this thesis provide better configurations of semantic segmentation models for stronger and safer usage in real-world applications. To summarize the second part of this thesis, we bridge the connection between two reverse task segmentation and generation, and show generation is able to improve segmentation as an auxiliary task in terms of performance and security issues. This would be a promising direction to design stronger segmentation model in the future.

In this chapter we further depict the contributions of this thesis (Section 8.1) and review open problems as well as potential future prospects (Section 8.2).

8.1 DISCUSSION OF CONTRIBUTIONS

The overall goal of this thesis is to improve fully convolutional networks for semantic segmentation in real-world applications. We investigate two points with respective to training segmentation models: *neural architectures for improving semantic segmentation models* and *better usage of training data in real-world applications*. In the following, we will discuss the main observations and insights of this thesis with respect to the individual chapters.

8.1.1 Neural Architectures for Improving Semantic Segmentation

The first part of this thesis aims to design basic neural modules for improving the performance of semantic segmentation.

Current deep neural networks are learned with back-propagation (Rumelhart *et al.*, 1988) based on chain rules. Therefore, all the layers (or modules) in a network receive gradients from next layers to update their parameters, and propagate the gradients to previous layers. Based on the learning scheme, we present two novel layers which are fully differentiable.

In Chapter 3, we propose a spatio-temporal data-driven pooling (STD2P) layer for semantic image segmentation from a video sequence. Our STD2P introduces superpixel priors into a semantic segmentation network to refine original predictions. Besides, we are able to establish region correspondences between individual frames of a sequence, and feed those correspondences into a segmentation network, and perform multi-view semantic segmentation that information are projected onto a target frame and aggregate those different regions of a same object into final outputs. Dense annotations of videos are expensive, however, our model is able to leverage partially annotated videos, and utilize the unlabeled frames of an input video. Even though our model is trained with multi-view information, it still performs better on single frame prediction comparing to baseline fully convolutional networks in indoor scene semantic segmentation. To conclude, our main observations are as follows. (1) Introducing superpixel priors into segmentation networks helps semantic segmentation, in particular, it leads to more precise predictions on rich boundary areas as well as more smooth predictions on very large regions. (2) Leveraging multi-view information with our region-based correspondences helps semantic segmentation with respective to recognition accuracy. (3) Our semi-supervised learning framework effectively utilizes unlabeled frames of partially annotated videos for training a stronger semantic segmentation model.

In Chapter 4, we propose a learnable dilated convolution layer which learns dilation factors as well as convolution filter weights jointly. Only difference of our method to traditional convolutions (Yu and Koltun, 2016) is that we regard dilation factors as a learnable parameter, which is able to be a positive fractional number instead of a positive integer only. We apply bilinear interpolation to handle fractional dilation factors in 2D convolution, and gradients are splitted into the

four neighbors of a sampled location. Besides, we further explored the channel-wise dilated convolutions, which capture wide context as well as local details simultaneously. To conclude, our main observations are as follows. (1) For different datasets, our segmentation models with learnable dilated convolutions finally obtain different dilation factors, and achieve better performance than baselines. (2) For different locations of convolution layers, our approach is able to learn different dilation factors for them. (3) Extensive studies demonstrate the effectiveness of proposed learnable dilated convolutions.

8.1.2 Better Usage of Training Data in Real-world Applications

In the last three chapters, we summarize the investigation and analysis of effective and safe modeling and utilization of training data. We present generative modeling of training data as well as applications in improving semantic segmentation with data augmentation and protecting membership privacy of a black-box semantic segmentation model.

In Chapter 5, we study the conditional image generation problem, that our goal is to improve diversity of generated examples while maintaining the quality of generation results. Specifically, we generate human faces and animal heads from facial landmarks and normal maps, respectively. We regard conditional image generation directly as a regression problem, and construct our model based on popular cascaded refinement networks (CRN) (Chen and Koltun, 2017), which is stable in training. While CRN could output multiple different examples with multiple choice learning framework (Guzman-Rivera *et al.*, 2012), it lacks of sampling capability of novel examples during inference due to its deterministic architecture. To overcome this drawback, we introduce latent random variables into a network with channel-wise dropout layers, namely latent dropout codes in our approach, leading to a stochastic network for the regression task. Furthermore, we explore neighbor enhanced loss function, which samples examples with close enough conditional inputs, and enforce a network to learn one-to-many mapping approximately and directly. To conclude, our main observations are as follows. (1) Our generator is able to synthesize examples with comparable performance to baseline models when sample the same number examples. In contrast, we are able to generate better examples by sampling new latent dropout codes. (2) Our generator is able to output synthetic examples with varying local characters and global structures. We evaluate the diversity of synthetic examples quantitatively and qualitatively.

In Chapter 6, we explore CNN feature generation and apply our synthetic features as data augmentation strategy to improve the training of semantic segmentation models. Conditional image generator (also refers to image translation) transforms an input condition image into an output image with the same spatial resolution (Isola *et al.*, 2017; Zhu *et al.*, 2017b). Different to generating images, features have compressed spatial resolution and much higher channel numbers. Therefore, we modify previous generative adversarial networks (GANs) for feature generation. To represent high resolution condition inputs, we add several atrous spatial pyramid pooling

(ASPP) layers at the entrance of the generator and the conditional discriminator of GANs. Finally, we show that applying our synthetic features with several strategies (i.e., hard negative mining, label smoothing regularization) achieves clear improvements. Particularly, several categories have been significantly boosted, such as Bus, Rider, Wall, which are the key classes in street scene parsing scenarios. To conclude, our main observations are as follows. (1) Generating synthetic features is more feasible to generating synthetic images in applying those as data augmentation strategy for semantic segmentation. (2) With incorporating synthetic features at different ratio between real images and synthetic features, the performance is always able to improve, which demonstrates the effectiveness of synthetic features. (3) Leveraging additional semantic layout masks is helpful to improve semantic segmentation further.

In Chapter 7, we investigate membership inference attacks against black-box semantic segmentation models, which is the first study on this topic for semantic segmentation. For semantic segmentation, we reveal the dangerous usage of training set in nowadays machine learning services, that a data pair can be determined if it is a part of training set by various adversaries. Specifically, the first setting is data and model dependent attacks with querying from APIs or servers. Because a model usually fits better on the training data than others, the confidence distribution gap provides possibility to classify between the training data and others as pointed out in Shokri *et al.* (2017). Similarly, our second setting is more feasible to carry out, that we do not have any interactions with servers and assumptions on the model configurations or data distributions. To mitigate the training data leakage, we develop an approach based on our synthetic features, which reduces the confidence distribution gap. Instead of returning the posterior of an query input, our approach reports an obfuscated posterior from a synthetic feature. Therefore, the confidence distribution of testing data would bias to the training set, since the generator is learned on the training set, and thus make the binary classification of adversaries harder. To conclude, our main observations are as follows. (1) Information leakage of training data happens under various attacking scenarios. (2) Semantic segmentation models involved with synthetic features significantly prevent membership inference attacks.

8.2 FUTURE PROSPECTS

In this section, we discuss several remaining challenges of our methods for semantic segmentation systems in real-world settings, and possible solutions to them.

Flexible context modelling in networks. Global context and hierarchical context have been exploited in achieving recent state-of-the-art semantic segmentation models. Although semantic segmentation has achieved significant improvements (Zhao *et al.*, 2017; Zhang *et al.*, 2018a; Chen *et al.*, 2018c,a), it still fails to perform well in some cases, such as objects with extreme scales or some challenging categories. By observing those state-of-the-art approaches, their network architectures have the same level of context for different locations, which potentially limit the development of semantic segmentation, since different categories might need different context information. Besides, there are also several works to address the flexibility of context with adaptive receptive modeling (Zhang *et al.*, 2017b) or applying superpixels (He *et al.*, 2017a; Caesar *et al.*, 2016; Gadde *et al.*, 2016), however, they achieve less performance improvement compared to current state-of-the-arts, although they provide more flexible context to networks. The performance of superpixel based approaches suffer from imperfect superpixel computations, and most approaches employ only one level of superpixels, which are either too fine (lacking of wider context) or too coarse (lacking of local details). The first possible solution is to leverage superpixel hierarchies in networks. In spired by the recent success of semantic segmentation, raw feature maps before context aggregation modules also encode many useful detailed information, therefore, strong segmentation models might need to apply superpixels in other way that utilizes superpixels to provide context instead of directly predicting labels on them, similar to previous work. Second, superpixels perform indecent on the regions with rich details and boundaries, hence designing a mechanism to handle the imperfect boundaries is necessary. A possible solution is to smooth the boundary of superpixels, in other words, the pixel at the border of two neighboring superpixels contribute both of the superpixels.

Investigation of applying cheap data. The amount of data play a crucial role in training deep neural networks, but dense annotations for semantic segmentation is quite expensive. Therefore, leveraging cheap data with annotations is still interesting and important for research and industry applications. Applying synthetic data from a generator or a graphic engine becomes popular recently. Research on leveraging synthetic data is still not enough to significantly boost training a semantic segmentation. Besides, numerous works focus on domain adaptation problems, that train a model with synthetic images and test on real images. Even though domain adaptation is an important task, very few work shows incorporating synthetic images in training achieves significant improvements than the model using real images only because of huge distribution gap between them. The distribution gap between synthetic data and real data comes from not only appearances but also semantic layouts. Therefore, diminishing layout gap could be promising in further

research. One of example for manipulating semantic layouts is Hong *et al.* (2018), which potentially shows the success in the processing of semantic layout as well as appearance. Second, collecting videos are much cheaper than annotations. Therefore, similar to our work in Chapter 3, effectively leveraging the redundancy of a video is still a promising open question. Instead of labeling equidistant frames, selecting most informative frames or regions is interesting, which is able to save much effort in annotating data.

Imbalanced training data. Due to natural distribution differences for individual classes, training data of each class for semantic segmentation is extremely imbalanced. For example, in *Cityscapes* dataset for street scene parsing, road class has $100\times$ and $800\times$ labeled pixels comparing to people and rider classes. Consequently, it remains to handle imbalanced data distribution with more effective strategy, rather than simply re-weighting each class in loss function. AutoML receives much attention recently, which aims to learn a machine learning model configurations, such as neural architecture (Zoph and Le, 2017), nonlinear activations (Ramachandran *et al.*, 2017) or data augmentation (Cubuk *et al.*, 2019). Inspired by the idea of AutoML, it is interesting to learn a regularizer for training a segmentation model under imbalanced training data. It might be more promising and effective than traditional regularizers like label smoothing regularization and simple re-weighting for difference classes, which regularize all the pixels equally.

Membership privacy in dense prediction. Semantic segmentation is a dense prediction task, which returns a posterior for each location. Besides, different locations compose a structured prediction, which also possibly leak training data information. In Chapter 7, we propose a method to reduce the distribution gap between training data and others. To further protect the membership privacy of black-box semantic segmentation models, it is necessary to obfuscate output structures, instead of posteriors only. Furthermore, the idea of protecting structures is able to extend to other dense prediction tasks, such as depth estimation. Particularly, depth estimation only outputs a value for each location, rather than a posterior vector like semantic segmentation. Therefore, study on protecting structures is another potential avenue of preventing information leakage.

LIST OF FIGURES

LIST OF TABLES

BIBLIOGRAPHY

M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang (2016). Deep learning with differential privacy, in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security 2016*. Cited on page 28.

R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk (2012). SLIC superpixels compared to state-of-the-art superpixel methods, *IEEE transactions on pattern analysis and machine intelligence*, vol. 34(11), pp. 2274–2282. Cited on page 14.

S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. Lawrence Zitnick, and D. Parikh (2015). Vqa: Visual question answering, in *ICCV 2015*. Cited on page 103.

A. Antoniou, A. Storkey, and H. Edwards (2017). Data Augmentation Generative Adversarial Networks, *arXiv preprint arXiv:1711.04340*. Cited on page 24.

P. Arbeláez, B. Hariharan, C. Gu, S. Gupta, L. Bourdev, and J. Malik (2012). Semantic segmentation using regions and parts, in *CVPR 2012*. Cited on page 33.

P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik (2011). Contour detection and hierarchical image segmentation, *TPAMI*. Cited on page 48.

A. Arnab, S. Jayasumana, S. Zheng, and P. H. Torr (2016). Higher order conditional random fields in deep neural networks, in *ECCV 2016*. Cited on page 31.

M. Backes, P. Berrang, M. Humbert, and P. Manoharan (2016). Membership privacy in MicroRNA-based studies, in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security 2016*. Cited on page 27.

V. Badrinarayanan, A. Kendall, and R. Cipolla (2017). Segnet: A deep convolutional encoder-decoder architecture for image segmentation, *IEEE transactions on pattern analysis and machine intelligence*, vol. 39(12), pp. 2481–2495. Cited on pages 1 and 12.

A. Bansal, X. Chen, B. Russell, A. G. Ramanan, *et al.* (2017). Pixelnet: Representation of the pixels, by the pixels, and for the pixels, *arXiv preprint arXiv:1702.06506*. Cited on pages 17, 75, and 121.

A. Bansal, Y. Sheikh, and D. Ramanan (2018). PixelNN: Example-based Image Synthesis, *ICLR*. Cited on pages 75 and 76.

D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba (2017). Network Dissection: Quantifying Interpretability of Deep Visual Representations, in *CVPR 2017*. Cited on page 72.

A. Bearman, O. Russakovsky, V. Ferrari, and L. Fei-Fei (2016). What's the point: Semantic segmentation with point supervision, in *European conference on computer vision 2016*. Cited on page 121.

C. Bowles, L. Chen, R. Guerrero, P. Bentley, R. Gunn, A. Hammers, D. A. Dickie, M. V. Hernández, J. Wardlaw, and D. Rueckert (2018). GAN Augmentation: Augmenting Training Data using Generative Adversarial Networks, *arXiv preprint arXiv:1810.10863*. Cited on pages 24 and 25.

A. Brock, J. Donahue, and K. Simonyan (2019). Large scale gan training for high fidelity natural image synthesis, in *ICLR 2019*. Cited on page 23.

G. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla (2008). Segmentation and recognition using structure from motion point clouds, *ECCV*. Cited on pages 20 and 58.

C. Cadena and J. Košecká (2014). Semantic segmentation with heterogeneous sensor coverages, in *ICRA 2014*. Cited on page 14.

H. Caesar, J. Uijlings, and V. Ferrari (2016). Region-based semantic segmentation with end-to-end training, in *ECCV 2016*. Cited on pages 15, 31, 33, 38, 39, 40, 41, 47, 48, 50, 128, 132, and 137.

L.-C. Chen, M. Collins, Y. Zhu, G. Papandreou, B. Zoph, F. Schroff, H. Adam, and J. Shlens (2018a). Searching for efficient multi-scale architectures for dense image prediction, in *NeurIPS 2018*. Cited on pages 16, 64, 111, and 128.

L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille (2014). Semantic image segmentation with deep convolutional nets and fully connected crfs, in *ICLR 2014*. Cited on pages 38, 39, 40, 41, 47, and 137.

L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille (2018b). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, *IEEE transactions on pattern analysis and machine intelligence*, vol. 40(4), pp. 834–848. Cited on pages 1, 5, 12, 13, 15, 16, 38, 39, 40, 41, 47, 48, 53, 57, 61, 62, 63, 91, 132, 133, and 137.

L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam (2018c). Encoder-decoder with atrous separable convolution for semantic image segmentation, in *ECCV 2018*. Cited on pages 1, 17, 111, and 128.

Q. Chen and V. Koltun (2017). Photographic Image Synthesis with Cascaded Refinement Networks, in *ICCV 2017*. Cited on pages 6, 27, 70, 71, 74, 75, 76, 79, 82, 83, 84, 87, 126, and 133.

M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele (2016). The cityscapes dataset for semantic urban scene understanding, in *CVPR 2016*. Cited on pages 1, 20, 57, 58, 92, 103, and 109.

C. Couprie, C. Farabet, L. Najman, and Y. LeCun (2013). Indoor semantic segmentation using depth information, in *ICLR 2013*. Cited on pages 14, 19, 38, and 44.

E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le (2019). AutoAugment: Learning Augmentation Strategies From Data, in *CVPR 2019*. Cited on page 129.

J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei (2017). Deformable convolutional networks, in *CVPR 2017*. Cited on page 17.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei (2009). Imagenet: A large-scale hierarchical image database, in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on 2009*. Cited on pages 1 and 11.

Z. Deng, S. Todorovic, and L. Jan Latecki (2015). Semantic Segmentation of RGBD Images with Mutex Constraints, in *CVPR 2015*. Cited on pages 33, 38, 39, 40, 41, 47, and 137.

P. Dollár and C. Zitnick (2013). Structured forests for fast edge detection, in *CVPR 2013*. Cited on page 35.

A. Dosovitskiy and T. Brox (2016). Generating images with perceptual similarity metrics based on deep networks, in *Advances in Neural Information Processing Systems 2016*. Cited on pages 23, 26, and 73.

A. Dosovitskiy, J. Tobias Springenberg, and T. Brox (2015). Learning to generate chairs with convolutional neural networks, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2015*. Cited on page 85.

C. Dwork (2011). Differential privacy, *Encyclopedia of Cryptography and Security*. Cited on page 28.

D. Eigen and R. Fergus (2015). Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture, in *ICCV 2015*. Cited on pages 38, 39, 40, 41, 44, and 137.

M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman (2010). The pascal visual object classes (voc) challenge, *IJCV*, vol. 88(2). Cited on page 57.

C. Farabet, C. Couprie, L. Najman, and Y. LeCun (2012). Learning hierarchical features for scene labeling, *IEEE transactions on pattern analysis and machine intelligence*, vol. 35(8), pp. 1915–1929. Cited on page 14.

P. F. Felzenszwalb and D. P. Huttenlocher (2004). Efficient graph-based image segmentation, *International journal of computer vision*, vol. 59(2), pp. 167–181. Cited on page 14.

V. Fischer, M. C. Kumar, J. H. Metzen, and T. Brox (2017). Adversarial examples for semantic image segmentation, *arXiv preprint arXiv:1703.01101*. Cited on page 28.

M. Frid-Adar, E. Klang, M. Amitai, J. Goldberger, and H. Greenspan (2018). Synthetic Data Augmentation using GAN for Improved Liver Lesion Classification, *arXiv preprint arXiv:1801.02385*. Cited on page 24.

R. Gadde, V. Jampani, M. Kiefel, and P. V. Gehler (2016). Superpixel Convolutional Networks using Bilateral Inceptions, in *ECCV 2016*. Cited on pages 15, 31, 33, 38, 39, 40, 41, 47, 48, 50, 128, 132, and 137.

F. Galasso, N. S. Nagaraja, T. J. Cardenas, T. Brox, and B. Schiele (2013). A Unified Video Segmentation Benchmark: Annotation, Metrics and Analysis, in *ICCV 2013*. Cited on page 48.

L. A. Gatys, A. S. Ecker, and M. Bethge (2016). Image style transfer using convolutional neural networks, in *CVPR 2016*. Cited on page 26.

A. Geiger, P. Lenz, and R. Urtasun (2012). Are we ready for autonomous driving? the kitti vision benchmark suite, in *CVPR 2012*. Cited on pages 1 and 103.

G. Ghiasi and C. C. Fowlkes (2016). Laplacian pyramid reconstruction and refinement for semantic segmentation, in *ECCV 2016*. Cited on page 91.

K. Gong, X. Liang, D. Zhang, X. Shen, and L. Lin (2017). Look into person: Self-supervised structure-sensitive learning and a new benchmark for human parsing, in *CVPR 2017*. Cited on page 1.

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio (2014). Generative adversarial nets, in *NeurIPS 2014*. Cited on pages 2, 6, 23, 69, 85, 87, and 108.

S. Gould, J. Rodgers, D. Cohen, G. Elidan, and D. Koller (2008). Multi-class segmentation with relative location prior, *International Journal of Computer Vision*, vol. 80(3), pp. 300–316. Cited on page 14.

M. Grundmann, V. Kwatra, M. Han, and I. Essa (2010). Efficient Hierarchical Graph Based Video Segmentation, in *CVPR 2010*. Cited on page 33.

I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville (2017). Improved training of wasserstein gans, in *Advances in Neural Information Processing Systems 2017*. Cited on page 23.

S. Gupta, P. Arbelaez, and J. Malik (2013). Perceptual organization and recognition of indoor scenes from RGB-D images, in *CVPR 2013*. Cited on pages 19 and 38.

S. Gupta, R. Girshick, P. Arbeláez, and J. Malik (2014). Learning rich features from RGB-D images for object detection and segmentation, in *ECCV 2014*. Cited on pages 14, 35, 37, 38, 39, 40, 41, 48, and 137.

A. Guzman-Rivera, D. Batra, and P. Kohli (2012). Multiple choice learning: Learning to produce multiple structured outputs, in *NIPS 2012*. Cited on pages 27, 70, 73, 74, and 126.

B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik (2015). Hypercolumns for object segmentation and fine-grained localization, in *CVPR 2015*. Cited on page 14.

K. He, X. Zhang, S. Ren, and J. Sun (2016). Deep residual learning for image recognition, in *CVPR 2016*. Cited on pages 1, 13, 57, 61, 109, 111, and 138.

Y. He, W.-C. Chiu, M. Keuper, and M. Fritz (2017a). STD2P: RGBD semantic segmentation using spatio-temporal data-driven pooling, in *CVPR 2017*. Cited on pages 8 and 128.

Y. He, M. Keuper, B. Schiele, and M. Fritz (2017b). Learning dilation factors for semantic segmentation of street scenes, in *GCPR 2017*. Cited on page 8.

Y. He, B. Schiele, and M. Fritz (2018). Diverse Conditional Image Generation by Stochastic Regression with Latent Drop-Out Codes, in *ECCV 2018*. Cited on page 8.

Y. He, B. Schiele, and M. Fritz (2019a). DFGAN: Synthetic Dense Features for Improved Semantic Segmentation, in *submission 2019*. Cited on page 8.

Y. He, B. Schiele, and M. Fritz (2019b). Segmentations Leak: Membership Inference Attacks on Black-Box Semantic Segmentation Models, in *ICCV submission 2019*. Cited on page 9.

A. Hermans, G. Floros, and B. Leibe (2014). Dense 3D semantic mapping of indoor scenes from RGB-D images, in *ICRA 2014*. Cited on pages 38 and 44.

G. E. Hinton and R. R. Salakhutdinov (2006). Reducing the dimensionality of data with neural networks, *Science*, vol. 313(5786), pp. 504–507. Cited on pages iii, v, and 1.

J. Hoffman, D. Wang, F. Yu, and T. Darrell (2016). Fcns in the wild: Pixel-level adversarial and constraint-based adaptation, *arXiv preprint arXiv:1612.02649*. Cited on page 87.

S. Hong, X. Yan, T. S. Huang, and H. Lee (2018). Learning hierarchical semantic image manipulation through structured representations, in *NIPS 2018*. Cited on page 129.

J. Hosang, R. Benenson, P. Dollár, and B. Schiele (2015). What makes for effective detection proposals?, *IEEE transactions on pattern analysis and machine intelligence*, vol. 38(4), pp. 814–830. Cited on page 14.

A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications, *arXiv preprint arXiv:1704.04861*. Cited on page 13.

G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger (2017). Densely connected convolutional networks, in *CVPR 2017*. Cited on page 13.

W.-C. Hung, Y.-H. Tsai, Y.-T. Liou, Y.-Y. Lin, and M.-H. Yang (2018). Adversarial Learning for Semi-Supervised Semantic Segmentation, in *BMVC 2018*. Cited on page 24.

P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros (2017). Image-to-Image Translation with Conditional Adversarial Networks, in *CVPR 2017*. Cited on pages 2, 24, 70, 75, 76, 82, 83, 84, 108, 126, and 133.

M. Jaderberg, K. Simonyan, A. Zisserman, *et al.* (2015). Spatial transformer networks, in *Advances in Neural Information Processing Systems (NIPS) 2015*. Cited on page 54.

Y. Jeon and J. Kim (2017). Active convolution: Learning the shape of convolution for image classification, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2017*. Cited on page 64.

X. Jia, B. De Brabandere, T. Tuytelaars, and L. V. Gool (2016). Dynamic filter networks, in *Advances in Neural Information Processing Systems 2016*. Cited on page 17.

Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell (2014). Caffe: Convolutional architecture for fast feature embedding, in *ACM Multimedia 2014*. Cited on pages 37, 58, and 75.

J. Johnson, A. Alahi, and L. Fei-Fei (2016). Perceptual losses for real-time style transfer and super-resolution, in *ECCV 2016*. Cited on pages 6, 23, 26, and 73.

T. Karras, T. Aila, S. Laine, and J. Lehtinen (2017). Progressive growing of gans for improved quality, stability, and variation, *arXiv preprint arXiv:1710.10196*. Cited on page 23.

A. Kendall, B. Vijay, and R. Cipolla (2015). Bayesian SegNet: Model Uncertainty in Deep Convolutional Encoder-Decoder Architectures for Scene Understanding, *arXiv preprint arXiv:1511.02680*. Cited on pages 38, 39, 40, 41, and 137.

D. Kingma and J. Ba (2015). Adam: A Method for Stochastic Optimization, *The Int. Conf. on Learning Representations*. Cited on page 75.

D. P. Kingma and M. Welling (2013). Auto-encoding variational bayes, *arXiv preprint arXiv:1312.6114*. Cited on pages 2, 6, 23, 25, 26, and 131.

P. Krähenbühl and V. Koltun (2011). Efficient inference in fully connected crfs with gaussian edge potentials, in *Advances in neural information processing systems 2011*. Cited on pages 1 and 13.

A. Krizhevsky, I. Sutskever, and G. E. Hinton (2012). Imagenet classification with deep convolutional neural networks, in *NIPS 2012*. Cited on pages iii, v, 1, and 13.

A. Kundu, V. Vineet, and V. Koltun (2016). Feature space optimization for semantic video segmentation, in *CVPR 2016*. Cited on page 63.

S. Kwak, S. Hong, and B. Han (2017). Weakly supervised semantic segmentation using superpixel pooling network, in *AAAI 2017*. Cited on page 15.

J. Lafferty, A. McCallum, and F. C. Pereira (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. Cited on page 13.

E. Learned-Miller, G. B. Huang, A. RoyChowdhury, H. Li, and G. Hua (2016). Labeled faces in the wild: A database for studying face recognition in unconstrained environments, *Advances in Face Detection and Facial Image Analysis*. Cited on page 75.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner (1998). Gradient-based Learning Applied to Document Recognition, *Proc. of the IEEE*, vol. 86(11), pp. 2278–2324. Cited on pages 1, 11, and 15.

C.-Y. Lee, P. W. Gallagher, and Z. Tu (2016). Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree, in *Artificial Intelligence and Statistics 2016*. Cited on page 17.

C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu (2015). Deeply-supervised nets, in *Artificial Intelligence and Statistics 2015*. Cited on page 18.

D. Lin, G. Chen, D. Cohen-Or, P.-A. Heng, and H. Huang (2017). Cascaded feature network for semantic segmentation of RGB-D images, in *ICCV 2017*. Cited on page 15.

T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick (2014). Microsoft coco: Common objects in context, in *ECCV 2014*. Cited on pages 1 and 57.

C. Liu, L.-C. Chen, F. Schroff, H. Adam, W. Hua, A. L. Yuille, and L. Fei-Fei (2019). Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2019*. Cited on page 64.

W. Liu, A. Rabinovich, and A. C. Berg (2015). Parsenet: Looking wider to see better, *arXiv preprint arXiv:1506.04579*. Cited on pages 1 and 17.

J. Long, E. Shelhamer, and T. Darrell (2015). Fully Convolutional Networks for Semantic Segmentation, in *CVPR 2015*. Cited on pages iii, v, 1, 12, 17, 36, 38, 39, 41, 42, 47, 48, 58, 123, 132, and 137.

P. Luc, C. Couprie, S. Chintala, and J. Verbeek (2016). Semantic Segmentation using Adversarial Networks, in *NIPS Workshop on Adversarial Training 2016*. Cited on page 24.

L. Ma, X. Jia, Q. Sun, B. Schiele, T. Tuytelaars, and L. Van Gool (2017). Pose guided person image generation, in *Advances in Neural Information Processing Systems 2017*. Cited on page 85.

K.-K. Maninis, J. Pont-Tuset, P. Arbeláez, and L. Van Gool (2016). Convolutional oriented boundaries, in *ECCV 2016*. Cited on page 14.

X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley (2017). Least squares generative adversarial networks, in *ICCV 2017*. Cited on page 23.

J. McCormac, A. Handa, A. Davison, and S. Leutenegger (2016). SemanticFusion: Dense 3D Semantic Mapping with Convolutional Neural Networks, *arXiv preprint arXiv:1609.05130*. Cited on pages 38 and 44.

B. H. Menze, A. Jakab, S. Bauer, J. Kalpathy-Cramer, K. Farahani, J. Kirby, Y. Burren, N. Porz, J. Slotboom, R. Wiest, *et al.* (2015). The multimodal brain tumor image segmentation benchmark (BRATS), *IEEE transactions on medical imaging*, vol. 34(10), pp. 1993–2024. Cited on page 1.

M. Mirza and S. Osindero (2014). Conditional generative adversarial nets, *arXiv preprint arXiv:1411.1784*. Cited on pages 24 and 70.

S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard (2017). Universal adversarial perturbations, in *CVPR 2017*. Cited on page 28.

S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard (2016). Deepfool: a simple and accurate method to fool deep neural networks, in *CVPR 2016*. Cited on page 28.

F. Mueller, F. Bernard, O. Sotnychenko, D. Mehta, S. Sridhar, D. Casas, and C. Theobalt (2018). Ganerated hands for real-time 3d hand tracking from monocular RGB, in *CVPR 2018*. Cited on pages 24, 25, and 87.

M. Nasr, R. Shokri, and A. Houmansadr (2018). Machine learning with membership privacy using adversarial regularization, in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security 2018*. Cited on page 28.

P. K. Nathan Silberman, Derek Hoiem and R. Fergus (2012). Indoor Segmentation and Support Inference from RGBD Images, in *ECCV 2012*. Cited on pages 1, 18, 19, and 38.

P. Neubert and P. Protzel (2012). Superpixel benchmark and comparison, in *Proc. Forum Bildverarbeitung 2012*. Cited on page 14.

G. Neuhold, T. Ollmann, S. Rota Bulo, and P. Kontschieder (2017). The mapillary vistas dataset for semantic understanding of street scenes, in *ICCV 2017*. Cited on pages 21, 103, and 109.

H. Noh, S. Hong, and B. Han (2015). Learning Deconvolution Network for Semantic Segmentation, in *ICCV 2015*. Cited on page 12.

A. Odena, C. Olah, and J. Shlens (2017). Conditional Image Synthesis with Auxiliary Classifier GANs, in *ICML 2017*. Cited on page 24.

S. J. Oh, M. Augustin, B. Schiele, and M. Fritz (2018). Towards reverse-engineering black-box neural networks, in *ICLR 2018*. Cited on page 28.

S. J. Oh, M. Fritz, and B. Schiele (2017). Adversarial image perturbation for privacy protection–a game theory perspective, in *ICCV 2017*. Cited on pages 28, 83, 84, and 138.

T. Orekondy, B. Schiele, and M. Fritz (2019). Knockoff Nets: Stealing Functionality of Black-Box Models, in *CVPR 2019*. Cited on page 28.

T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu (2019). Semantic image synthesis with spatially-adaptive normalization, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2019*. Cited on page 27.

O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. Jawahar (2012). Cats and dogs, in *CVPR 2012*. Cited on page 75.

C. Peng, X. Zhang, G. Yu, G. Luo, and J. Sun (2017). Large Kernel Matters–Improve Semantic Segmentation by Global Convolutional Network, in *CVPR 2017*. Cited on page 17.

X. Peng, Z. Tang, F. Yang, R. S. Feris, and D. Metaxas (2018). Jointly optimize data augmentation and network training: Adversarial data augmentation in human pose estimation, in *CVPR 2018*. Cited on pages 24 and 25.

J. Pont-Tuset, P. Arbelaez, J. T. Barron, F. Marques, and J. Malik (2017). Multiscale combinatorial grouping for image segmentation and object proposal generation, *IEEE transactions on pattern analysis and machine intelligence*, vol. 39(1), pp. 128–140. Cited on pages 5 and 14.

A. Pyrgelis, C. Troncoso, and E. De Cristofaro (2018). Knock knock, who's there? Membership inference on aggregate location data, *NDSS*. Cited on page 27.

X. Qi, Q. Chen, J. Jia, and V. Koltun (2018). Semi-parametric Image Synthesis, in *CVPR 2018*. Cited on pages 27, 87, and 93.

P. Ramachandran, B. Zoph, and Q. V. Le (2017). Searching for activation functions, *arXiv preprint arXiv:1710.05941*. Cited on page 129.

S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee (2016). Generative adversarial text to image synthesis, in *ICML 2016*. Cited on page 24.

S. Ren, K. He, R. Girshick, and J. Sun (2015). Faster r-cnn: Towards real-time object detection with region proposal networks, in *Advances in neural information processing systems 2015*. Cited on page 12.

X. Ren, L. Bo, and D. Fox (2012). Rgb-(d) scene labeling: Features and algorithms, in *CVPR 2012*. Cited on page 14.

J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid (2015). EpicFlow: Edge-preserving interpolation of correspondences for optical flow, in *CVPR 2015*. Cited on pages 35 and 37.

S. R. Richter, V. Vineet, S. Roth, and V. Koltun (2016). Playing for data: Ground truth from computer games, in *ECCV 2016*. Cited on pages 2 and 63.

O. Ronneberger, P. Fischer, and T. Brox (2015). U-net: Convolutional networks for biomedical image segmentation, in *MICCAI 2015*. Cited on page 91.

G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez (2016). The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes, in *CVPR 2016*. Cited on page 2.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams (1988). Learning representations by back-propagating errors, *Cognitive modeling*, vol. 5(3), p. 1. Cited on pages 1, 36, 56, 73, and 125.

F. S. Saleh, M. S. Aliakbarian, M. Salzmann, L. Petersson, and J. M. Alvarez (2018). Effective use of synthetic data for urban scene semantic segmentation, in *ECCV 2018*. Cited on page 87.

A. Salem, Y. Zhang, M. Humbert, M. Fritz, and M. Backes (2019). Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models, in *NDSS 2019*. Cited on pages 7, 27, and 104.

S. Sankaranarayanan, Y. Balaji, A. Jain, S. N. Lim, and R. Chellappa (2018). Learning from synthetic data: Addressing domain shift for semantic segmentation, in *CVPR 2018*. Cited on page 87.

M. Schwarz, A. Milan, A. S. Periyasamy, and S. Behnke (2018). RGB-D object detection and semantic segmentation for autonomous manipulation in clutter, *The International Journal of Robotics Research*, vol. 37(4-5), pp. 437–451. Cited on page 1.

R. Shokri and V. Shmatikov (2015). Privacy-preserving deep learning, in *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security 2015*. Cited on page 28.

R. Shokri, M. Stronati, C. Song, and V. Shmatikov (2017). Membership inference attacks against machine learning models, in *IEEE Symposium on Security and Privacy (SP) 2017*. Cited on pages 3, 7, 27, 104, 124, and 127.

A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb (2017). Learning from Simulated and Unsupervised Images through Adversarial Training, in *CVPR 2017*. Cited on pages 24, 25, and 87.

N. Silberman, D. Hoiem, P. Kohli, and R. Fergus (2012). Indoor segmentation and support inference from RGBD images, in *ECCV 2012*. Cited on page 1.

K. Simonyan and A. Zisserman (2014). Very deep convolutional networks for large-scale image recognition, *arXiv preprint arXiv:1409.1556*. Cited on pages 13 and 57.

L. Sixt, B. Wild, and T. Landgraf (2016). Rendergan: Generating realistic labeled data, *arXiv preprint arXiv:1611.01331*. Cited on pages 24 and 25.

K. Sohn, H. Lee, and X. Yan (2015). Learning structured output representation using deep conditional generative models, in *NIPS 2015*. Cited on pages 26 and 69.

N. Souly, C. Spampinato, and M. Shah (2017). Semi Supervised Semantic Segmentation Using Generative Adversarial Network, in *2017 IEEE International Conference on Computer Vision (ICCV) 2017*. Cited on page 24.

J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller (2014). Striving for simplicity: The all convolutional net, *arXiv preprint arXiv:1412.6806*. Cited on page 17.

N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov (2014). Dropout: a simple way to prevent neural networks from overfitting., *JMLR*. Cited on pages 71 and 81.

J. Strassburg, R. Grzeszick, L. Rothacker, and G. A. Fink (2015). On the Influence of Superpixel Methods for Image Parsing., in *VISAPP 2015*. Cited on page 14.

J. Stückler, B. Waldvogel, H. Schulz, and S. Behnke (2015). Dense real-time mapping of object-class semantics from RGB-D video, *Journal of Real-Time Image Processing*, vol. 10(4). Cited on pages 38 and 44.

P. Sturgess, K. Alahari, L. Ladicky, and P. H. Torr (2009). Combining appearance and structure from motion features for road scene understanding, in *BMVC 2009*. Cited on page 61.

D. Stutz, A. Hermans, and B. Leibe (2018). Superpixels: An evaluation of the state-of-the-art, *Computer Vision and Image Understanding*, vol. 166, pp. 1–27. Cited on pages 5 and 14.

C. Sun, A. Shrivastava, S. Singh, and A. Gupta (2017). Revisiting unreasonable effectiveness of data in deep learning era, in *ICCV 2017*. Cited on pages 1 and 103.

C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich (2015). Going deeper with convolutions, in *CVPR 2015*. Cited on page 13.

C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna (2016). Rethinking the inception architecture for computer vision, in *CVPR 2016*. Cited on pages 18, 91, 111, and 138.

J. Tighe and S. Lazebnik (2010). Superparsing: scalable nonparametric image parsing with superpixels, in *ECCV 2010*. Cited on page 14.

Y.-H. Tsai, W.-C. Hung, S. Schulter, K. Sohn, M.-H. Yang, and M. Chandraker (2018). Learning to adapt structured output space for semantic segmentation, *arXiv preprint arXiv:1802.10349*. Cited on page 87.

A. Wang, J. Lu, W. Gang, J. Cai, and T.-J. Cham (2014). Multi-modal Unsupervised Feature Learning for RGB-D Scene Labeling, in *ECCV 2014*. Cited on pages 38 and 44.

J. Wang, Z. Wang, D. Tao, S. See, and G. Wang (2016). Learning Common and Specific Features for RGB-D Semantic Segmentation with Deconvolutional Networks, in *ECCV 2016*. Cited on pages 38 and 44.

P. Wang, P. Chen, Y. Yuan, D. Liu, Z. Huang, X. Hou, and G. Cottrell (2018a). Understanding convolution for semantic segmentation, in *WACV 2018*. Cited on page 16.

T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro (2018b). High-resolution image synthesis and semantic manipulation with conditional gans, in *CVPR 2018*. Cited on pages 2, 27, 87, and 93.

X. Wang, D. Fouhey, and A. Gupta (2015). Designing deep networks for surface normal estimation, in *CVPR 2015*. Cited on page 76.

Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli (2004). Image quality assessment: from error visibility to structural similarity, *TIP*. Cited on pages 76 and 83.

Y. Wen, K. Zhang, Z. Li, and Y. Qiao (2016). A discriminative feature learning approach for deep face recognition, in *ECCV 2016*. Cited on pages 83, 84, and 138.

M. Wrenninge and J. Unger (2018). Synscapes: A Photorealistic Synthetic Dataset for Street Scene Parsing, *arXiv preprint arXiv:1810.08705*. Cited on page 98.

Z. Wu, X. Han, Y.-L. Lin, M. G. Uzunbas, T. Goldstein, S. N. Lim, and L. S. Davis (2018). DCAN: Dual Channel-wise Alignment Networks for Unsupervised Scene Adaptation, in *ECCV 2018*. Cited on page 87.

Y. Xian, T. Lorenz, B. Schiele, and Z. Akata (2018). Feature Generating Networks for Zero-Shot Learning, in *CVPR 2018*. Cited on page 24.

J. Xiao, A. Owens, and A. Torralba (2013). SUN3D: A Database of Big Spaces Reconstructed using SfM and Object Labels, in *ICCV 2013*. Cited on pages 20 and 38.

C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie, and A. Yuille (2017). Adversarial examples for semantic segmentation and object detection, in *ICCV 2017*. Cited on page 28.

F. Yu and V. Koltun (2016). Multi-Scale Context Aggregation by Dilated Convolutions, in *ICLR 2016*. Cited on pages iii, v, 1, 5, 15, 53, 54, 63, 64, and 125.

F. Yu, V. Koltun, and T. Funkhouser (2017). Dilated residual networks, in *CVPR 2017*. Cited on page 16.

F. Yu, W. Xian, Y. Chen, F. Liu, M. Liao, V. Madhavan, and T. Darrell (2018). BDD100K: A diverse driving video database with scalable annotation tooling, *arXiv preprint arXiv:1805.04687*. Cited on pages 1, 20, 103, and 109.

H. Zhang, K. Dana, J. Shi, Z. Zhang, X. Wang, A. Tyagi, and A. Agrawal (2018a). Context encoding for semantic segmentation, in *CVPR 2018*. Cited on pages 1, 18, and 128.

H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. N. Metaxas (2017a). Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks, in *ICCV 2017*. Cited on page 23.

K. Zhang, Z. Zhang, Z. Li, and Y. Qiao (2016). Joint face detection and alignment using multitask cascaded convolutional networks, *IEEE Signal Processing Letters*. Cited on page 75.

L. Zhang, L. Zhang, X. Mou, and D. Zhang (2011). FSIM: A feature similarity index for image quality assessment, *TIP*. Cited on pages 76 and 83.

R. Zhang, S. Tang, Y. Zhang, J. Li, and S. Yan (2017b). Scale-adaptive convolutions for scene parsing, in *ICCV 2017*. Cited on pages 64 and 128.

T. Zhang, Z. He, and R. B. Lee (2018b). Privacy-preserving machine learning through data obfuscation, *arXiv preprint arXiv:1807.01860*. Cited on page 28.

X. Zhang, X. Zhou, M. Lin, and J. Sun (2018c). Shufflenet: An extremely efficient convolutional neural network for mobile devices, in *CVPR 2018*. Cited on page 13.

Z. Zhang, P. Luo, C. C. Loy, and X. Tang (2014). Facial landmark detection by deep multi-task learning, in *ECCV 2014*. Cited on pages 75 and 83.

H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia (2017). Pyramid Scene Parsing Network, *CVPR*. Cited on pages 1, 17, 18, 53, 57, 58, 61, 62, 63, 92, 93, 100, 111, 128, and 133.

S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. Torr (2015). Conditional Random Fields as Recurrent Neural Networks, in *ICCV 2015*. Cited on pages 38, 39, 40, 41, 47, 48, 132, and 137.

Z. Zheng, L. Zheng, and Y. Yang (2017). Unlabeled Samples Generated by GAN Improve the Person Re-Identification Baseline in Vitro, in *ICCV 2017*. Cited on pages 24 and 25.

B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba (2015). Object detectors emerge in deep scene cnns, in *ICLR 2015*. Cited on page 72.

B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba (2016). Learning deep features for discriminative localization, in *CVPR 2016*. Cited on pages 109, 114, 116, and 135.

B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba (2018). Places: A 10 million image database for scene recognition, *IEEE transactions on pattern analysis and machine intelligence*, vol. 40(6), pp. 1452–1464. Cited on pages 1 and 11.

B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba (2017). Scene parsing through ade20k dataset, in *CVPR 2017*. Cited on pages 1, 22, and 92.

J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros (2017a). Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks, in *ICCV 2017*. Cited on page 24.

J.-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, and E. Shechtman (2017b). Toward Multimodal Image-to-Image Translation, in *NIPS 2017*. Cited on pages 24, 26, 70, 75, 76, 82, 83, 84, 89, 90, 92, 93, 94, 95, 108, 126, and 133.

X. Zhu, H. Hu, S. Lin, and J. Dai (2019). Deformable convnets v2: More deformable, better results, in *CVPR 2019*. Cited on page 17.

B. Zoph and Q. V. Le (2016). Neural architecture search with reinforcement learning, *arXiv preprint arXiv:1611.01578*. Cited on page 124.

B. Zoph and Q. V. Le (2017). Neural architecture search with reinforcement learning, in *ICLR 2017*. Cited on pages 13 and 129.

B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le (2018). Learning transferable architectures for scalable image recognition, in *CVPR 2018*. Cited on page 13.