Pixel-level Scene Understanding with Deep Structured Models

Anurag Arnab
Linacre College
University of Oxford

A thesis submitted for the degree of
Doctor of Philosophy

Trinity 2019

Declaration

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

Anurag Arnab, Linacre College

Acknowledgements

I would like to thank my supervisor, Prof. Phil Torr, for having faith in me and providing the opportunity to pursue a PhD. His infectious energy and creativity have been pivotal in making this thesis possible.

I have also had the pleasure of working closely with many collaborators during my PhD. My mentors in my initial projects – Michael, Ondra, Stuart and Sadeep – have played a significant role in my development as a researcher. I have also enjoyed subsequent collaborations with Kyle, Qizhu, Måns, Fredrik, Li, Rodrigo and Carl. Further thanks go to the computer vision teams in DeepMind and Google (particularly Abhanshu, Matt and Sammy) for hosting me. Bernardino and Stuart have also taken the time to provide detailed and thoughtful feedback on drafts of my papers on numerous occasions.

I have been fortunate to be part of the computer vision community in Oxford. There are too many past and present members in TVG to name here, along with many others from VGG, OVAL and AVL from whom I have learned things from whilst sharing an "office". One of the best perks of studying at Oxford has been the impressive array of seminar speakers who have given talks here and then been available for further discussions afterwards. I am also grateful to my transfer report examiners, Professors Pawan Kumar and Paul Newman, and also Prof. Andrew Zissermann whom I collaborated with. My thesis examiners, Professors Andrea Vedaldi and Iasonas Kokkinos, also provided insightful comments on the manuscript and the viva itself turned out to be a stimulating discussion.

I have thoroughly enjoyed my time here in Oxford, both inside and outside of the lab, thanks to Sven, Aravindh, Namhoon, Jessamy, Qizhu, Li, Arnab, Harkirat, Daniela, Oscar, Piotr, Vivek and the badminton clubs of Linacre and Jesus College.

Maddy, and then later Cassandra, have also been helpful regarding all administrative matters. And thanks to Jerry, as well as the ARC team, for managing the group's computing resources.

I am grateful to my family for their continual support during my PhD. In particular, I am thankful to Didi for proof-reading yet another thesis and spoiling me on every vacation.

Finally, I am grateful to have been funded by the Clarendon Scholarship.

Abstract

Although humans can effortlessly recognise a scene in its totality, it is an extremely challenging problem for computers which is why scene understanding remains one of the fundamental problems in computer vision. This thesis concentrates on pixel-level scene understanding tasks such as semantic- and instance-segmentation, which have applications in diverse fields such as autonomous vehicles, medical diagnosis and assistive technologies for the partially sighted among others.

Firstly, this thesis addresses the task of semantic segmentation by integrating mean-field inference of a Conditional Random Field (CRF) with higher order potentials directly into a deep neural network. This approach enables joint, end-to-end training of both the parameters of the CRF and the underlying CNN, and achieved state-of-the-art results on public leaderboards at the time of publication.

This method is then extended to the task of instance segmentation. In contrast to previous work, the proposed formulation jointly processes all instances in the image. As such, one pixel can only be assigned to one instance and the network must thus learn to reason about occlusions between instances. Moreover, unlike previous work, this approach can naturally segment "stuff" classes. This method also achieved state-of-the-art results at the time of publication.

Realising the fact that pixel-level training data for segmentation is time-consuming and thus expensive to obtain, this thesis then proposes a method of training semantic- and instance-segmentation models with weaker supervision. In particular, annotations in the form of bounding-boxes and image-level tags are considered, which are shown to significantly reduce annotation time with a relatively small impact on the final performance compared to a fully-supervised baseline.

Finally, this thesis studies the adversarial robustness of popular semantic segmentation architectures. This topic is motivated by the fact that during the course of this thesis, segmentation systems have become accurate enough to use in real-world applications, and thus the security of models deployed in production is critical. The effect of various architectural components on adversarial robustness are thoroughly evaluated, and mean-field inference of CRFs, multiscale processing (and more generally, input transformation) are shown to naturally implement concurrently proposed adversarial defences.

Contents

List of Figures

List of Tables

Chapter 1

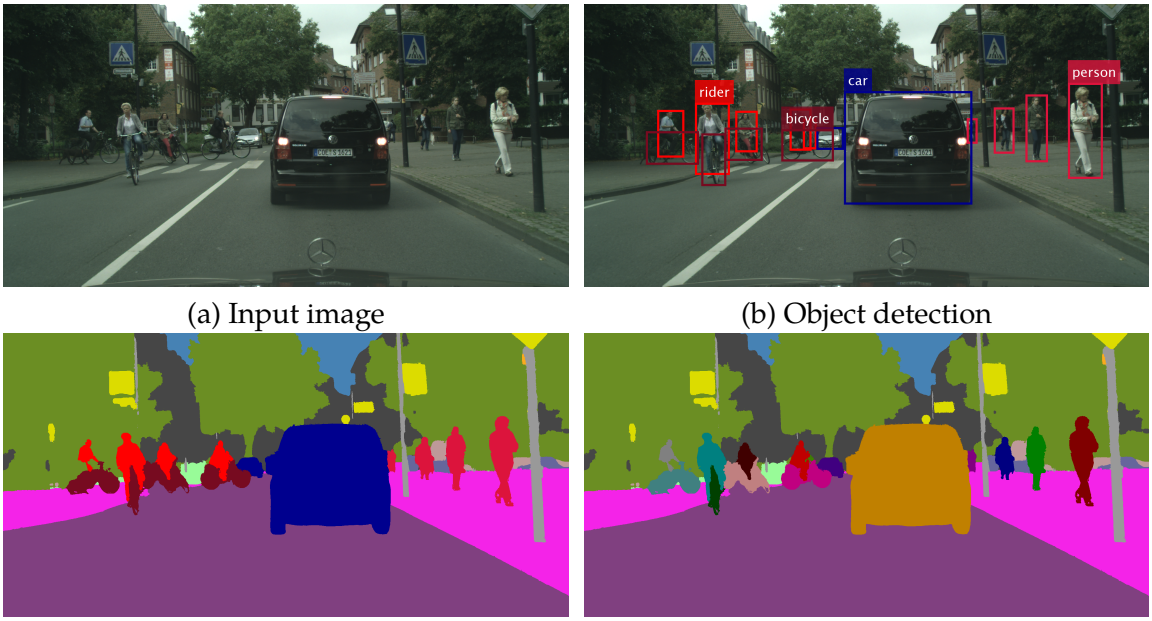
Introduction

1.1 Scene Understanding in Computer Vision

Scene understanding is one of the fundamental problems in computer vision, and aims to bridge the gap between how computers store visual information and how humans comprehend it. Computers store images digitally as pixels, each with specific colour intensity values. Humans, on the other hand, do not interpret images as a large matrix of numbers, but rather extract relevant details from the image. Examples include the type and number of objects present in the image, the affordances of the different objects, the relationships between different objects in the scene, and forecasting how a scene will change in the near future.

As shown in Figure 1.1, scene understanding has typically been studied as multiple different tasks in the computer vision literature. A high-level summary of a scene can be obtaining by predicting image tags that describe the objects present in the image (such as "person" and "car") or the scene (such as "city"). This task is known as image classification. The object detection task, on the other hand, aims to localise different objects in an image by placing bounding boxes around each instance of a predefined object category (Fig. 1.1b).

This thesis concentrates on pixel-level scene understanding problems such as semantic- and instance segmentation. Semantic segmentation (Fig. 1.1c) aims for a more precise understanding of the scene by assigning an object category label to each pixel within the image. Instance segmentation extends this by also assigning a unique identifier to each of the segmented objects in the image (Fig. 1.1d). It can thus be regarded as being at the intersection of object detection (which localises different instances of an object, but at a coarse bounding-box level) and semantic segmentation (which has no notion of instance of the same object, but classifies individual pixels). These segmentation tasks are motivated by their applications:



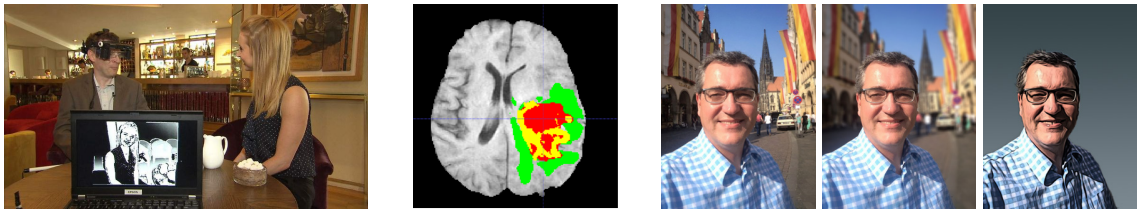
(a) Input image (b) Object detection

(c) Semantic segmentation (d) Instance segmentation

Figure 1.1: Example of the scene understanding tasks, and their application to perception for autonomous driving. This thesis focusses on semantic- and instance segmentation. Semantic segmentation (c) labels every pixel in the image with an object class label. Instance segmentation (d) extends this by identifying unique instances of an object class as well. It is thus at the intersection of object detection (b) (which localises different instances of an object, but at a coarse bounding-box level) and semantic segmentation (which operates at a pixel-level, but has no notion of different instances of the same class). The results shown here have been produced by the algorithms described in this thesis.

Autonomous vehicles Perception is a key component of self-driving cars, as they need to understand their environment before planning the route ahead. Examples of this are shown in Fig. 1.1. High accuracy for these perception algorithms are also of paramount importance due to the safety-critical nature of this task. Self-driving cars also have the potential to greatly reduce fatalities due to car accidents, and to decrease traffic and the industry's ecological footprint by reducing the number of vehicles required to meet an area's transportation requirements.

Medical diagnosis Computer vision algorithms can provide cost-effective solutions to diagnose a wide range of medical conditions [79, 70, 268, 303, 199]. Automated algorithms can also relieve specialist doctors from having to perform time-consuming annotations of medical imagery, and can expand access to healthcare by providing diagnoses in areas where specialist doctors are not available.



(a) Smart-glasses for the partially sighted

(b) Medical imaging

(c) Photographic effects

Figure 1.2: Applications of segmentation: a) "Smart glasses" enhance the vision of the partially sighted [124, 207]. b) The output of brain tumour segmentation algorithm from MRI images [211]. c) Effects such as depth-of-field and stylisation can be automatically applied once the person is segmented from the photo [262].

Image- and video-editing Accurately segmenting out objects from an image or video is used for editing operations such as compositing, rig-removal and automatic depth-of-field or bokeh effects (as shown in Fig. 1.2). These applications are ubiquitous in the digital- and print-media industries.

Augmented Reality In augmented reality, objects and scenes in the real-world are modified with computer-generated information. As a result, such systems (an example being augmented reality glasses) need a detailed and pixel-level understanding of the physical world around them so that the digitally modified environment appears realistic.

Assistive technologies for the partially sighted There have been recent developments in designing "smart glasses" for the blind and partially sighted [124, 292], as there are more than 285 million people in the world with vision impairments that affect day-to-day living [218]. These glasses understand the environment around them and stimulate the user's residual vision (as shown in Fig. 1.2), as about 85% of these people have some remaining vision [218].

Note that though this thesis concentrates on semantic- and instance segmentation, many other scene understanding problems are actively studied in the computer vision literature, such as 2D- and 3D pose estimation, 3D reconstruction, depth estimation, scene flow estimation, scene graph parsing, tracking and activity forecasting [95, 299, 4, 145, 156].

1.2 Challenges in pixel-level scene understanding

Whilst humans can effortlessly recognise everything in a scene, it is extremely challenging for machines. This fact is reflected by performance of leading methods on various benchmarks.

Figure 1.3: Object variability in the real world makes recognition difficult: Examples of "chair" images from the Pascal VOC [81] dataset are shown, along with their segmentation masks in the next row (red indicates "chair", orange "table" and pink "person"; grey pixels are "ambiguous"). Note the intra-class variation between chair instances in each image. Furthermore, there are significant illumination and viewpoint changes across the images, and there are also objects occluding the chairs. Chairs also contain thin and elongated structures which are difficult to segment accurately.

For example, the state-of-the-art method on the Cityscapes instance segmentation benchmark [57] still only achieves a mean Average Precision (mAP) of less than 40 (the original baseline approach achieved 4.6). As a result, we first discuss why scene understanding (and also semantic- and instance segmentation in particular) are so difficult before describing the overall approach of this thesis in the next section.

1.2.1 Object variability

Visual recognition systems need to generalise across large variations in the appearance of an object due to physical, geometric and photometric factors such as viewpoint, occlusions, illumination, blur and sensor noise, as shown in Fig. 1.3. Moreover, considerable intra-class variation needs to be accounted for as well. Figure 1.3 shows several examples of "chairs" in the Pascal VOC dataset [81] that all vary considerably among each other in visual appearance. These significant differences in appearance must still be abstracted away by an object recognition system in order to classify all inputs correctly.

Furthermore, as shown in Fig. 1.3, objects may also be heavily occluded, or be observed at a small scale, such that the most salient landmarks of an object may not be visible at all. Nevertheless, the aim of semantic- and instance-segmentation is still to classify each pixel constituting the object correctly (potentially by exploiting the context around the occluded object). Small and/or occluded objects are thus typically difficult to recognise, and this is observed on benchmarks as well. On the COCO object detection challenge [180], the performance measured in terms of the mAP for objects considered "small" (measured in

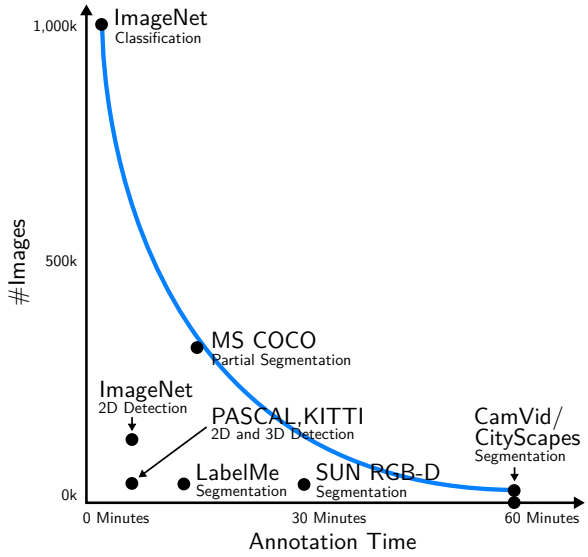
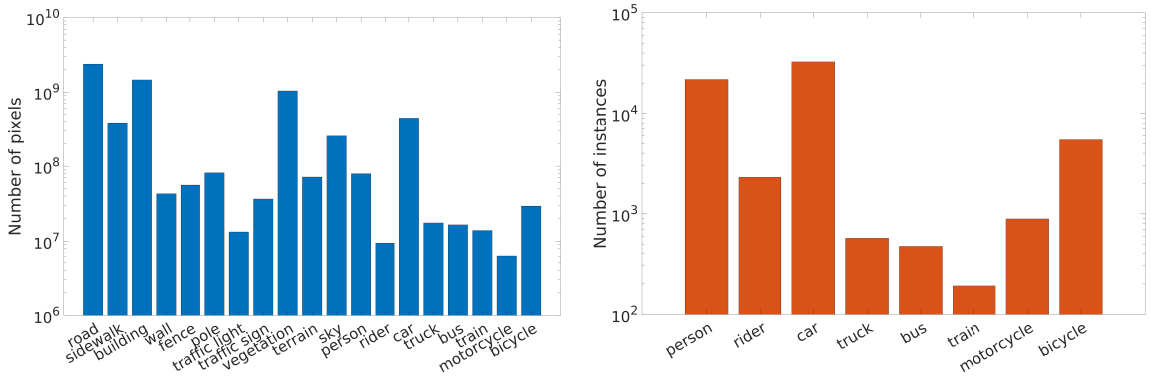


Figure 1.4: The "curse of dataset annotation" [310]. Deep neural networks require a lot of training data. However, annotations for complex tasks such as segmentation are time-consuming and hence expensive to collect, which reduces the sizes of datasets for these tasks. Figure from [310].

terms of number of pixels they occupy in the image) is about half that for objects considered "large" [99]. Similarly, on the Cityscapes benchmark [57], the average Intersection over Union (IoU) computed globally for all pixels of a given object class in the dataset is higher than the IoU averaged over individual instances by about 30%. This also shows that current state-of-the-art methods perform better on larger objects than on smaller objects.

1.2.2 Datasets and dataset bias

Large labelled datasets like ImageNet [71, 254] have driven the emergence of deep neural networks as the de facto tool for classification tasks in computer vision. However, the ability of deep neural networks to learn powerful feature representations automatically from data also means that deep learning approaches to various computer vision problems all require suitable datasets for training. This dependence of neural networks on large, labelled datasets has been termed by Xie *et al*. [310] (among others) as the "Curse of dataset annotation" due to the time, and thus cost, incurred in creating labelled datasets. This issue is exacerbated in the case of segmentation where every single pixel in the image needs to be labelled, and is illustrated by the Cityscapes dataset where each image took 90 minutes to label [57]. The annotation cost also meant that only 5000 out of the 25000 images in the dataset were fully annotated. As shown in Fig. 1.4, tasks that require more complex annotations, such as segmentation, typically have smaller datasets due to the annotation costs involved.



(a) Number of pixels per class (b) Number of instances per class

Figure 1.5: Statistics for the Cityscapes dataset [57]. The number of pixels, (a), are shown for all ("thing" and "stuff" [91]) classes. The number of instances (b), are only shown for "thing" classes. The axes are on a logarithmic scale, due to the imbalance in the classes.

Additionally, for segmentation tasks, annotations along the boundaries of the objects typically have more errors than within the interior as these boundaries are more difficult for annotators to label correctly. Furthermore, as there are substantially fewer pixels at the boundaries of the object than the interior, it is difficult to learn models that are accurate at boundaries of the object.

Moreover, datasets are imbalanced or biased in other ways [283]. Common segmentation datasets like Cityscapes [57] and Pascal VOC [81] have a long-tailed distribution of objects (Fig. 1.5) as some scenes and objects were more frequently observed when collecting the dataset. Figure 1.5 also shows how some "stuff" [91] classes such as "road" and "vegetation", which are the easiest to classify as they have little intra-class variation and similar texture, actually have the most labelled pixels. For other categories, the intra-class variability is sometimes large, and the training set does not capture all possible visual appearances of the class. This deficiency leads to models failing on unseen appearances in the test set or when deployed in the real world. For example, models trained on the Cityscapes driverless car dataset captured in Germany and Switzerland [57] perform significantly worse on road scenes from other parts of the world [49]. The performance degradation in [49] was the greatest for "thing" classes [91] with high intra-class variation or low number of training examples such as "motorcycle", "bicycle", "rider" and "person". Similarly, Zendel *et al*. [323] also noted how extreme weather conditions and other hazards significantly reduce accuracy of models trained on Cityscapes which does not have any weather variations.

(a) (b)

Figure 1.6: The importance of context in scene understanding. Parsing (a) is difficult for computer vision algorithms due to the similar appearance of the objects in the scene. However, humans have no such difficulty as they recognise it as an image of a bedroom, and have prior knowledge of the objects typically in a bedroom and its layout. This image is from the ADE20K dataset [331]. (b) Oliva and Torralba [217] noted the strong assumptions that observers made regarding object identities according to their size and location in the scene. Observers described the scene as a car and a pedestrian in the street. However, the "pedestrian" is actually the car that has been rotated by 90°.

1.2.3 Context and other priors for object recognition

Objects do not occur in isolation in natural scenes, but rather co-vary with other objects and environments. As shown in Fig. 1.6, humans have the ability to exploit contextual information at multiple levels [22, 53, 217], with examples including semantic co-occurence (table and chairs or bed and pillows which are usually present in the same scene), spatial configurations (a car is on the ground plane, and not in the air) and pose (cars are oriented along the driving direction of a street) [217]. These contextual priors help humans to focus on the salient parts of an image and quickly recognise objects in cluttered scenes [53]. However, exploiting contextual information, and capturing the real-world relationships between objects is challenging for computer vision algorithms. Furthermore, it may also be difficult to ensure that all relevant contextual relationships are present in the training dataset, making it difficult to learn automatically from data.

1.2.4 Differences in scene understanding tasks

Note that examples which are easy for semantic segmentation, may be difficult for instance segmentation. For example, Fig. 1.7 shows cases where there are multiple instances of the same object with very similar visual appearance and texture. Here, classifying each pixel with a predefined object class label is simpler than differentiating different instances of

(a) (b) (c)

Figure 1.7: Examples that are difficult for instance segmentation, but not semantic segmentation. (a) and (b) consist of many similar looking instances. Given an accurate appearance model, the class label of all the instances can be obtained for all constituent pixels due to the low intra-class variation. Conversely, the similar appearances of each of the instances, and the fact that they occlude each other, makes distinguishing the individual instances difficult. In (c), the man's arm around the woman's shoulder is difficult to associate with the right person as it is disconnected (visually) from the man's body.

the said class, which requires reasoning about the shape of the object class, and occlusion relationships between different instances.

1.3 Approach

To address the aforementioned challenges in pixel-level scene understanding, this thesis makes use of deep neural networks (DNNs) and Conditional Random Fields (CRFs), and shows how they can be combined to provide the benefits of both approaches.

Deep neural networks have become the de facto tool for classification tasks in computer vision as they are able to learn powerful feature representations automatically from data. This makes them suited to dealing with the large variability of real-world data as long as the training datasets are sufficiently large. However, structured prediction tasks such as segmentation involve predicting many random variables that are statistically related (in segmentation, each pixel can be associated with a random variable). Standard DNNs are not able take these complex dependencies between multiple output variables into account.

On the other hand, Conditional Random Fields (CRFs), and probabilistic graphical models in general, have long been used in computer vision for such structured prediction tasks. CRFs provide a principled way to model the dependencies between the different correlated variables being predicted, and thus to incorporate prior information about the task being solved. The priors that can be encoded with CRFs are useful for dealing with limited or biased data, explicitly encoding contextual priors and can also be used to incorporate other information that is obvious to humans but difficult to extract automatically from data.

A desirable feature of neural networks is that they are trained "end-to-end" – a single objective function is used to optimise all parameters in the differentiable network via

stochastic gradient descent (SGD) or its variants. In contrast to traditional computer vision algorithms, complex pre- or post-processing steps are usually not employed. This thesis integrates inference of CRFs directly into deep neural networks so that the entire model can still be trained end-to-end, thus obtaining the benefits of both DNNs and CRFs.

1.4 Thesis Outline

Chapter 2 Chapter 2 provides a background on DNNs and CRFs, which are used throughout the rest of this thesis.

Chapter 3 Chapter 3 then addresses the task of semantic segmentation (Fig. 1.1c) by integrating mean-field inference of a Conditional Random Field (CRF) with higher order potentials into a deep neural network.

Chapter 4 We then extend the method from the previous chapter to perform the task of instance segmentation (Fig. 1.1d) in Chapter 4.

Chapter 5 Both of the methods in the previous chapters are fully supervised, and require training data with per-pixel annotation which is very time-consuming, and thus expensive, to collect. To address this issue, Chapter 5 presents an approach of performing both semantic- and instance-segmentation with weaker supervision in the form of bounding boxes and image-level tags.

Chapter 6 During the course of this thesis, the performance of segmentation systems have greatly increased to the level that they are suitable for use in real-world applications. Consequently, the security of deep neural network models deployed in production becomes more crucial. Chapter 6 considers this issue by studying the adversarial robustness of common segmentation architectures to gain insight into how we can train models that are both accurate and robust to adversarial attacks (modified images with minimal perceptual differences to the original which cause a classifier to fail).

Chapter 7 Finally, we conclude in Chapter 7, by summarising the contributions in the thesis and discussing open questions, future directions and the impact that this work presented in this thesis has had on the field.

Note that this is an integrated thesis, and that each of Chapters 3 through 6 have been published at a leading computer vision conference or journal. The papers have only been reformatted, and the supplementary material has been included as an appendix at the end of the chapter. Each chapter is self-contained with its own related work section centred around the contribution of the paper. These contributions are now detailed next.

1.5 Contributions

Chapter 3: Higher Order Conditional Random Fields in Deep Neural Networks Chapter 3 proposes a Conditional Random Field (CRF) with higher order potentials for the task of semantic segmentation (Fig. 1.1c). Furthermore, it shows how mean-field inference of this CRF can be seen as a "layer" of a neural network. This is done by unrolling the iterative mean-field inference algorithm to form a recurrent network which can then be backpropagated through (previous work [329] had only shown this for a specific type of pairwise potential). If a mean-field inference "layer" is appended to a neural network, the parameters of the underlying neural network can always be optimised by backpropagation. And if the parameters of the CRF's potential functions are differentiable (as they are in this case), they can be learned jointly with the parameters of the underlying neural network. This method also achieved state-of-the-art results on two popular segmentation benchmarks at the time of publication.

Chapter 4: Pixelwise Instance Segmentation with a Dynamically Instantiated Network Chapter 4 extends the neural network from the previous chapter to perform the task of instance segmentation. Most prior art modified objection detection architectures to output segmentation masks instead of bounding boxes. However, these approaches all have a common set of limitations: they all process each instance independently of one another, meaning that one pixel can actually be assigned to multiple instances at the same time. Consequently, occlusions between instances are handled very poorly. Our proposed method, on the other hand, considers all instances jointly, and as each pixel can only belong to a single instance, the network must learn to reason about occlusions. Furthermore, the proposed method can handle a variable number of instances per image and requires no post-processing to produce the final result, unlike the previous detection-based approaches. Additionally, in contrast to detection-based approaches which are only suited for "thing" classes, the proposed formulation can deal naturally with both "thing" and "stuff" classes. Finally, the proposed method achieved state-of-the-art results on multiple instance segmentation datasets at the time of publication.

Chapter 5: Weakly- and Semi-Supervised Panoptic Segmentation Chapter 5 trains the model from the previous chapter with weaker supervision: Instead of using pixel-level segmentation masks as the ground truth, weaker annotations in the form of bounding boxes and image-level tags (which specify if an object is present or not in the image) are used as supervision. Furthermore, a combination of full supervision (pixel-level ground truth) and weak supervision (bounding boxes and image-level tags) can be readily used as well. To the best of our knowledge, this is the first work to train a model for non-overlapping instance segmentation without full supervision. The approach, based on the Expectation Maximisation (EM) algorithm, is demonstrated on multiple datasets, obtaining up to 95% of fully-supervised performance with the same data, and reducing the estimated annotation time by up to a factor of 35.

Chapter 6: On the Robustness of Semantic Segmentation Models to Adversarial Attacks This chapter presents what to our knowledge is the first rigorous evaluation of adversarial attacks on modern semantic segmentation models, using two large-scale datasets. It analyses the effect of different network architectures, model capacity and multiscale processing, and shows that many observations made on the task of classification do not always transfer to the more complex task of segmentation. Moreover, this chapter shows how mean-field inference of CRFs, multiscale processing (and more generally, input transformations) naturally implement recently proposed adversarial defences. However, in contrast to these prior works, this chapter also shows how these defences are ineffectual as soon as knowledge of them is used in the attack algorithm. This chapter will aid future efforts in understanding and defending against adversarial examples, whilst in the shorter term, shows how to effectively benchmark robustness and suggests which segmentation models should currently be preferred in safety-critical applications due to their inherent robustness.

1.6 Publications

The following publications form the individual chapters of this thesis:
Chapter 3

> Anurag Arnab, Sadeep Jayasumana, Shuai Zheng, Philip H.S Torr. Higher Order Conditional Random Fields in Deep Neural Networks. *European Conference on Computer Vision (ECCV)*, 2016.

Chapter 4

> Anurag Arnab, Philip H.S Torr. Pixelwise Instance Segmentation with a Dynamically Instantiated Network. *Computer Vision and Pattern Recognition (CVPR)*, 2017.

Chapter 5

Qizhu Li*, Anurag Arnab*, Philip H.S Torr. Weakly- and Semi-Supervised Panoptic Segmentation. *European Conference on Computer Vision (ECCV)*, 2018.
* Joint first authors

Chapter 6

Anurag Arnab, Ondrej Miksik, Philip H.S Torr. On the Robustness of Semantic Segmentation Models to Adversarial Attacks. *To appear in Pattern Analysis and Machine Intelligence (PAMI)*, 2019.
The version presented here has been accepted into PAMI, and is an extension of the CVPR 2018 conference paper with the same title and authors.

Publications in related topics were also made during this thesis:

Anurag Arnab*, Carl Doersch*, Andrew Zisserman. Exploiting temporal context for 3D human pose estimation in the wild. *Computer Vision and Pattern Recognition (CVPR)*, 2019.
* Joint first authors

Måns Larsson, Anurag Arnab, Shuai Zheng, Philip H.S Torr, Fredrik Kahl. Revisiting Deep Structured Models for Pixel-Level Labelling with Gradient-Based Inference. *SIAM Journal on Imaging Sciences*, 2018.

Anurag Arnab, Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Måns Larsson, Alexander Kirillov, Bogdan Savchynskyy, Carsten Rother, Fredrik Kahl, Philip Torr. Conditional Random Fields meet Deep Neural Networks for Semantic Segmentation. *IEEE Signal Processing Magazine*, 2018.
Material from this paper was used in Chapter 2 (Background).

Qizhu Li*, Anurag Arnab*, Philip H.S Torr. Holistic, Instance-level, Human Parsing. *British Machine Vision Conference (BMVC)*, 2017.
* Joint first authors

Måns Larsson, Anurag Arnab, Fredrik Kahl, Shuai Zheng, Philip H.S Torr. A Projected Gradient Descent Method for CRF Inference allowing End-To-End Training of Arbitrary Pairwise Potentials. *Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR)*, 2017.

Anurag Arnab, Philip H.S Torr. Bottom-up Instance Segmentation using Deep Higher Order CRFs. *British Machine Vision Conference (BMVC)*, 2016.

Chapter 2

Background

This chapter provides a brief review of the two primary concepts used in this thesis: Deep Neural Networks (DNNs) are discussed in Sec. 2.1, followed by a review of Conditional Random Fields (CRFs) in Sec. 2.2, which are use to address labelling problems in computer vision. The contributions of this thesis are detailed in Chapters 3 through 6, and each chapter contains a literature review directly related to its contribution.

2.1 Deep Neural Networks

Deep neural networks, and particularly convolutional neural networks (CNNs) in computer vision, have quickly become the standard tool for supervised learning following the success of Krizhevsky *et al*. [157] in the ImageNet image classification challenge in 2012. The neural network developed by [157], known as AlexNet, significantly outperformed all other entries based on traditional object recognition pipelines. These traditional methods can be broadly categorised into pipelines consisting of three separate steps: 1) Computing local feature descriptors (such as SIFT [191] and HOG [67]) from interest points [202, 203] detected in the image, 2) Aggregating these local descriptors into global descriptors using bag-of-visual word histograms [269, 60] or Fisher vectors [230] and 3) Classifying these global descriptors with support vector machines [58]. The parameters in all but the final classification step were tuned manually.

Deep neural networks, such as AlexNet, in constrast consist of a single model whose parameters are all optimised jointly and learned from data. Concretely, a neural network is a function f mapping data $\mathbf{x}$ (for example, an image) to an output $\mathbf{y}$ (for example, the label describing the image). The function $f = g_L \circ g_{L-1} \ldots \circ g_1$ is the composition of a sequence of simpler functions g_i which are known as layers [294]. If we denote $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_L$ as the outputs of each layer of the network, then each intermediate output $\mathbf{x}_i = g_i(\mathbf{x}_{i-1}; \mathbf{w}_i)$ is computed from the previous output $\mathbf{x}_{i-1}$ by applying the function g_i with parameters $\mathbf{w}_i$.

Note that $\mathbf{x}_0 = \mathbf{x}$ is the network input, and $\mathbf{x}_L = \mathbf{y}$ is the network output. The parameters of all layers of the network, $\mathbf{w}$, are learned by stochastic gradient descent (SGD) [246], or one of its variants [235, 140, 322], using backpropagation to compute the gradients of a scalar-valued loss function with respect to all parameters in the network. The intermediate outputs of the network, $\mathbf{x}_1, \ldots, \mathbf{x}_{L-1}$ can thus be thought of as the intermediate representations of the data learned by the network. In popular neural network architectures, the layer functions, g_i, consist of convolutional operations, elementwise non-linearities (such as rectified linear units (ReLUs) [212]) and pooling operations.

Although AlexNet [157] was a major breakthrough in the ImageNet image classification challenge, CNNs had been successfully applied to digit classification in 1989 [166]. Furthermore, the backpropagation algorithm for efficiently computing the gradients used by SGD was published in 1986 [253]. It has primarily been the emergence of large-scale datasets such as ImageNet [71, 254] and the parallel computational power of graphics processing units (GPUs) that have enabled neural networks to learn effective representations from data and become very successful in most machine learning tasks today. The primary algorithmic advances since the 1980's have included the ReLU non-linearity [212], regularisation methods such as dropout [271] and batch normalisation [130], improved initialisation methods [102, 118] and residual connections [117] which enable training networks with much greater depth and modelling capacity.

Network architectures such as AlexNet [157], VGG [267] and ResNet [117] have initially been designed for the image classification task and trained with the large ImageNet dataset. However, as detailed in the next sections, they can be adapted for more complex scene understanding tasks such as semantic segmentation (Sec. 2.1.1) and object detection (Sec. 2.1.2). The labelled datasets for these tasks are however not as large as ImageNet. This problem can be bypassed by "fine-tuning" a network that has been trained on ImageNet for downstream tasks that have less data. More concretely, the layers of the network for the downstream task that are identical to ImageNet-trained network can be initialised with its parameters. This method has been shown to perform better than randomly initialising the network [100, 189], and enables training on a different task with smaller labelled datasets.

2.1.1 Networks for semantic segmentation

A key idea to extending CNNs designed for image classification to pixel-level prediction tasks such as semantic segmentation is realising that a fully-connected layer can be considered a convolutional layer, where the filter size is equal to the size of the input feature map for that layer [101, 189]. Long *et al*. converted the fully-connected layers of AlexNet [157] and VGG [267] into convolutional ones and named such networks "Fully Convolutional

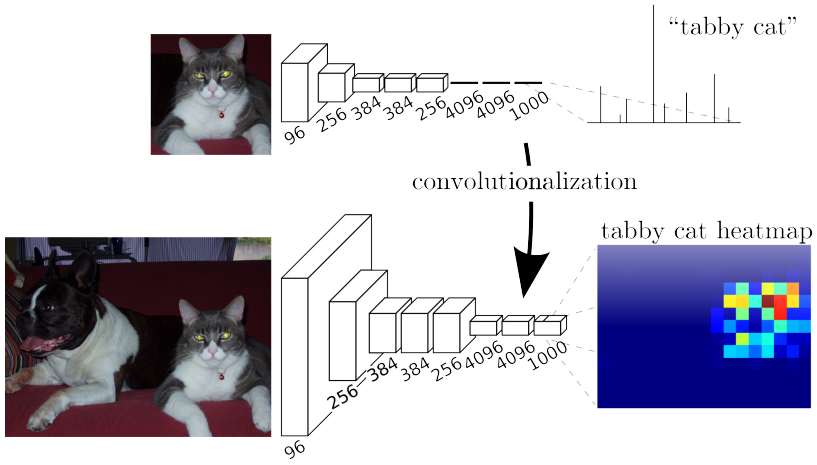
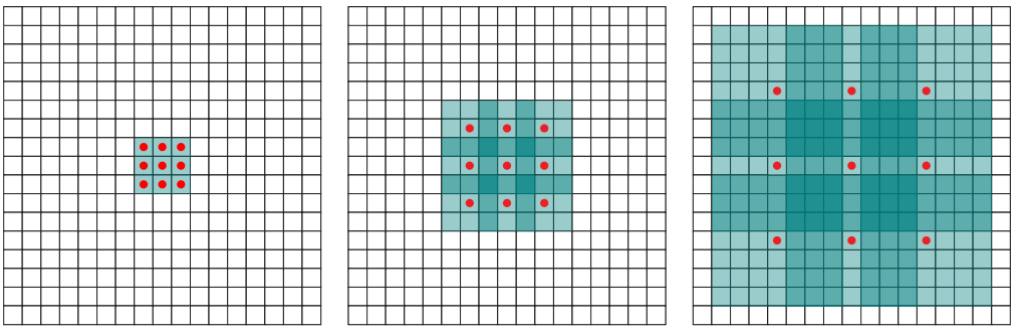


Figure 2.1: Fully convolutional networks: Fully connected layers can easily be converted into convolutional layers by recognising that a fully-connected layer is simply a convolutional layer where the size of the convolutional filter and input feature map are identical. This enables CNNs trained for image classification to output a coarse segmentation when the input is a larger image. This simple method enables good initialisation and efficient training of CNNs for pixelwise prediction. Figure from [189].

Networks" (FCNs). These networks can operate on images of any size as they only consist of convolutional-, pooling- and RelU non-linearity layers which are all dimension independent, making them suitable for pixel-level prediction tasks. However, due to the pooling layers within the network, the output would be a downsampled version of the input, as shown in Fig. 2.1. Common architectures such as AlexNet, VGG and ResNet all consist of five pooling layers which downsample the input by a factor of 2, leading to an output that is downsampled by a factor of 32. In the most rudimentary version of FCN, Long *et al*. [189] showed that simply bilinearly upsampling the coarse predictions up to the original size of the image could reach state-of-the-art performance at the time. The FCN network proposed by Long *et al*. [189] could be initialised with all the parameters of an ImageNet-trained CNN and fine-tuned on smaller datasets, leading to significantly improved results over a network initialised with random weights. Moreover, as it was simple to implement, and significantly outperformed competing methods at the time, it has been improved further in numerous subsequent works.

A shortcoming of the FCN network was that it tended to produce quite coarse and "blobby" results, as the max-pooling layers throughout the network resulted in spatial information being lost. Consequently, fine structures and object boundaries were typically segmented very poorly. Completely removing pooling layers from a CNN architecture for segmentation would not solve this problem, as layers later on in the network would not have sufficient context or "receptive field" to make a good prediction. To combat this issue,



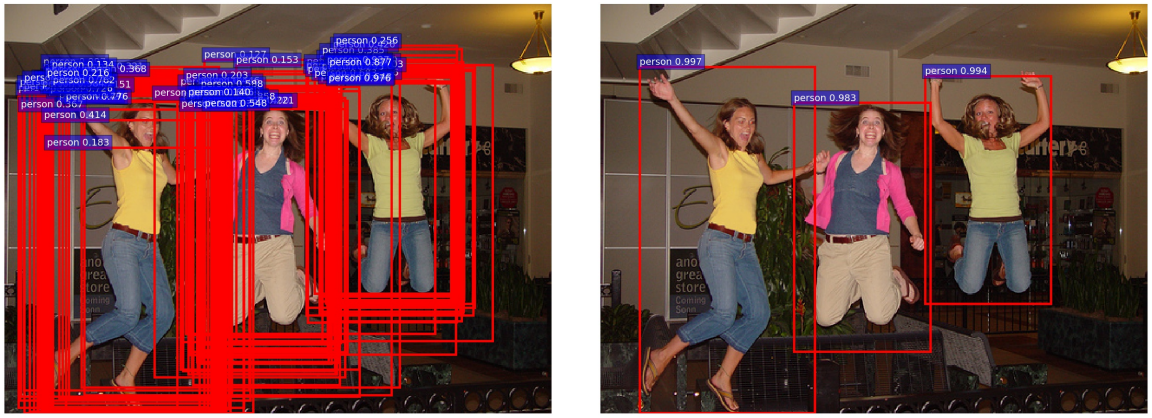
(a) (b) (c)

Figure 2.2: The red dots show the convolutional filter weights for a filter with a dilation rate of 1 (a), 2 (b) and 4 (c). The receptive field, visualised by the blue cells, is 3×3 in (a), 7×7 in (b) and 15×15 in (c). Thus, increasing the dilation rate exponentially with respect to the number of layers increases the receptive field exponentially as well, whilst the number of parameters grows only linearly. Without any dilation, the receptive field would increase only linearly with the number of layers. Figure from [319].

Atrous [44] or Dilated [319] convolutions were proposed (inspired by the "algorithme à trous" used in computing the undecimated wavelet transform [126]). As shown in Fig. 2.2, dilated convolutions allow the receptive field of a convolutional filter to be increased without increasing the number of parameters compared to an undilated filter. In [44] and [319], the last two max-pooling layers were removed, and dilated convolutions were used thereafter to ensure a large receptive field. Note that all max-pooling layers in the network were not removed due to the memory requirements of processing images at full resolution.

Other works have learned more complex networks to upsample the low-resolution output of an FCN: in [250, 215], an additional "decoder" network is employed which progressively upsamples the initial prediction to obtain the final, full-resolution output. In [250, 215], the "decoder" subnetwork contained as many layers as the original "encoder" part (an image-classification CNN), and such networks are typically referred to as "encoder-decoder" architectures. Ghiasi and Fowlkes [97] on the other hand learned the basis functions with which to upsample for a coarse-to-fine architecture.

Another avenue of improving FCN was to incorporate it within a structured prediction framework. Chen *et al*. [44] used the outputs of an FCN as the unary potentials of a DenseCRF model (detailed in Sec. 2.2). The smoothness priors encouraged by the CRF improved results, and provided sharper boundaries as well. However, Chen *et al*. [44] used the DenseCRF model as a separate module applied as post-processing to an FCN. In Chapter 3, we show how mean-field inference of CNNs can be seen as a layer of a CNN, and thus incorporated directly into a deep network. This formulation enables joint optimisation of the parameters of both the CNN and CRF, and leads to improved accuracy.



(a) Result before NMS (b) Result after NMS

Figure 2.3: Object detectors score region proposals with the likelihood of containing the target object (a). Non-maximal suppression (NMS) is used in object detection algorithms to remove duplicate detections of the same object (b). These duplicates are present because multiple proposals covering the same object (a) were processed.

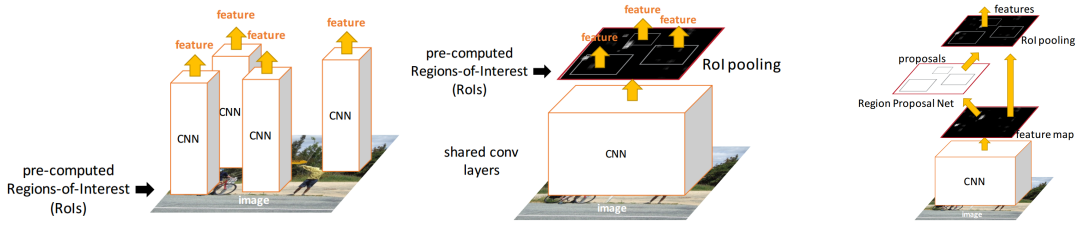
2.1.2 Networks for object detection

Object detection is the problem of localising objects (of predefined categories) in an image with a bounding box encompassing them. We first consider traditional approaches to addressing this problem, before describing how deep neural networks have been used.

2.1.2.1 History of object detection

Object detection was originally accomplished using a sliding window approach – a detection system would process a "window" of an entire image, and perform a binary classification of whether a particular object class was present in it or not [290, 297, 67]. These sliding windows were processed at multiple scales and aspect ratios in order to detect different objects. However, since this could result in multiple, overlapping predicted bounding boxes of the same object, a greedy, non-maximal suppression post-processing step was performed to remove duplicate detections [32, 88] as shown in Fig. 2.3.

An alternative approach to sliding windows is the use of region proposals. Here, a pool of (probably overlapping) image regions are first proposed, each of which is a candidate object. These candidates are then scored with how likely they are to contain the target object class. Region proposals can be seen as a method of decreasing the search space of sliding windows (which need to computed at multiple scales and aspect ratios). Common region proposal methods such as Selective Search [288] and MCG [5] were based on performing oversegmentations [89, 263] of images and grouping them into regions. Region proposals,



(a) R-CNN [100] (b) Fast R-CNN [98] (c) Faster R-CNN [243]

Figure 2.4: R-CNN (a) was one of the first object detection systems using neural networks. It used region proposals from an external system to crop out part of the input image which was then classified with an image classification network designed for ImageNet. Fast R-CNN (b) significantly increased the speed of R-CNN by computing convolutional features once for the entire region, and then pooling them for each individual proposal which were then subsequently classified. Finally, Faster-RCNN (c) generates region proposals in the network itself using a Region Proposal Network (RPN). Images are from [115].

unlike sliding windows, do not have specific aspect ratios or scales, and do not uniformly tile the image.

2.1.2.2 Region-based CNNs for detection

The Region-CNN (R-CNN) framework proposed by Girshick *et al*. [100] used neural networks to implement a region proposal-based object detection system. In R-CNN, object proposals were generated by an external system (such as [288] or [5]), and used to crop a portion of the image, as shown in Fig. 2.4(a). These proposals were then classified into one of the predefined object categories using an image-classification CNN. Bounding box regression was subsequently performed to refine the initial proposal's bounding box. Additionally, Girshick *et al*. [100] were among the first to perform transfer learning with CNNs by showing that a neural network trained on the large ImageNet dataset could be "fine-tuned" for the smaller Pascal VOC detection dataset.

In the subsequent work of Fast-RCNN, Girshick [98] accelerated this pipeline significantly by computing convolutional features once over the whole image, and then pooling the features for each object proposal. The object proposals, however, were still generated by external systems using handcrafted features. This limitation was addressed in Faster-RCNN [243] where the object proposals were generated by the neural network itself to improve both accuracy and runtime. Specifically, the authors introduced a Region Proposal Network (RPN) which produced object proposals by predicting the offset with respect to predefined "anchor boxes". These object proposals were then classified using the same network as Fast-RCNN.

Although this line of work (summarised in Fig. 2.4) has progressively incorporated more and more elements of an object detection system into a neural network that is trained end-to-end, non-maximal suppression still remains a post-processing step performed at inference time in state-of-the-art approaches. Furthermore, note that these detection algorithms all process each object proposal independently of each other, and it is only the hand-crafted non-maximal suppression post-processing step that considers all the predictions to produce the final, refined output.

2.1.2.3 Single stage detectors

The R-CNN family of object detectors, from Sec. 2.1.2.2, are sometimes referred to as "two-stage detectors" as the model first produces a set of of object proposals which are subsequently classified. Single-stage detectors, in contrast, do not have a region proposal stage.

The YOLO [241] detector splits an image into a predefined grid. For each cell in the grid, the network predicts if an object category is present or not and also its bounding box coordinates relative to this cell. The SSD detector [185] uses the same idea and proposes numerous modifications which improve both the accuracy and runtime of single-stage detectors. Single-stage detectors do not use explicit region proposals, and compensate for this with the bounding box regression that is performed by the network. An earlier work [167] also concluded that explicit region proposals contributed little to the overall performance of R-CNN [100].

In practice, two-stage detectors are typically more accurate than single-stage methods, but are also slower. An in-depth study of different CNN-based detection algorithms and their speed/accuracy trade-offs can be found in [129].

2.1.3 Shortcomings of neural networks

Deep neural networks have provided significant performance improvements in a number of classification and regression tasks [157, 117, 276, 125]. However, a common criticism is that they are not interpretable as their complexity means that one does not know why a neural network made a particular classification. This problem is illustrated by the phenomenon of adversarial examples [278] where a neural network misclassifies an (artifically created) input that has minimal perceptual differences to an input that it classified correctly. Furthermore, a theoretical understanding of why neural networks perform so well when they are optimised with stochastic gradient descent (SGD) [52, 136] – which can converge to poor local optima, and also converge very slowly – and why batch normalisation is effective [255, 25, 316] remain open questions.

2.2 Solving labelling problems with Conditional Random Fields

Many problems in computer vision, such as semantic segmentation, involve assigning a label to every pixel in the image, where the labels of each of the pixels are statistically related to each other. These can be formulated as discrete labelling problems where each node (which corresponds to each pixel in semantic segmentation) is assigned a discrete label (its object class). A common approach is to represent this labelling problem as a probabilistic graphical model, where the idea is to introduce a set of random variables $\mathcal{X} = \{X_1, X_2, \ldots, X_N\}$ corresponding to the set of nodes, $\mathcal{V} = \{1, 2, \ldots, N\}$. Each discrete random variable X_i is associated with a node $i \in \mathcal{V}$ and takes on a label l from the label set $\mathcal{L} = \{l_1, l_2, \ldots, l_L\}$ based on the observed image $\mathbf{I}$. The labels are defined by the application – in semantic segmentation, each label corresponds to an object class defined in the dataset, such as "car" or "person". Furthermore, to model the relationships between different random variables, a neighourhood system is also defined, where $\mathcal{N}_i$ denotes the set of all neighbours of variable X_i.

Any possible assignment of labels to the random variables is called a "configuration" or a "labelling", which we denote as $\mathbf{x} = (x_1, x_2, \ldots, x_N)$ where x_i is the label for the i^{th} variable. Probabilistic graphical models then model the joint, $\Pr(\mathbf{x}, \mathbf{I})$, or conditional, $\Pr(\mathbf{x}|\mathbf{I})$, probability distribution of the random variables. Models of the joint distribution are known as Markov Random Fields (MRFs), and models of the conditional distribution are known as Conditional Random Fields (CRFs). Our goal is then to find the most probable assignment, $\mathbf{x}^*$, which is defined as

$$\mathbf{x}^* = \arg\max_{\mathbf{x} \in \mathcal{L}^N} \Pr(\mathbf{x}|\mathbf{I}), \tag{2.1}$$

and known as the maximum a posteriori (MAP) solution. In the general case, obtaining the MAP solution is an NP-hard problem as it involves enumerating L^N possible configurations. In other words, the complexity of the problem scales exponentially with the number of variables being inferred. For example, the semantic segmentation problem on the Cityscapes dataset [57] involves L = 19 labels, and $N = 2048 \times 1024 \approx 2 \times 10^6$ variables (since each pixel is a random variable). However, a number of approximate inference algorithms, or exact inference algorithms for specific cases, have been developed and are widely used in computer vision [153, 28, 298, 134]. In this section, we discuss Conditional Random Fields (CRFs) and mean-field inference for them. A more detailed overview of probabilistic graphical models can be found in [152, 18, 113].

2.2.1 Conditional Random Fields (CRFs)

Conditional Random Fields model the conditional probability distribution, $\Pr(\mathbf{x}|\mathbf{I})$, of the hidden variables $\mathbf{x}$ given the observed image, $\mathbf{I}$. They are used in the segmentation problems studied in this thesis as they alleviate the need to model the observed data [165].

A probability distribution is a CRF if and only if it satisfies the following properties

$$\Pr(\mathbf{x}|\mathbf{I}) > 0, \quad \forall \mathbf{x} \in \mathcal{L}^N \qquad \text{(Positivity)} \tag{2.2}$$

$$\Pr(x_i|\{x_j : j \in \mathcal{V} - \{i\}\}, \mathbf{I}) = \Pr(x_i|\{x_j : j \in \mathcal{N}_i\}, \mathbf{I}), \quad \forall i \in \mathcal{V} \qquad \text{(Markovian)}. \tag{2.3}$$

The Markovian property says that each variable X_i is conditionally independent from all other variables given the set of all its neighbours $\mathcal{N}_i$. To this end, we define a set of cliques $c \in \mathcal{C}$ where each clique c denotes a set of random variables which are conditionally dependent on each other. A potential function $\psi_c(\mathbf{x}_c|\mathbf{I}_c)$ is also defined for each clique where $\mathbf{I}_c$ corresponds to the observed variables in the clique c. According to the Hammersley-Clifford theorem [109], the conditional probability distribution $\Pr(\mathbf{x}|\mathbf{I})$ is a Gibbs distribution that can be expressed as the product of potential functions

$$\Pr(\mathbf{x}|\mathbf{I}) = \frac{1}{Z(\mathbf{I})} \prod_{c \in \mathcal{C}} \exp\left(-\psi_c\left(\mathbf{x}_c|\mathbf{I}_c\right)\right), \tag{2.4}$$

$$= \frac{1}{Z(\mathbf{I})} \exp\left(-\sum_{c \in \mathcal{C}} \psi_c(\mathbf{x}_c|\mathbf{I}_c)\right). \tag{2.5}$$

Here, $Z(\mathbf{I}) = \sum_{\mathbf{x}} \prod_{c \in \mathcal{C}} \exp(-\psi_c(\mathbf{x_c}|\mathbf{I}_c))$ is a normalising constant known as the "partition function" which ensures that the probabilities sum to 1. Note that it is a function of the observed image $\mathbf{I}$ and the summation is only over the possible label configurations. Additionally, as shown in Fig. 2.5, a CRF and its neighourhood system can be represented as a graph where each random variable corresponds to a vertex, and relationships between vertices are represented by edges [152, 238].

The negative log-likelihood of the conditional probability is known as the energy function, E,

$$\Pr(\mathbf{x}|\mathbf{I}) = \frac{1}{Z(\mathbf{I})} \exp(-E(\mathbf{x}|\mathbf{I})), \tag{2.6}$$

$$E(\mathbf{x}|\mathbf{I}) = \sum_{c \in \mathcal{C}} \psi_c(\mathbf{x}_c|\mathbf{I}_c), \tag{2.7}$$

and fully describes the model. As a result, the MAP estimate of the conditional probability corresponds to the minimum of the energy function (which as aforementioned, is intractable to solve in the general case).

$$\mathbf{x}^* = \arg\max_{\mathbf{x}} \Pr(\mathbf{x}|\mathbf{I}) = \arg\min_{\mathbf{x}} E(\mathbf{x}|\mathbf{I}). \tag{2.8}$$

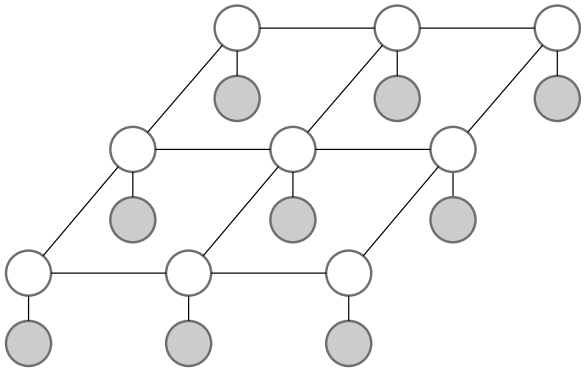


Figure 2.5: Diagram of a CRF: The hidden variables, X_j are represented by white circles, and the observed variables, I_j are represented by grey circles. The edges between observed and hidden nodes represent the unary potentials. The edges connecting the hidden nodes represent the neighbourhood system of the CRF. In this case, each node is connected to its four immediate neighbours, and the edges between every two hidden nodes represent the pairwise potentials.

In this thesis, inference of CRFs refers to estimating the MAP solution, or equivalently, minimising the energy E. The next section now details the CRF models that are used in segmentation problems.

2.2.2 CRF models

CRF models typically consist of unary, pairwise and higher order potentials, where the size of the cliques are one, two and three or more respectively,

$$E(\mathbf{x}|\mathbf{I}) = \sum_{i \in \mathcal{V}} \psi_i(x_i) + \sum_i \sum_{j \in \mathcal{N}_i} \psi_{i,j}(x_i, x_j) + \sum_{h \in \mathcal{H}} \psi_h(\mathbf{x}_h). \tag{2.9}$$

The unary term, $\psi_i(x_i)$, is defined over a clique of one variable and captures the correlation between an unobserved X_i variable and the observed evidence, I_i. It is typically obtained from a classifier, as the negative log-likelihood of variable X_i taking label x_i, which could be a fully convolutional neural network as described in the Sec. 2.1.1. With only a unary term, each variable x_i would be predicted independently of each other. As local evidence is usually noisy (Fig. 2.6), this usually leads to suboptimal results.

The pairwise term, $\psi_{i,j}(x_i, x_j)$, models interactions between pairs of variables, x_i and x_j. It is usually used to encourage predictions to be smooth by encouraging neighbouring variables to take on the same label.

Higher order terms, $\psi_h(\mathbf{x}_h)$, act on cliques containing more than two variables and encourage higher-order consistency constraints. These include consistency over larger regions [147], consistency with respect to other scene understanding algorithms [164, 318], and co-occurence priors [162].

As pairwise terms are common in most CRF models used in computer vision problems, we now consider them in more detail.

Pairwise terms Pairwise terms generally encourage smoothness in the labelling by encouraging a pair of neighbouring variables to take on the same label. An example is the contrast-sensitive Potts model:

$$\psi_{i,j}(x_i, x_j) = \begin{cases} 0 & \text{if } x_i = x_j \\ \kappa(I_i, I_j) & \text{otherwise.} \end{cases} \tag{2.10}$$

Here, no cost is incurred if two variables X_i and X_j take on the same label. Otherwise, a penalty dependent on the image features is used. A common approach [154] is to use a mixture of Gaussian kernels over intensity values, f, and positional features p,

$$\kappa(I_i, I_j) = w_1 \exp\left(-\frac{|p_i - p_j|^2}{2\theta_\alpha^2} - \frac{|f_i - f_j|^2}{2\theta_\beta^2}\right) + w_2 \exp\left(-\frac{|p_i - p_j|^2}{2\theta_\gamma^2}\right). \tag{2.11}$$

Here, w_1 and w_2 are the co-efficients of the kernel components whose bandwidth is determined by the θ hyperparameters. The first kernel is an edge-preserving bilateral filter and encourages pixels with similar appearance to take on the same label. The second kernel is a Gaussian smoothing filter which enforces spatial smoothness by removing isolated, and thus noisy, regions [154].

Early works used four- or eight-neighbourhood connectivity (Fig. 2.5) for the pairwise terms, which we denote as "Grid-CRF". Such models were popular as inference algorithms (which had theoretical guarantees and were in some cases exact) existed for these models which were based on reducing the energy minimisation problem (Eq. 2.8) to the minimum cut in a graph [153]. Grid-CRF models, however, have limited expressivity as they are not able to model long-range interactions between variables. This problem is remedied by considering densely-connected graphs where every pair of pixels is connected. The primary challenge for densely-connected graphs is the prohibitive run-time for graph-cut based inference methods. However, for models with pairwise potentials that are a mixture of Gaussian kernels (Eq. 2.11), fast-run times are achievable using the permutohedral lattice filtering method [2] and approximate, parallel mean-field inference [154]. Krähenbühl *et al*. [154] showed that with parallel mean-field inference, the runtime was only 0.2 seconds for an image from the MSRC dataset [266], compared to 72 hours for graph-cuts [153].

In practice, the more expressive densely-connected pairwise model achieves the best results and runtime. This result is shown in Fig. 2.6 which illustrates how segmentation quality has been improved with densely-connected pairwise terms and more accurate unary potentials from CNNs. The next section provides an overview of mean-field inference, which facilitates densely-connected pairwise terms.

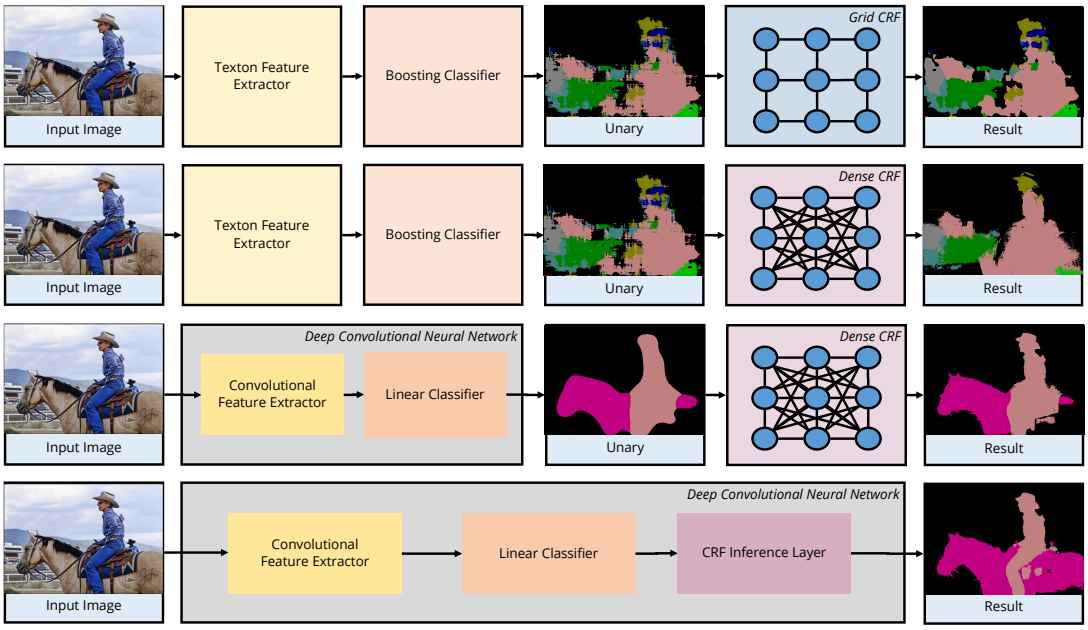


Figure 2.6: The evolution of semantic segmentation systems. The first row shows the early "TextonBoost" work [266] that computed unary potentials using handcrafted Texton [266] features and employed a CRF with limited 8-connectivity. The DenseCRF work of [154], shown in the second row, used densely connected pairwise potentials and approximate mean-field inference. The more expressive model achieved significantly improved performance. Numerous works, such as [44], have replaced the early hand-crafted features with deep neural networks which can learn features from large amounts of data and used the outputs of a CNN as the unary potentials of a DenseCRF model. In fact, works such as [189] showed that unary potentials from CNNs (without any CRF) on their own achieved significantly better results than previous methods. Subsequent methods (such as the algorithm described in Chapter 3) have combined inference of a CRF within the deep neural network itself to obtain state-of-the-art results. Result images for this figure were obtained using the publicly available code of [163, 154, 44]. The final row shows the result from the algorithm in Chapter 3.

2.2.3 Mean-field inference

Mean-field inference approximates a complex probability distribution, P, with a simpler distribution, Q. Inference problems, such as finding the MAP solution, are then performed on this simpler distribution as a replacement for the actual probability distribution (such as the one defined by the CRF) of interest. This technique is employed when performing inference directly on P is intractable.

Thus given a complex probability distribution, $P(\mathbf{x})$, which one cannot solve for, the main steps are as follows:

1. Define a function that allows us to compare the similarity between the complex, intractable distribution, P and its tractable approximation Q.

2. Specify the class of probability distributions in which we want to find a similar distribution Q.

3. Find Q from the chosen class that is the closest to P.

4. Solve the inference problem for Q.

We now detail these steps below. A more detailed analysis of mean-field inference can be found in [113].

Distance function A natural measure to measure the similarity of two probability distributions, P and Q is the KL divergence, $D_{KL}(Q||P)$:

$$D_{KL}(Q||P) = \sum_{\mathbf{x}} Q(\mathbf{x}) \log \frac{Q(\mathbf{x})}{P(\mathbf{x})} \tag{2.12}$$

$$= \sum_{\mathbf{x}} Q(\mathbf{x}) \log Q(\mathbf{x}) - \sum_{\mathbf{x}} Q(\mathbf{x}) \log P(\mathbf{x}). \tag{2.13}$$

The KL divergence satisfies the basic properties of an error measure as $D_{KL}(Q||P) \geq 0$ for all P and Q, and $D_{KL}(Q||P) = 0$ if and only if $P = Q$. However, it is not a distance metric as it is not commutative, $D_{KL}(Q||P) \neq D_{KL}(P||Q)$ and it does not satisfy the triangle inequality either.

Substituting the Gibbs distribution of the CRF (Eq. 2.6) into the KL divergence (and omitting the conditioning on $\mathbf{I}$ to simplify the notation), we obtain

$$D_{KL}(Q||P) = -\sum_{\mathbf{x}} Q(\mathbf{x}) \log \left(\frac{1}{Z} \exp\left(-E(\mathbf{x})\right) \right) + \sum_{\mathbf{x}} Q(\mathbf{x}) \log Q(\mathbf{x}) \tag{2.14}$$

$$= \sum_{\mathbf{x}} Q(\mathbf{x}) E(\mathbf{x}) + \log Z + \sum_{\mathbf{x}} Q(\mathbf{x}) \log Q(\mathbf{x}). \tag{2.15}$$

For the second term, we used the fact that $\sum_{\mathbf{x}} Q(\mathbf{x}) = 1$. Since $\log Z$ is a constant, minimising the KL divergence between Q and P is equivalent to minimising

$$F(Q) = \sum_{\mathbf{x}} Q(\mathbf{x}) E(\mathbf{x}) + \sum_{\mathbf{x}} Q(\mathbf{x}) \log Q(\mathbf{x}). \tag{2.16}$$

The first term is thus the expected value of the energy, $E(\mathbf{x})$, under the distribution $Q(\mathbf{x})$ whilst the second term is the negative entropy of $Q(\mathbf{x})$.

Expanding the first term further, and rearranging the order of summation, we obtain

$$\sum_{\mathbf{x}} Q(\mathbf{x}) E(\mathbf{x}) = \sum_{c \in \mathcal{C}} \sum_{\mathbf{x}} Q(\mathbf{x}) \psi_c(\mathbf{x}_c). \tag{2.17}$$

For a given clique c, the summation over $\mathbf{x}$ can be broken down further into a sum over the variables within the clique c, and those variables not within c, which we denote as c'. Thus,

$$\sum_{\mathbf{x}} Q(\mathbf{x}) \psi_c(\mathbf{x}_c) = \sum_{\mathbf{x}_c} \sum_{\mathbf{x}_{c'}} Q(\mathbf{x}) \psi_c(\mathbf{x}_c) \tag{2.18}$$

$$= \sum_{\mathbf{x}_c} \psi_c(\mathbf{x}_c) \sum_{\mathbf{x}_{c'}} Q(\mathbf{x}). \tag{2.19}$$

Note that we can move $\psi_c(\mathbf{x}_c)$ outside the last summation as it is constant as $\mathbf{x}'_c$ varies. Observing that $\sum_{\mathbf{x}_{c'}} Q(\mathbf{x})$ is the marginal probability, $Q(\mathbf{X}_c = \mathbf{x}_c)$, Eq. 2.17 can be written as

$$Q(\mathbf{x})E(\mathbf{x}) = \sum_{c \in \mathcal{C}} \sum_{\mathbf{x}_c} \psi_c(\mathbf{x}_c) Q(\mathbf{x}_c). \tag{2.20}$$

In other words, the expected value of the energy, E, under the distribution Q is equal to the sum of the expected clique energies, $\psi_c(\mathbf{x}_c)$.

Class of distributions for Q The simplest choice of Q, and also the naïve mean-field approximation, is to define Q as a product of independent distributions. Defining a distribution for each random variable X_i, we obtain

$$Q(\mathbf{X} = \mathbf{x}) = \prod_{i \in \mathcal{V}} Q(X_i = x_i). \tag{2.21}$$

An analysis of this simple approximation can be found in [113]. An advantage of it is that the negative entropy term from Eq. 2.16 decomposes into

$$Q(\mathbf{x}) \log Q(\mathbf{x}) = \sum_{i \in \mathcal{V}} \sum_{l \in \mathcal{L}} Q(X_i = l) \log Q(X_i = l). \tag{2.22}$$

Combining the above equations, the term minimised by the KL divergence is thus

$$F(Q) = \sum_{\mathbf{x}} Q(\mathbf{x})E(\mathbf{x}) + \sum_{\mathbf{x}} Q(\mathbf{x}) \log Q(\mathbf{x}). \tag{2.23}$$

$$= \sum_{c \in \mathcal{C}} \sum_{\mathbf{x}_c} \psi_c(\mathbf{x}_c) \prod_{i \in c} Q(x_i) + \sum_{i \in \mathcal{V}} \sum_{l \in \mathcal{L}} Q(X_i = l) \log Q(X_i = l). \tag{2.24}$$

Minimising the KL divergence To find the fully-factorised distribution Q that is closest to the original CRF distribution P, we formulate the optimisation problem

$$\begin{aligned} &\underset{Q(\mathbf{x})}{\text{minimise}} && F(Q) \\ &\text{subject to} && \sum_{l \in \mathcal{L}} Q(X_i = l) = 1 \quad \forall i \in \mathcal{V} \end{aligned} \tag{2.25}$$

This problem is approached with Lagrange multipliers. Denoting the Lagrange parameters as λ_i, the Lagrangian can be written as

$$L(Q, \lambda) = F(Q) + \sum_{i \in \mathcal{V}} \lambda_i \left(\sum_{l \in \mathcal{L}} Q(X_i = l) - 1 \right). \tag{2.26}$$

Taking the partial derivative of $L(Q, \lambda)$ with respect to an element $Q(X_i = l)$, setting it to zero, rearranging terms and renormalising [113] leads to the mean-field update for the

marginal distribution of variable i for label l, $Q(X_i = l)$, as

$$Q^{t+1}(X_i = l) = \frac{1}{Z_i} \exp\left(-\sum_{c \in \mathcal{C}} \sum_{\{\mathbf{x}_c | x_i = l\}} Q^t(\mathbf{x}_{c-i})\, \psi_c(\mathbf{x}_c) \right). \tag{2.27}$$

where Q^t is the marginal after the t^{th} iteration, $\mathbf{x}_c$ an assignment to all variables in clique c and $\mathbf{x}_{c-i}$ an assignment to all variables in c except for X_i. The normalisation constant, Z_i, is defined as

$$Z_i = \sum_{l \in \mathcal{L}} \exp\left(-\sum_{c \in \mathcal{C}} \sum_{\{\mathbf{x}_c | x_i = l\}} Q^t(\mathbf{x}_{c-i})\, \psi_c(\mathbf{x}_c) \right). \tag{2.28}$$

Note that by substituting Eq. 2.28 into Eq. 2.27, we can see that the mean-field update thus involves a softmax operation to normalise the values, ensuring that the marginal distribution Q is valid (non-negative and sums to 1) after each iteration. The mean-field update for the DenseCRF model [154] with unary and pairwise potentials is thus

$$Q^{t+1}(X_i = l) = \frac{1}{Z_i} \exp\left(-\psi_i(l) - \sum_{l' \in \mathcal{L}} \sum_{j \neq i} Q(X_j = l') \psi_{i,j}(l, l') \right). \tag{2.29}$$

Updating each marginal distribution, Q_i, sequentially will converge to a local minimum [113, 152] (note that one cycle through all the indices, i, is an "iteration" that updates t). However, for densely connected pairwise terms, this means that updating all N marginal distributions has a time complexity of $O(N^2)$. This is because the update for each variable requires a summation involving all other variables. This quadratic complexity is prohibitive for the large graphs typically involved in computer vision problems. Krähenbühl *et al*. [154], however, showed that using fast filtering techniques [2] which can be used with Gaussian pairwise potentials such as Eq. 2.11, all the marginal distributions could be updated in parallel with a time complexity of $O(N)$. Although this parallel update has no convergence guarantees, it has been empirically observed to converge [154] and is widely used in practice as it makes inference of DenseCRF's practical. Finally, note that as Q is fully factorised, its MAP solution is simply the maximiser of each of its marginals.

The next chapter now extends the pairwise model of DenseCRF with more expressive higher order potentials which improve accuracy in semantic segmentation problems. Parallel mean-field inference is performed to obtain the final solution, and its iterations are unrolled to form a differentiable recurrent neural network. This enables joint optimisation of both the parameters of the underlying CNN which produce the unary potentials, and the CRF with higher order potentials using stochastic gradient descent.

Chapter 3

Higher Order Conditional Random Fields in Deep Neural Networks

This paper addresses the problem of semantic segmentation using deep learning. Most segmentation systems include a Conditional Random Field (CRF) to produce a structured output that is consistent with the image's visual features. Recent deep learning approaches have incorporated CRFs into Convolutional Neural Networks (CNNs), with some even training the CRF end-to-end with the rest of the network. However, these approaches have not employed higher order potentials, which have previously been shown to significantly improve segmentation performance. In this paper, we demonstrate that two types of higher order potential, based on object detections and superpixels, can be included in a CRF embedded within a deep network. We design these higher order potentials to allow inference with the differentiable mean field algorithm. As a result, all the parameters of our richer CRF model can be learned end-to-end with our pixelwise CNN classifier. We achieve state-of-the-art segmentation performance on the PASCAL VOC benchmark with these trainable higher order potentials.

3.1 Introduction

Semantic segmentation involves assigning a visual object class label to every pixel in an image, resulting in a segmentation with a semantic meaning for each segment. While a strong pixel-level classifier is critical for obtaining high accuracy in this task, it is also important to enforce the consistency of the semantic segmentation output with visual features of the image. For example, segmentation boundaries should usually coincide with strong edges in the image, and regions in the image with similar appearance should have the same label.

Recent advances in deep learning have enabled researchers to create stronger classifiers, with automatically learned features, within a Convolutional Neural Network (CNN) [157,

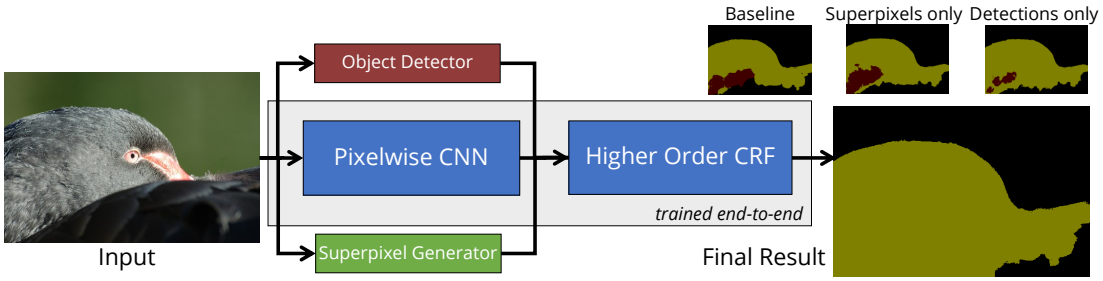


Figure 3.1: Overview of our system. We train a Higher Order CRF end-to-end with a pixelwise CNN classifier. Our higher order detection and superpixel potentials improve significantly over our baseline containing only pairwise potentials.

267, 189]. This has resulted in large improvements in semantic segmentation accuracy on widely used benchmarks such as PASCAL VOC [81]. CNN classifiers are now considered the standard choice for pixel-level classifiers used in semantic segmentation.

On the other hand, probabilistic graphical models have long been popular for structured prediction of labels, with constraints enforcing label consistency. Conditional Random Fields (CRFs) have been the most common framework, and various rich and expressive models [163, 164, 296], based on higher order clique potentials, have been developed to improve segmentation performance.

Whilst some deep learning methods showed impressive performance in semantic segmentation without incorporating graphical models [189, 112], current state-of-the-art methods [188, 329, 178, 44] have all incorporated graphical models into the deep learning framework in some form. However, we observe that the CRFs that have been incorporated into deep learning techniques are still rather rudimentary as they consist of only unary and pairwise potentials [329]. In this paper, we show that CRFs with carefully designed higher order potentials (potentials defined over cliques consisting of more than two nodes) can also be modelled as CNN layers when using mean field inference [152]. The advantage of performing CRF inference within a CNN is that it enables joint optimisation of CNN classifier weights and CRF parameters during the end-to-end training of the complete system. Intuitively, the classifier and the graphical model learn to optimally co-operate with each other during the joint training.

We introduce two types of higher order potential into the CRF embedded in our deep network: object-detection based potentials and superpixel-based potentials. The primary idea of using object-detection potentials is to use the outputs of an off-the-shelf object detector as additional semantic cues for finding the segmentation of an image. Intuitively, an object detector with a high recall can help the semantic segmentation algorithm by finding objects appearing in an image. As shown in Fig. 3.1, our method is able to recover from poor

segmentation unaries when we have a confident detector response. However, our method is robust to false positives identified by the object detector since CRF inference identifies and rejects false detections that do not agree with other types of energies present in the CRF.

Superpixel-based higher order potentials encourage label consistency over superpixels obtained by oversegmentation. This is motivated by the fact that regions defined by superpixels are likely to contain pixels from the same visual object. Once again, our formulation is robust to the violations of this assumption and errors in the initial superpixel generation step. In practice, we noted that this potential is effective for getting rid of small regions of spurious labels that are inconsistent with the correct labels of their neighbours.

We evaluate our higher order potentials on the PASCAL VOC 2012 semantic segmentation benchmark as well as the PASCAL Context dataset, to show significant improvements over our baseline and achieve state-of-the art results.

3.2 Related Work

Before deep learning became prominent, semantic segmentation was performed with dense hand-crafted features which were fed into a per-pixel or region classifier [266]. The individual predictions made by these classifiers were often noisy as they lacked global context, and were thus post-processed with a CRF, making use of prior knowledge such as the fact that nearby pixels, as well as pixels of similar appearance, are likely to share the same class label [266].

The CRF model of [266] initially contained only unary and pairwise terms in an 8-neighbourhood, which [148] showed can result in shrinkage bias. Numerous improvements to this model were subsequently proposed including: densely connected pairwise potentials facilitating interactions between all pairs of image pixels [154], formulating higher order potentials defined over cliques larger than two nodes [148, 163] in order to capture more context, modelling co-occurrence of object classes [162, 239, 103], and utilising the results of object detectors [164, 318, 306].

Recent advances in deep learning have allowed us to replace hand-crafted features with features learned specifically for semantic segmentation. The strength of these representations was illustrated by [189] who achieved significant improvements over previous hand-crafted methods without using any CRF post-processing. Chen *et al.* [44] showed further improvements by post-processing the results of a CNN with a CRF. Subsequent works [329, 178, 179, 188] have taken this idea further by incorporating a CRF as layers within a deep network and then learning parameters of both the CRF and CNN together via backpropagation.

In terms of enhancements to conventional CRF models, Ladicky *et al.* [164] proposed using an off-the-shelf object detector to provide additional cues for semantic segmentation. Unlike other approaches that refine a bounding-box detection to produce a segmentation [112, 317], this method used detector outputs as a soft constraint and can thus recover from object detection errors. Their formulation, however, used graph-cut inference, which was only tractable due to the absence of dense pairwise potentials. Object detectors have also been used by [318, 275], who also modelled variables that describe the degree to which an object hypothesis is accepted.

We formulate the detection potential in a different manner to [164, 275, 318] so that it is amenable to mean field inference. Mean field permits inference with dense pairwise connections, which results in substantial accuracy improvements [154, 44, 329]. Furthermore, mean field updates related to our potentials are differentiable and its parameters can thus be learned in our end-to-end trainable architecture.

We also note that while the semantic segmentation problem has mostly been formulated in terms of pixels [266, 189, 329], some have expressed it in terms of superpixels [38, 83, 64]. Superpixels can capture more context than a single pixel and computational costs can also be reduced if one considers pairwise interactions between superpixels rather than individual pixels [318]. However, such superpixel representations assume that the segments share boundaries with objects in an image, which is not always true. As a result, several authors [163, 296] have employed higher order potentials defined over superpixels that encourage label consistency over regions, but do not strictly enforce it. This approach also allows multiple, non-hierarchical layers of superpixels to be integrated. Our formulation uses this kind of higher order potential, but in an end-to-end trainable CNN.

Graphical models have been used with CNNs in other areas besides semantic segmentation, such as in pose-estimation [282] and group activity recognition [72]. Alternatively, Ionescu *et al.* [131] incorporated structure into a deep network with structured matrix layers and matrix backpropagation. However, the nature of models used in these works is substantially different to ours. Some early works that advocated gradient backpropagation through graphical model inference for parameter optimisation include [73, 155] and [251].

Our work differentiates from the above works since, to our knowledge, we are the first to propose and conduct a thorough experimental investigation of higher order potentials that are based on detection outputs and superpixel segmentation in a CRF which is learned end-to-end in a deep network. Note that although [296] formulated mean field inference with higher order potentials, they did not consider object detection potentials at all, nor were the parameters learned.

3.3 Conditional Random Fields

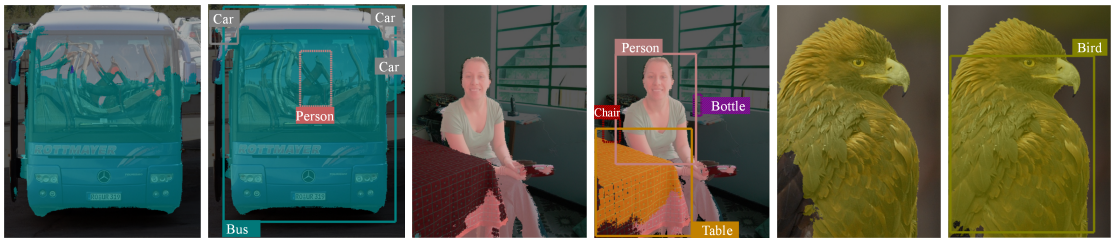
We now review conditional random fields used in semantic segmentation and introduce the notation used in the paper. Take an image $\mathbf{I}$ with N pixels, indexed $1, 2, \ldots, N$. In semantic segmentation, we attempt to assign every pixel a label from a predefined set of labels $\mathcal{L} = \{l_1, l_2, \ldots, l_L\}$. Define a set of random variables $X_1, X_2, \ldots, X_N$, one for each pixel, where each $X_i \in \mathcal{L}$. Let $\mathbf{X} = [X_1\ X_2\ \ldots\ X_N]^T$. Any particular assignment $\mathbf{x}$ to $\mathbf{X}$ is thus a solution to the semantic segmentation problem.

We use notations $\{\mathbf{V}\}$, and $\mathbf{V}^{(i)}$ to represent the set of elements of a vector $\mathbf{V}$, and the i^{th} element of $\mathbf{V}$, respectively. A CRF models the conditional distribution, $\Pr(\mathbf{x}|\mathbf{I}) = (1/Z(\mathbf{I}))\exp(-E(\mathbf{x}|\mathbf{I}))$, where $E(\mathbf{x}|\mathbf{I})$ is the *energy* of the assignment $\mathbf{x}$ and $Z(\mathbf{I})$ is the normalisation factor known as the partition function. We drop the conditioning on $\mathbf{I}$ hereafter to keep the notation uncluttered. The energy, $E(\mathbf{x})$, of an assignment is defined using the set of cliques $\mathcal{C}$ defined in the CRF. More specifically, $E(\mathbf{x}) = \sum_{c\in\mathcal{C}} \psi_c(\mathbf{x}_c)$, where $\mathbf{x}_c$ is a vector formed by selecting elements of $\mathbf{x}$ that correspond to random variables belonging to the clique c, and $\psi_c(.)$ is the cost function for the clique c. The function, $\psi_c(.)$, usually uses prior knowledge about a good segmentation, as well as information from the image, the observation the CRF is conditioned on.

Minimising the energy yields the maximum a posteriori (MAP) labelling of the image *i.e.* the most probable label assignment given the observation (image). When dense pairwise potentials are used in the CRF to obtain higher accuracy, exact inference is impracticable, and one has to resort to an approximate inference method such as mean field inference [154]. Mean field inference is particularly appealing in a deep learning setting since it is possible to formulate it as a Recurrent Neural Network [329].

3.4 CRF with Higher Order Potentials

Many CRF models that have been incorporated into deep learning frameworks [44, 329] have so far used only unary and pairwise potentials. However, potentials defined on higher order cliques have been shown to be useful in previous works such as [148, 296]. The key contribution of this paper is to show that a number of explicit higher order potentials can be added to CRFs to improve image segmentation, while staying compatible with deep learning. We formulate these higher order potentials in a manner that mean field inference can still be used to solve the CRF. Advantages of mean field inference are twofold: First, it enables efficient inference when using densely-connected pairwise potentials. Multiple works, [155, 329] have shown that dense pairwise connections result in substantial accuracy improvements, particularly at image boundaries [44, 154]. Secondly, we keep all our mean



(a) (b) (c) (d) (e) (f)

Figure 3.2: Utility of object detections as another cue for semantic segmentation. For every pair, segmentation on the left was produced with only unary and pairwise potentials. Detection based potentials were added to produce the result on the right. Note how we are able to improve our segmentations for the bus, table and bird over their respective baselines. Furthermore, our system is able to reject erroneous detections such as the person in (b) and the bottle and chair in (d). Images were taken from the PASCAL VOC 2012 reduced validation set. Baseline results were produced using the public code and model of [329].

field updates differentiable with respect to their inputs as well as the CRF parameters introduced. This design enables us to use backpropagation to automatically learn all the parameters in the introduced potentials.

We use two types of higher order potential, one based on object detections and the other based on superpixels. These are detailed in Sections 3.4.1 and 3.4.2 respectively. Our complete CRF model is represented by

$$E(\mathbf{x}) = \sum_i \psi_i^U(x_i) + \sum_{i<j} \psi_{ij}^P(x_i, x_j) \; + \sum_d \psi_d^{\text{Det}}(\mathbf{x}_d) + \sum_s \psi_s^{\text{SP}}(\mathbf{x}_s), \tag{3.1}$$

where the first two terms $\psi_i^U(.)$ and $\psi_{ij}^P(.,.)$ are the usual unary and densely-connected pairwise energies [154] and the last two terms are the newly introduced higher order energies. Energies from the object detection take the form $\psi_d^{\text{Det}}(\mathbf{x}_d)$, where vector $\mathbf{x}_d$ is formed by elements of $\mathbf{x}$ that correspond to the foreground pixels of the d^{th} object detection. Superpixel label consistency based energies take the form $\psi_s^{\text{SP}}(\mathbf{x}_s)$, where $\mathbf{x}_s$ is formed by elements of $\mathbf{x}$ that correspond to the pixels belonging to the s^{th} superpixel.

3.4.1 Object Detection Based Potentials

Semantic segmentation errors can be classified into two broad categories [63]: recognition and boundary errors. Boundary errors occur when semantic labels are incorrect at the edges of objects, and it has been shown that densely connected CRFs with appearance-consistency terms are effective at combating this problem [154]. On the other hand, recognition errors occur when object categories are recognised incorrectly or not at all. A CRF with only unary and pairwise potentials cannot effectively correct these errors since they are caused by poor unary classification. However, we propose that a state-of-the-art object detector [98, 243]

capable of recognising and localising objects, can provide important information in this situation and help reduce the recognition error, as shown in Fig. 3.2.

A key challenge in feeding-in object-detection potentials to semantic segmentation are false detections. A naïve approach of adding an object detector's output to a CRF formulated to solve the problem of semantic segmentation would confuse the CRF due to the presence of the false positives in the detector's output. Therefore, a robust formulation, which can automatically reject object detection false positives when they do not agree with other types of potentials in the CRF, is desired. Furthermore, since we are aiming for an end-to-end trainable CRF which can be incorporated into a deep neural network, the energy formulation should permit a fully differentiable inference procedure. We now propose a formulation which has both of these desired properties.

Assume that we have D object detections for a given image, and that the d^{th} detection is of the form (l_d, s_d, F_d), where $l_d \in \mathcal{L}$ is the class label of the detected object, s_d is the confidence score of the detection, and $F_d \subseteq \{1, 2, \ldots, N\}$, is the set of indices of the pixels belonging to the foreground of the detection. The foreground within a detection bounding box could be obtained using a foreground/background segmentation method (*i.e.* GrabCut [252]), and represents a crude segmentation of the detected object. Using our detection potentials, we aim to encourage the set of pixels represented by F_d, to take the label l_d. However, this should not be a hard constraint since the foreground segmentation could be inaccurate and the detection itself could be a false detection. We therefore seek a soft constraint that assigns a penalty if a pixel in F_d takes a label other than l_d. Moreover, if other energies used in the CRF strongly suggest that many pixels in F_d do not belong to the class l_d, the detection d should be identified as invalid.

An approach to accomplish this is described in [164] and [318]. However, in both cases, dense pairwise connections were absent and different inference methods were used. In contrast, we would like to use the mean field approximation to enable efficient inference with dense pairwise connections [154], and also because its inference procedure is fully differentiable. We therefore use a detection potential formulation quite different to the ones used in [164] and [318].

In our formulation, as done in [164] and [318], we first introduce latent binary random variables $Y_1, Y_2, \ldots Y_D$, one for each detection. The interpretation for the random variable Y_d that corresponds to the d^{th} detection is as follows: If the d^{th} detection has been found to be valid after inference, Y_d will be set to 1, it will be 0 otherwise. Mean field inference probabilistically decides the final value of Y_d. Note that, through this formulation, we can account for the fact that the initial detection could have been a false positive: some of the

detections obtained from the object detector may be identified to be false following CRF inference.

All Y_d variables are added to the CRF which previously contained only X_i variables. Let each $(\mathbf{X}_d, Y_d)$, where $\{\mathbf{X}_d\} = \{X_i \in \{\mathbf{X}\} | i \in F_d\}$, form a clique c_d in the CRF. We define the detection-based higher order energy associated with a particular assignment $(\mathbf{x}_d, y_d)$ to the clique $(\mathbf{X}_d, Y_d)$ as follows:

$$\psi_d^{\text{Det}}(\mathbf{x}_d, y_d) = \begin{cases} w_{\text{Det}} \frac{s_d}{n_d} \sum_{i=1}^{n_d} [x_d^{(i)} = l_d] & \text{if } y_d = 0, \\ \\ w_{\text{Det}} \frac{s_d}{n_d} \sum_{i=1}^{n_d} [x_d^{(i)} \neq l_d] & \text{if } y_d = 1, \end{cases} \tag{3.2}$$

where $n_d = |F_d|$ is the number of foreground pixels in the d^{th} detection, $x_d^{(i)}$ is the i^{th} element of the vector $\mathbf{x}_d$, w_{Det} is a learnable weight parameter, and $[\,.\,]$ is the Iverson bracket. Note that this potential encourages $X_d^{(i)}$s to take the value l_d when Y_d is 1, and at the same time encourages Y_d to be 0 when many $X_d^{(i)}$s do not take l_d. In other words, it enforces the consistency among $X_d^{(i)}$s and Y_d.

An important property of the above definition of $\psi_d^{\text{Det}}(.)$ is that it can be simplified as a sum of pairwise potentials between Y_d and each $X_d^{(i)}$ for $i = 1, 2, \ldots, n_d$. That is,

$$\begin{aligned} \psi_d^{\text{Det}}(\mathbf{x}_d, y_d) &= \sum_{i=1}^{n_d} f_d(x_d^{(i)}, y_d), \text{ where,} \\ f_d(x_d^{(i)}, y_d) &= \begin{cases} w_{\text{Det}} \frac{s_d}{n_d} [x_d^{(i)} = l_d] & \text{if } y_d = 0, \\ w_{\text{Det}} \frac{s_d}{n_d} [x_d^{(i)} \neq l_d] & \text{if } y_d = 1. \end{cases} \end{aligned} \tag{3.3}$$

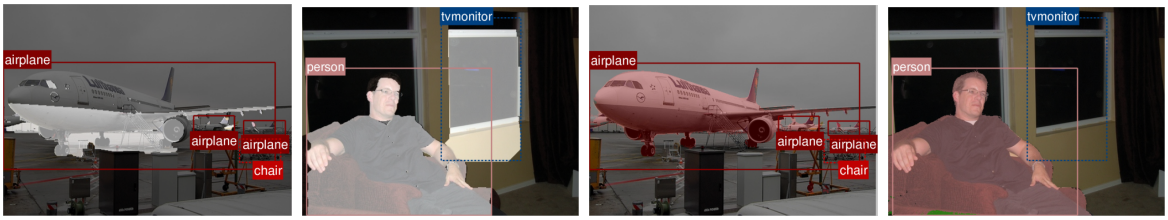
We make use of this simplification in Section 3.5 when deriving the mean field updates associated with this potential.

For the latent Y variables, in addition to the joint potentials with X variables, described in Eq. (3.2) and (3.3), we also include unary potentials, which are initialised from the score s_d of the object detection. The underlying idea is that if the object detector detects an object with high confidence, the CRF in turn starts with a high initial confidence about the validity of that detection. This confidence can, of course, change during the CRF inference depending on other information (*e.g.* segmentation unary potentials) available to the CRF.

Examples of input images with multiple detections and GrabCut foreground masks are shown in Figure 3.3. Note how false detections are ignored and erroneous parts of the foreground mask are also largely ignored.

3.4.2 Superpixel Based Potentials

The next type of higher order potential we use is based on the idea that superpixels obtained from oversegmentation [89, 1] quite often contain pixels from the same visual object. It is



(a) (b) (c) (d)

Figure 3.3: Effects of imperfect foreground segmentation. (a,b) Detected objects, as well as the foreground masks obtained from GrabCut. (c,d) Output using detection potentials. Incorrect parts of the foreground segmentation of the main aeroplane, and entire TV detection have been ignored by CRF inference as they did not agree with the other energy terms. The person is a failure case though as the detection has caused part of the sofa to be erroneously labelled.

(a) (b) (c)

Figure 3.4: Segmentation enhancement from superpixel based potentials. (a) The output of our system without any superpixel potentials. (b) Superpixels obtained from the image using the method of [89]. Only one “layer" of superpixels is shown. In practice, we used four. (c) The output using superpixel potentials. The result has improved as we encourage consistency over superpixel regions. This removes some of the spurious noise that was present previously.

therefore natural to encourage pixels inside a superpixel to have the same semantic label. Once again, this should not be a hard constraint in order to keep the algorithm robust to initial superpixel segmentation errors and to violations of this key assumption.

We use two types of energies in the CRF to encourage superpixel consistency in semantic segmentation. Firstly, we use the P^n-Potts model type energy [147], which is described by,

$$\psi_s^{\mathrm{SP}}(\mathbf{x}_s) = \begin{cases} w_{\mathrm{Low}}(l) & \text{if all } x_s^{(i)} = l, \\ w_{\mathrm{High}} & \text{otherwise,} \end{cases} \tag{3.4}$$

where $w_{\mathrm{Low}}(l) < w_{\mathrm{High}}$ for all l, and $\mathbf{x}_s$ are the elements of $\mathbf{x}$ that correspond to a superpixel. The primary idea is that assigning different labels to pixels in the same superpixel incurs a higher cost, whereas one obtains a lower cost if the labelling is consistent throughout the superpixel. Costs $w_{\mathrm{Low}}(l)$ and w_{High} are learnable during the end-to-end training of the network.

Secondly, to make this potential stronger, we average initial unary potentials from the classifier (the CNN in our case), across all pixels in the superpixel and use the average as an additional unary potential for those pixels. During experiments, we observed that superpixel based higher order energy helps in getting rid of small spurious regions of wrong labels in the segmentation output, as shown in Fig. 3.4.

3.5 Mean Field Updates and Their Differentials

This section discusses the mean field updates for the higher order potentials previously introduced. These update operations are differentiable with respect to the $Q_i(X_i)$ distribution inputs at each iteration, as well as the parameters of our higher order potentials. This allows us to train our CRF end-to-end as another layer of a neural network.

Take a CRF with random variables $V_1, V_2, \ldots, V_N$ and a set of cliques $\mathcal{C}$, which includes unary, pairwise and higher order cliques. Mean field inference approximates the joint distribution $\Pr(\mathbf{V} = \mathbf{v})$ with the product of marginals $\prod_i Q(V_i = v_i)$. We use $Q(\mathbf{V}_c = \mathbf{v}_c)$ to denote the marginal probability mass for a subset $\{\mathbf{V}_c\}$ of these variables. Where there is no ambiguity, we use the short-hand notation $Q(\mathbf{v}_c)$ to represent $Q(\mathbf{V}_c = \mathbf{v}_c)$. General mean field updates of such a CRF take the form [113, 152]

$$Q^{t+1}(V_i = v) = \frac{1}{Z_i} \exp\left(-\sum_{c \in \mathcal{C}} \sum_{\{\mathbf{v}_c | v_i = v\}} Q^t(\mathbf{v}_{c-i})\, \psi_c(\mathbf{v}_c) \right), \tag{3.5}$$

where Q^t is the marginal after the t^{th} iteration, $\mathbf{v}_c$ an assignment to all variables in clique c, $\mathbf{v}_{c-i}$ an assignment to all variables in c except for V_i, $\psi_c(\mathbf{v}_c)$ is the cost of assigning $\mathbf{v}_c$ to the clique c, and Z_i is the normalisation constant that makes $Q(V_i = v)$ a probability mass function after the update.

3.5.1 Updates from Detection Based Potentials

Following Eq. (3.3) above, we now use Eq. (3.5) to derive the mean field updates related to ψ_d^{Det}. The contribution from ψ_d^{Det} to the update of $Q(X_d^{(i)} = l)$ takes the form

$$\sum_{\{(\mathbf{x}_d, y_d) | x_d^{(i)} = l\}} Q(\mathbf{x}_{d-i}, y_d)\, \psi_d^{\text{Det}}(\mathbf{x}_d, y_d) = \begin{cases} w_{\text{Det}}\, \frac{s_d}{n_d}\, Q(Y_d = 0) & \text{if } l = l_d, \\ w_{\text{Det}}\, \frac{s_d}{n_d}\, Q(Y_d = 1) & \text{otherwise,} \end{cases} \tag{3.6}$$

where $\mathbf{x}_{d-i}$ is an assignment to $\mathbf{X}_d$ with the i^{th} element deleted. Using the same equations, we derive the contribution from the energy ψ_d^{Det} to the update of $Q(Y_d = b)$ to take the

form

$$\sum_{\{(\mathbf{x}_d, y_d)|y_d=b\}} Q(\mathbf{x}_d)\, \psi_d^{\mathrm{Det}}(\mathbf{x}_d, y_d) = \begin{cases} w_{\mathrm{Det}}\, \frac{s_d}{n_d} \sum_{i=1}^{n_d} Q(X_d^{(i)} = l_d) & \text{if } b = 0, \\ w_{\mathrm{Det}}\, \frac{s_d}{n_d} \sum_{i=1}^{n_d} (1 - Q(X_d^{(i)} = l_d)) & \text{otherwise.} \end{cases} \tag{3.7}$$

It is possible to increase the number of parameters in $\psi_d^{\mathrm{Det}}(.)$. Since we use backpropagation to learn these parameters automatically during end-to-end training, it is desirable to have a high number of parameters to increase the flexibility of the model. Following this idea, we made the weight w_{Det} class specific, that is, a function $w_{\mathrm{Det}}(l_d)$ is used instead of w_{Det} in Eqs. (3.2), (3.6) and (3.7). The underlying assumption is that detector outputs can be very helpful for certain classes, while being not so useful for classes that the detector performs poorly on, or classes for which the foreground segmentation is often inaccurate.

Note that due to the presence of detection potentials in the CRF, error differentials calculated with respect to the X variable unary potentials and pairwise parameters will no longer be valid in the forms described in [329]. The error differentials with respect to the X and Y variables, as well as class-specific detection potential weights $w_{\mathrm{Det}}(l)$ are included in the supplementary material.

3.5.2 Updates for Superpixel Based Potentials

The contribution from the P^n-Potts type potential to the mean field update of $Q(x_i = l)$, where pixel i is in the superpixel clique s, was derived in [296] as

$$\sum_{\{\mathbf{x_s}|x_s^{(i)}=l\}} Q(\mathbf{x}_{s-i})\, \psi_s^{\mathrm{SP}}(\mathbf{x}_s) = w_{\mathrm{Low}}(l) \prod_{j \in c, j \neq i} Q(X_j = l) + w_{\mathrm{High}} \left(1 - \prod_{j \in c-i} Q(X_j = l) \right). \tag{3.8}$$

This update operation is differentiable with respect to the parameters $w_{\mathrm{Low}}(l)$ and w_{High}, allowing us to optimise them via backpropagation, and also with respect to the $Q(X)$ values enabling us to optimise previous layers in the network.

3.5.3 Convergence of parallel mean field updates

Mean field with parallel updates, as proposed in [154] for speed, does not have any convergence guarantees in the general case. However, we usually empirically observed convergence with higher order potentials, without damping the mean field update as described in [296, 17]. This may be explained by the fact that the unaries from the initial pixelwise-prediction part of our network provide a good initialisation. In cases where the mean field energy did not converge, we still empirically observed good final segmentations.

3.6 Experiments

We evaluate our new CRF formulation on two different datasets using the CRF-RNN network [329] as the main baseline, since we are essentially enriching the CRF model of [329]. We then present ablation studies on our models.

3.6.1 Experimental set-up and results

Our deep network consists of two conceptually different, but jointly trained stages. The first, "unary" part of our network is formed by the FCN-8s architecture [189]. It is initialised from the Imagenet-trained VGG-16 network [267], and then fine-tuned with data from the VOC 2012 training set [81], extra VOC annotations of [110] and the MS COCO [180] dataset.

The output of the first stage is fed into our CRF inference network. This is implemented using the mean field update operations and their differentials described in Section 3.5. Five iterations of mean field inference were performed during training. Our CRF network has two additional inputs in addition to segmentation unaries obtained from the FCN-8s network: data from the object detector and superpixel oversegmentations of the image.

We used the publicly available code and model of the Faster R-CNN [243] object detector. The fully automated version of GrabCut [252] was then used to obtain foregrounds from the detection bounding boxes. These choices were made after conducting preliminary experiments with alternate detection and foreground segmentation algorithms.

Four levels of superpixel oversegmentations were used, with increasing superpixel size to define the cliques used in this potential. Four levels were used since performance on the VOC validation set stopped increasing after this number. We used the superpixel method of [89] as it was shown to adhere to object boundaries the best [1], but our method generalises to any oversegmentation algorithm.

We trained the full network end-to-end, optimising the weights of the CNN classifier (FCN-8s) and CRF parameters jointly. We initialised our network using the publicly available weights of [329], and trained with a learning rate of 10^{-10} and momentum of 0.99. The learning rate is low because the loss was not normalised by the number of pixels in the training image. This is to have a larger loss for images with more pixels. When training our CRF, we only used VOC 2012 data [81] as it has the most accurate labelling, particularly around boundaries.

3.6.1.1 PASCAL VOC 2012 Dataset

The improvement obtained by each higher order potential was evaluated on the same reduced validation set [189] used by our baseline [329]. As Table 3.1 shows, each new higher

Table 3.1: Comparison of each higher order potential with respect to our baseline on the VOC 2012 reduced validation set.

Method	Reduced val set(%)
Baseline (unary + pairwise)[329]	72.9
Superpixels only	74.0
Detections only	74.9
Detections and Superpixels	75.8

Table 3.2: Mean IoU accuracy on VOC 2012 test set. All methods are trained with MS COCO [180] data

Method	Test set(%)
Ours	**77.9**
DPN[188]	77.5
Centrale Super Boundaries[149]	75.7
Dilated Convolutions[319]	75.3
BoxSup[63]	75.2
DeepLab Attention[47]	75.1
CRF-RNN (baseline) [329]	74.7
DeepLab WSSL[221]	73.9
DeepLab[44]	72.7

Table 3.3: Mean Intersection over Union (IoU) results on PASCAL Context validation set compared to other current methods.

Method	**Ours**	BoxSup[63]	ParseNet[186]	CRF-RNN [329]	FCN-8s[189]	CFM[64]
Mean IoU (%)	**41.3**	40.5	40.4	39.3	37.8	34.4

order potential improves the mean IoU over the baseline. We only report test set results for our best method since the VOC guidelines discourage the use of the test set for ablation studies. On the test set (Table 3.2), we outperform our baseline by 3.2% which equates to a 12.6% reduction in the error rate. This sets a new state-of-the-art on the VOC dataset. Qualitative results highlighting success and failure cases of our algorithm, as well as more detailed results, are shown in our supplementary material.

3.6.1.2 PASCAL Context

Table 3.3 shows our state-of-the-art results on the recently released PASCAL Context dataset [209]. We trained on the provided training set of 4998 images, and evaluated on the validation set of 5105 images. This dataset augments VOC with annotations for all objects in the scene. As a result, there are 59 classes as opposed to the 20 in the VOC dataset. Many of these new labels are "stuff" classes such as "grass" and "sky". Our object detectors are therefore only trained for 20 of the 59 labels in this dataset. Nevertheless, we improve by 0.8% over the previous state-of-the-art [63] and 2% over our baseline [329].

3.6.2 Ablation Studies

We perform additional experiments to determine the errors made by our system, show the benefits of end-to-end training and compare our detection potentials to a simpler

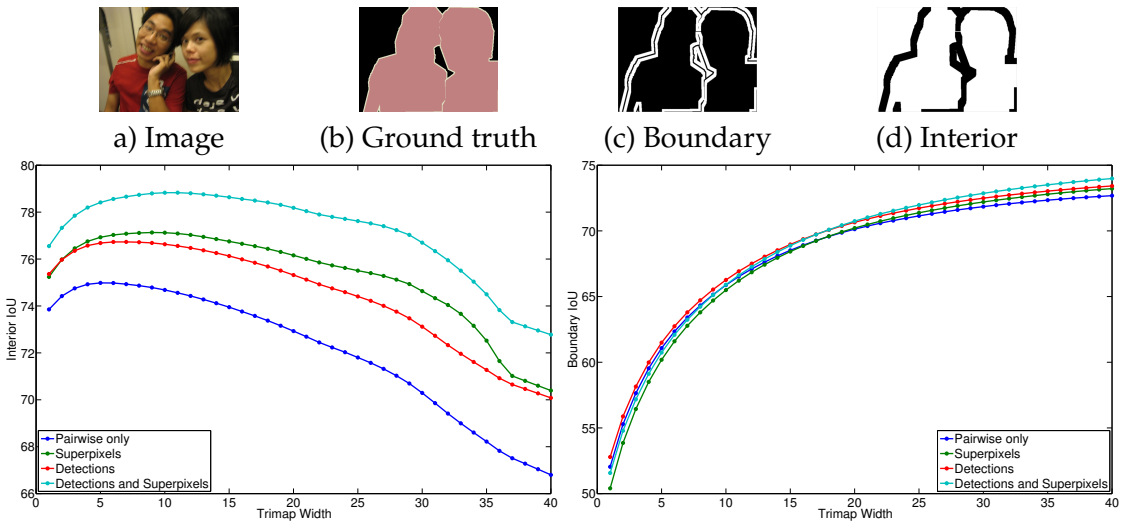


Figure 3.5: Error analysis on VOC 2012 reduced validation set. The IoU is computed for boundary and interior regions for various trimap widths. An example of the Boundary and Interior regions for a sample image using a width of 9 pixels is shown in white in the top row. Black regions are ignored in the IoU calculation.

baseline. Unless otherwise stated, these experiments are performed on the VOC 2012 reduced validation set.

3.6.2.1 Error Analysis

To analyse the improvements made by our higher order potentials, we separately evaluate the performance on the "boundary" and "interior" regions in a similar manner to [63]. As shown in Fig. 3.5 c) and d), we consider a narrow band (trimap [148]) around the "void" labels annotated in the VOC 2012 reduced validation set. The mean IoU of pixels lying within this band is termed the "Boundary IoU" whilst the "Interior IoU" is evaluated outside this region.

Figure 3.5 shows our results as the trimap width is varied. Adding the detection potentials improves the Interior IoU over our baseline (only pairwise potentials [329]) as the object detector may recognise objects in the image which the pixelwise classification stage of our network may have missed out. However, the detection potentials also improve the Boundary IoU for all tested trimap widths as well. Improving the recognition of pixels in the interior of an object also helps with delineating the boundaries since the strength of the pairwise potentials exerted by the Q distributions at each of the correctly-detected pixels increase.

Our superpixel priors also increase the Interior IoU with respect to the baseline. Encouraging consistency over regions helps to get rid of spurious regions of wrong labels

Table 3.4: Comparison of mean IoU (%) obtained on VOC 2012 reduced validation set from end-to-end and piecewise training.

Method	FCN-8s	DCN
Unary only, fine-tuned on COCO	68.3	68.6
Pairwise CRF trained piecewise	69.5	70.7
Pairwise CRF trained end-to-end	72.9	72.5
Higher Order CRF trained piecewise	73.6	73.5
Higher Order CRF trained end-to-end	75.8	75.0
Test set performance of best model	77.9	76.9

(as shown in Fig. 3.4). Fig. 3.5 suggests that most of this improvement occurs in the interior of an object. The Boundary IoU is slightly lower than the baseline, and this may be due to the fact that superpixels do not always align correctly with the edges of an object (the "boundary recall" of various superpixel methods are evaluated in [1]).

We can see that the combination of detection and superpixel potentials results in a substantial improvement in our Interior IoU. This is the primary reason our overall IoU on the VOC benchmark increases with higher order potentials.

3.6.2.2 Benefits of end-to-end training

Table 3.4 shows how end-to-end training outperforms piecewise training. We trained the CRF piecewise by freezing the weights of the unary part of the network, and only learning the CRF parameters.

Our results in Table 3.2 used the FCN-8s [189] architecture to generate unaries. To show that our higher order potentials improve performance regardless of the underlying CNN used for producing unaries, we also perform an experiment using our reimplementation of the "front-end" module proposed in the Dilated Convolution Network (DCN) of [319] instead of FCN-8s.

Table 3.4 shows that end-to-end training of the CRF yields considerable improvements over piecewise training. This was the case when using either FCN-8s or DCN for obtaining the initial unaries before performing CRF inference with higher order potentials. This suggests that our CRF network module can be plugged into different architectures and achieve performance improvements.

3.6.2.3 Baseline for detections

To evaluate the efficacy of our detection potentials, we formulate a simpler baseline since no other methods use detection information at inference time (BoxSup [63] derives ground truth for training using ground-truth bounding boxes).

Our baseline is similar to CRF-RNN [329], but prior to CRF inference, we take the segmentation mask from the object detection and add a unary potential proportional to the detector's confidence to the unary potentials for those pixels. We then perform mean-field inference (with only pairwise terms) on these "augmented" unaries. Using this method, the mean IoU increases from 72.9% to 73.6%, which is significantly less than the 74.9% which we obtained using only our detection potentials without superpixels (Table 3.1).

Our detection potentials perform better since our latent Y detection variables model whether the detection hypothesis is accepted or not. Our CRF inference is able to evaluate object detection inputs in light of other potentials. Inference increases the relative score of detections which agree with the segmentation, and decreases the score of detections which do not agree with other energies in the CRF. Figures 3.2 b) and d) show examples of false-positive detections that have been ignored and correct detections that have been used to refine our segmentation. Our baseline, on the other hand, is far more sensitive to erroneous detections as it cannot adjust the weight given to them during inference.

3.7 Conclusion

We presented a CRF model with two different higher order potentials to tackle the semantic segmentation problem. The first potential is based on the intuitive idea that object detection can provide useful cues for semantic segmentation. Our formulation is capable of automatically rejecting false object detections that do not agree at all with the semantic segmentation. Secondly, we used a potential that encourages superpixels to have consistent labelling. These two new potentials can co-exist with the usual unary and pairwise potentials in a CRF.

Importantly, we showed that efficient mean field inference is still possible in the presence of the new higher order potentials and derived the explicit forms of the mean field updates and their differentials. This enabled us to implement the new CRF model as a stack of CNN layers and to train it end-to-end in a unified deep network with a pixelwise CNN classifier. We experimentally showed that the addition of higher order potentials results in a significant increase in semantic segmentation accuracy allowing us to reach state-of-the-art performance.

Appendices

3.A Derivatives of Mean Field Updates

The pseudocode for the mean field inference algorithm with latent Y detection variables is shown below in Algorithm 1. We use the same notation used in the main paper.

Algorithm 1 Mean Field Inference

$Q^0(X_i = l) \leftarrow \frac{1}{Z_i} \exp\left(-\psi_i^U(l)\right), \quad \forall i, l$

$Q^0(Y_d = b) \leftarrow s_d^b (1 - s_d)^{(1-b)}, \quad \forall d, b$

▷ Initialisation

for t = 0 : $T - 1$ **do**

$E^t(X_i = l) \leftarrow$ UnaryUpdate + PairwiseUpdate + DetectionUpdate + SuperpixelUpdate, $\quad \forall i, l$

$E^t(Y_d = b) \leftarrow$ Y_UnaryUpdate + Y_DetectionUpdate

▷ Mean field updates

$Q^{t+1}(X_i = l) \leftarrow \frac{1}{Z_i} \exp\left(-E^t(X_i = l)\right), \quad \forall i, l$

$Q^{t+1}(Y_d = b) \leftarrow \frac{1}{Z_d} \exp\left(-E^t(Y_d = b)\right), \quad \forall d, b$

▷ Normalising

end for

For the explicit forms of the $\mathrm{UnaryUpdate}$ and $\mathrm{PairwiseUpdate}$ above, and their differentials, we refer the reader to [329] and discuss the terms $\mathrm{DetectionUpdate}$ and $\mathrm{SuperpixelUpdate}$ in detail below.

Let us assume that only one object detection of the form (l_d, s_d, F_d) is available for the image under consideration. When multiple detections are present, simply a summation of the updates and differentials discussed below apply. Therefore, no generality is lost with this assumption. Similarly, we can assume that only one superpixel clique $\{\mathbf{X}_s\}$ is present, without a loss of generality.

Assuming that pixel i in Algorithm 1 belongs to F_d, Eq. (3.6) in the main paper described the exact form of $\mathrm{DetectionUpdate}$. Similarly, assuming that pixel i belongs to $\{X_s\}$ Eq. (3.8) described the form of $\mathrm{SuperpixelUpdate}$.

Let L denote the value of the loss function calculated at the output of the deep network. This could be the softmax loss or any other appropriate loss function. During backpropagation, we get the error signal $\frac{\partial L}{\partial Q^T}$ at the output of the mean field inference. Using this error information, we need to compute the derivative of the loss L with respect to the X unaries and various CRF parameters. Note that, if we compute the relevent differentials for only one iteration of the mean field algorithm, it is possible to calculate them for multiple iterations using the recurrent behaviour of the iterations.

Note that, by looking at *Normalising* step of Algorithm 1, it is trivial to calculate $\frac{\partial Q^{t+1}}{\partial E^t}$. Therefore, we can then calculate $\frac{\partial L}{\partial E^t}$ using the chain rule. This is same as backpropagation

of the usual softmax operation in a deep network (up to a negative sign). Using this observation we can calculate the necessary differentials to take the forms shown below:

$$\begin{aligned}\frac{\partial L}{\partial w_{\text{Det}}} = \frac{s_d}{n_d}\sum_{i=1}^{n_d}\Big(&\frac{\partial L}{E^t(X_d^{(i)}=l_d)}Q^t(Y_d=1) \;+\\ &\sum_{l'\neq l_d}\frac{\partial L}{\partial E^t(X_d^{(i)}=l')}Q^t(Y_d=1)\Big) \;+\\ &\frac{\partial L}{\partial E^t(Y_d=0)}\frac{s_d}{n_d}\sum_{i=1}^{n_d}Q^t(X_d^{(i)}=l_d) \;+\\ &\frac{\partial L}{\partial E^t(Y_d=1)}\frac{s_d}{n_d}\sum_{i=1}^{n_d}\Big(1-Q^t(X_d^{(i)}=l_d)\Big)\end{aligned} \tag{3.9}$$

$$\frac{\partial L}{\partial Q^t(X_d^{(i)}=l_d)} = w_{\text{Det}}\frac{\partial L}{\partial E^t(Y_d=0)} - w_{\text{Det}}\frac{\partial L}{\partial E^t(Y_d=1)} \tag{3.10}$$

$$\frac{\partial L}{\partial Q^t(Y_d=0)} = w_{\text{Det}}\frac{s_d}{n_d}\sum_{i=1}^{n_d}\left(\frac{\partial L}{E^t(X_d^{(i)}=l_d)}\right) \tag{3.11}$$

$$\frac{\partial L}{\partial Q^t(Y_d=1)} = w_{\text{Det}}\frac{s_d}{n_d}\sum_{i=1}^{n_d}\sum_{l\neq l_d}\left(\frac{\partial L}{\partial E^t(X_d^{(i)}=l')}\right) \tag{3.12}$$

$$\frac{\partial L}{\partial w_{\text{Low}}(l)} = \sum_{i\in s}\left(\frac{\partial L}{\partial E^t(X_s^{(i)}=l)}\prod_{j\in c, j\neq i}Q^t(X_j=l)\right) \tag{3.13}$$

$$\frac{\partial L}{\partial w_{\text{High}}} = \sum_{i\in s}\sum_{l\in\mathcal{L}}\left(\frac{\partial L}{\partial E^t(X_s^{(i)}=l)}\left(1-\prod_{j\in c, j\neq i}Q^t(X_j=l)\right)\right) \tag{3.14}$$

Effect of the superpixel potentials on the derivatives $\frac{\partial L}{\partial Q^t(X_i=l)}$ were negligible. Therefore, we ignored them in our calculations.

3.B Additional Experimental Results

Evaluation of the rescoring of detections

As mentioned in Sec. 3.4.1, the unary potentials of the latent Y detection variables in our CRF are obtained from the confidence score of the object detector, and are then updated

Table 3.5: Comparison between the adjust detection scores as a result of CRF inference and original detection scores

	Faster RCNN	Faster RCNN with rescored confidences
Mean Average Precision (%)	64.3	64.6

during mean-field inference. We view the final value of the Y variables after inference as the rescored or calibrated confidence value of the object detector.

We performed semantic segmentation on the images in the test set of the Pascal VOC 2012 detection challenge using initial bounding boxes from Faster R-CNN [243]. Although our network does not change the bounding box predictions of the detector, it does adjust the confidence scores. As shown in Tab. 3.5, we observe a slight improvement of 0.3% in the mean average precision when using our recalibrated scores.

This suggests that our CRF inference is able to evaluate object detection inputs in light of other potentials (unary, pairwise, and superpixels). Inference increases the relative score of detections which agree with the segmentation, and decreases the score of detections that do not agree with other energies in the CRF. Figures 3.2b) and d) show examples of false positive detection that have been ignored and correct detections that have been used to refine our segmentation.

Note that we used the publicly available version (code and model) of Faster R-CNN. We applied non-maximal suppression and thresholded the detections such that detections with a score lower than 0.6 (out of 1) were not used for CRF inference.

Additional quantitative results

Table 3.6 presents more detailed results of our method, and that of other state-of-the-art techniques, on the PASCAL VOC 2012 test set. In particular, we present the accuracy for every class in the VOC test set. Note that our per-class accuracy improves over our baseline, CRF-RNN [329], for all of the 20 classes in PASCAL VOC.

Additional qualitative results

Figure 3.6 shows more sample results of our system, compared to our baseline, CRF-as-RNN [329]. Figure 3.7 shows examples of failure cases of our method. Figure 3.8 examines the effect of each of our potentials. Finally, Figure 3.9 shows a qualitative comparison between the output of our system and other current methods on the PASCAL VOC 2012 test set.

Table 3.6: Comparison of the mean Intersection over Union (IoU) accuracy of our approach and other state-of-the-art methods on the Pascal VOC 2012 test set. Scores for other methods were taken from the original authors' publications.

Methods trained with COCO	Mean IoU(%)	aero-plane	bike	bird	boat	bot-tle	bus	car	cat	chair	cow	ta-ble	dog	horse	mbike	per-son	plant	sheep	sofa	train	tv
Our method	**77.9**	**92.5**	59.1	**90.3**	**70.6**	74.4	92.4	84.1	88.3	**36.8**	85.6	**67.1**	85.1	86.9	**88.2**	82.6	62.6	85.0	56.2	81.9	**72.5**
DPN [188]	77.5	89.0	**61.6**	87.7	66.8	**74.7**	91.2	84.3	87.6	36.5	**86.3**	66.1	84.4	87.8	85.6	85.4	**63.6**	87.3	**61.3**	79.4	66.4
Super Bound.[149]	75.7	90.3	37.9	89.6	67.8	74.6	89.3	84.1	89.1	35.8	83.6	66.2	82.9	81.7	85.6	84.6	60.3	84.8	60.7	78.3	68.3
Dilated Conv. [319]	75.3	91.7	39.6	87.8	63.1	71.8	89.7	82.9	89.8	37.2	84.0	63.0	83.3	**89.0**	83.8	85.1	56.8	**87.6**	56.0	80.2	64.7
BoxSup [63]	75.2	89.8	38.0	89.2	68.9	68.0	89.6	83.0	87.7	34.4	83.6	**67.1**	81.5	83.7	85.2	83.5	58.6	84.9	55.8	81.2	70.7
Attention [47]	75.1	92.0	41.2	87.8	57.2	72.7	**92.8**	**85.9**	**90.5**	30.5	78.0	62.8	**85.8**	85.3	87.2	**85.6**	57.7	85.1	56.5	**83.0**	65.0
CRF-as-RNN [329]	74.7	90.4	55.3	88.7	68.4	69.8	88.3	82.4	85.1	32.6	78.5	64.4	79.6	81.9	86.4	81.8	58.6	82.4	53.5	77.4	70.1
WSSL [221]	73.9	89.2	46.7	88.5	63.5	68.4	87.0	81.2	86.3	32.6	80.7	62.4	81.0	81.3	84.3	82.1	56.2	84.6	58.2	76.2	67.2
DeepLab [44]	72.7	89.1	38.3	88.1	63.3	69.7	87.1	83.1	85.0	29.3	76.5	56.5	79.8	77.9	85.8	82.4	57.4	84.3	54.9	80.5	64.1
Methods trained without COCO																					
Our method	73.9	89.3	40.0	81.6	65.1	71.7	90.1	81.3	85.7	32.4	82.1	62.2	82.6	83.7	84.5	81.1	60.8	85.2	49.6	80.0	69.9
DPN [188]	74.1	87.7	59.4	78.4	64.9	70.3	89.3	83.5	86.1	31.7	79.9	62.6	81.9	80.0	83.5	82.3	60.5	83.2	53.4	77.9	65.0
DeconvNet [215]	72.5	89.9	39.3	79.7	63.9	68.2	87.4	81.2	86.1	28.5	77.0	62.0	79.0	80.3	83.6	80.2	58.8	83.4	54.3	80.7	65.0
CRF-as-RNN [329]	72.0	87.5	39.0	79.7	64.2	68.3	87.6	80.8	84.4	30.4	78.2	60.4	80.5	77.8	83.1	80.6	59.5	82.8	47.8	78.3	67.1
DeepLab [44]	71.6	84.4	54.5	81.5	63.6	65.9	85.1	79.1	83.4	30.7	74.1	59.8	79.0	76.1	83.2	80.8	59.7	82.2	50.4	73.1	63.7
Piecewise [178]	70.7	87.5	37.7	75.8	57.4	72.3	88.4	82.6	80.0	33.4	71.5	55.0	79.3	78.4	81.3	82.7	56.1	79.8	48.6	77.1	66.3
Zoomout [208]	69.6	85.6	37.3	83.2	62.5	66.0	85.1	80.7	84.9	27.2	73.2	57.5	78.1	79.2	81.1	77.1	53.6	74.0	49.2	71.7	63.3
FCN-8s [189]	62.2	76.8	34.2	68.9	49.4	60.3	75.3	74.7	77.6	21.4	62.5	46.8	71.8	63.9	76.5	73.9	45.2	72.4	37.4	70.9	55.1
CFM [64]	61.8	75.7	26.7	69.5	48.8	65.6	81.0	69.2	73.3	30.0	68.7	51.5	69.1	68.1	71.7	67.5	50.4	66.5	44.4	58.9	53.5
NUS_UDS [74]	50.0	67.0	24.5	47.2	45.0	47.9	65.3	60.6	58.5	15.5	50.8	37.4	45.8	59.9	62.0	52.7	40.8	48.2	36.8	53.1	45.6
O2P [37]	47.8	64.0	27.3	54.1	39.2	48.7	56.6	57.7	52.5	14.2	54.8	29.6	42.2	58.0	54.8	50.2	36.6	58.6	31.6	48.4	38.6

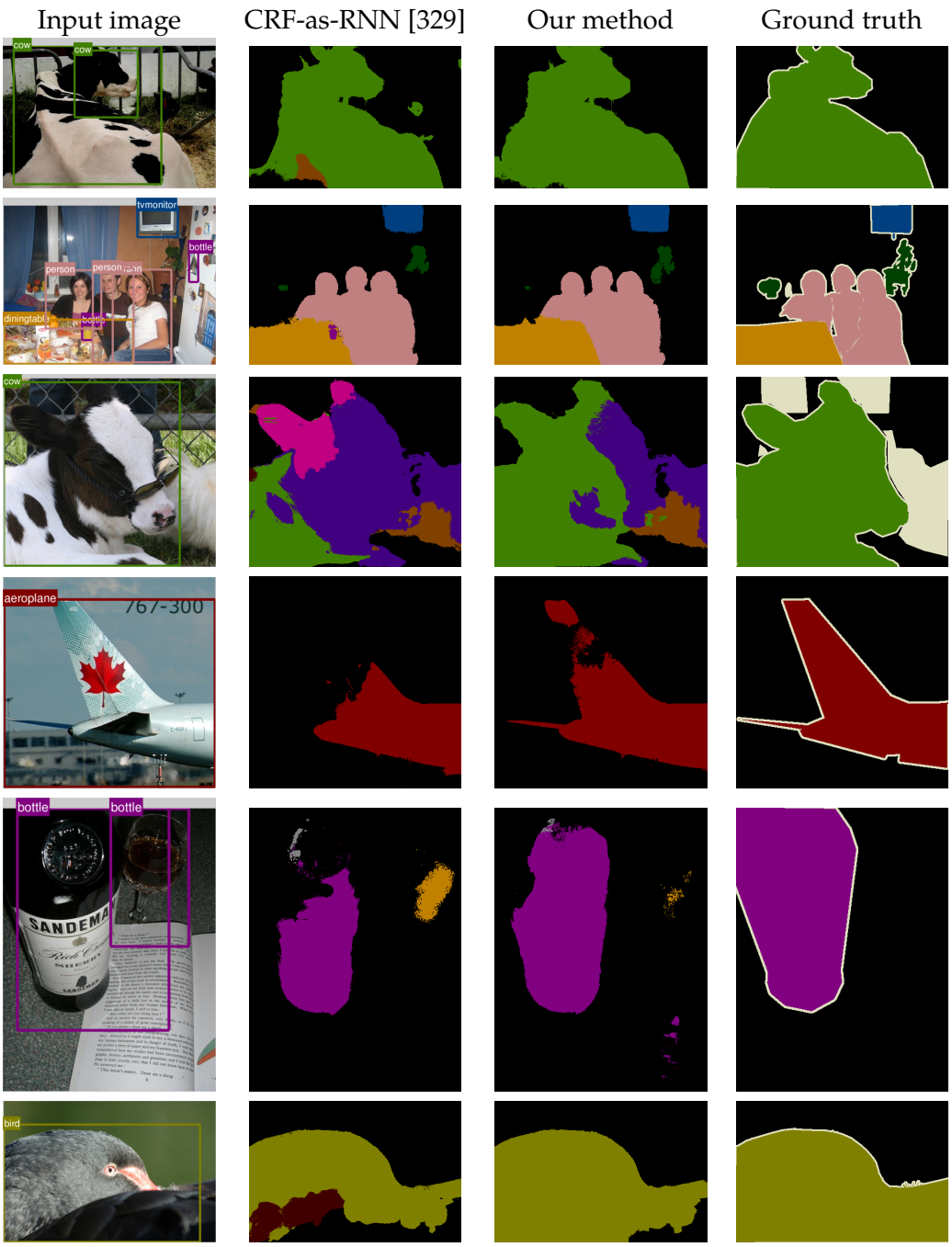


Figure 3.6: Examples of images where our method has improved over our baseline, CRF-as-RNN [329]. The input images have the detection bounding boxes overlaid on them. Note that the method of [329] does not make use of this information. The improvements from our method are due to our detection potentials, as well as our superpixel based potentials. Note that all images are from the reduced validation set of VOC 2012 and have not been trained on at all.

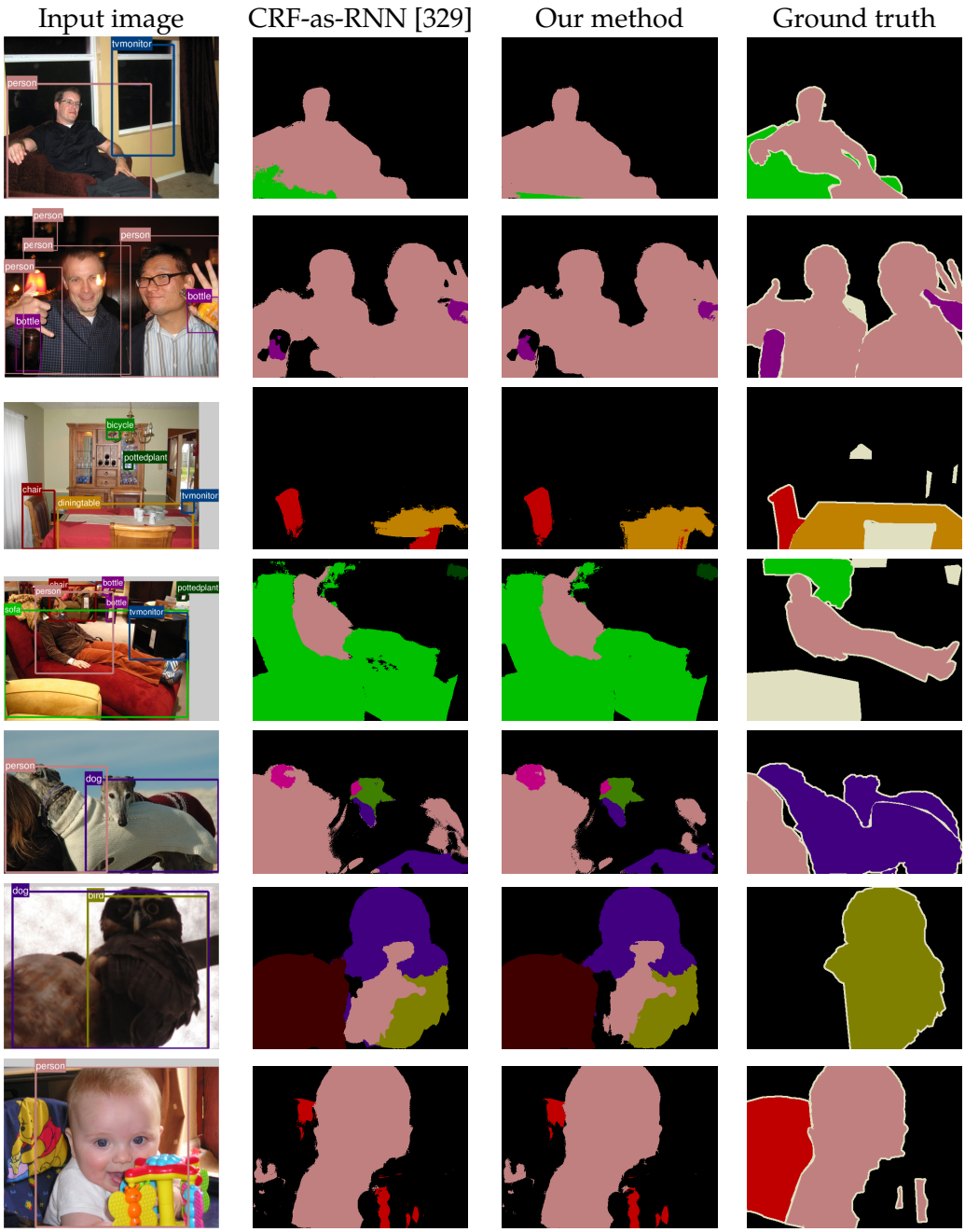


Figure 3.7: Examples of failure cases where our method has performed poorly. The first row shows an example of how the detection of the person has now resulted in the sofa being misclassified (although our system is able to reject the other false detection). Our superpixel potentials have a tendency to remove spurious noise by enforcing consistency within regions. However, as shown in the second row, sometimes the "noise" being removed is actually the correct label. In the other cases, we are limited by our pixelwise classification unaries which are poor. Our superpixel and detection potentials are not always able to compensate for this. Note that all images are from the reduced validation set of VOC 2012 and have not been trained on at all. The input images have the detection bounding boxes overlaid on them. Note that the method of [329] does not make use of this information.

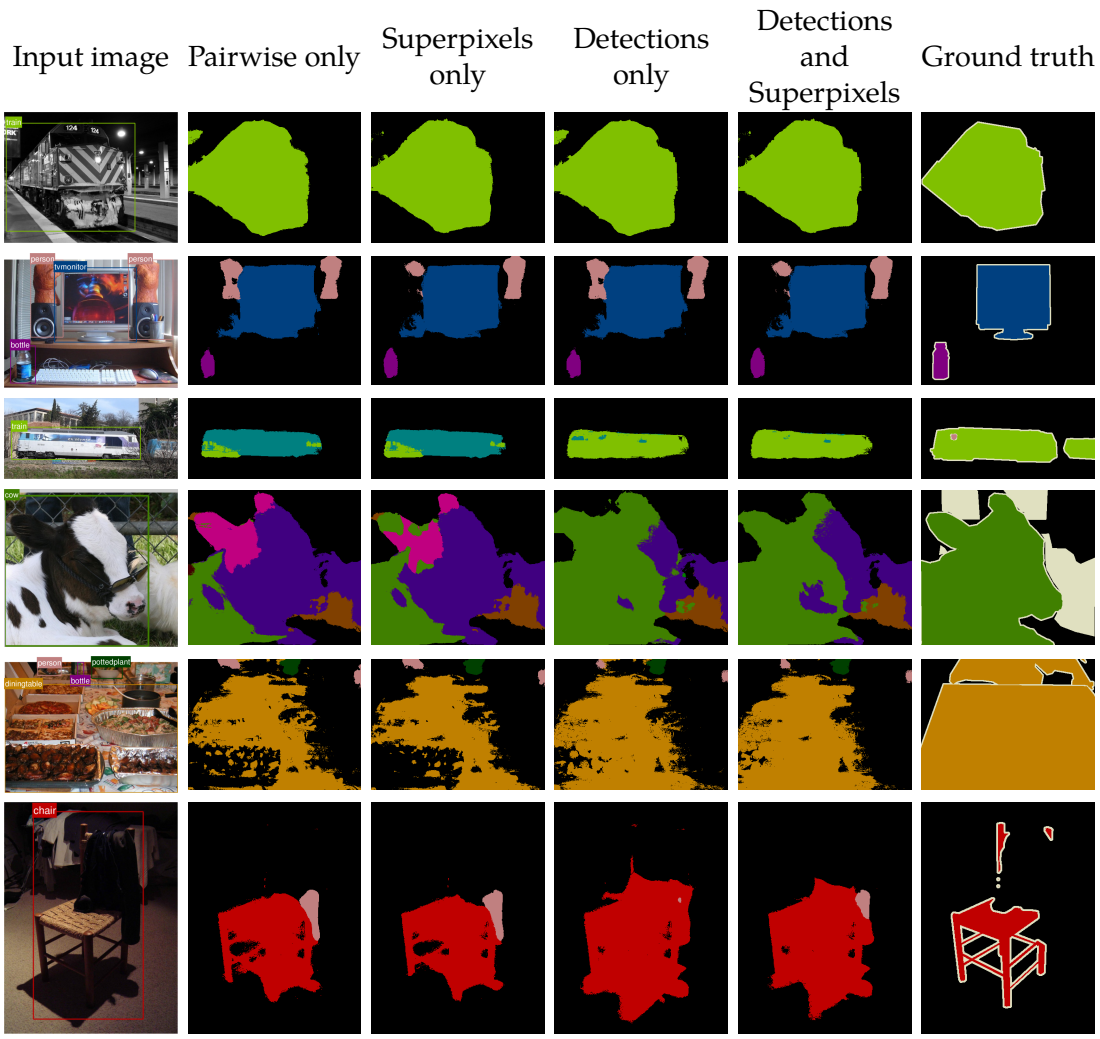


Figure 3.8: Comparison of pairwise potentials, superpixel and pairwise potentials, detection and pairwise potentials, and a combination of all three. (Row 1 and 2) These are examples where superpixel potentials help to remove spurious noise in the output but detection potentials do not affect the result. The final result still improves when all potentials are combined. (Row 3) Detection potentials greatly improve the result by recognising the train correctly (the pixelwise unaries are largest for "bus"). And superpixels, when combined with detections, slightly improve the output. (Row 4) An example where both superpixel and detection potentials improve the final output. (Row 5) A case where the superpixel worsens the result as, although the output is more consistent among superpixel regions, some pixels have had their correct labels removed. However, the correct detection improves the result, and the output of combining superpixel and detection potentials is actually better than either potential in isolation. (Row 6) Here, the detection (although correct) worsens the output due to its imprecise foreground mask. Superpixel potentials also exacerbate the result, since the legs of the chair and the chair's shadow are confused to be part of the same superpixel region. However, when the two potentials are combined, the result is slightly better than with only detection potentials.

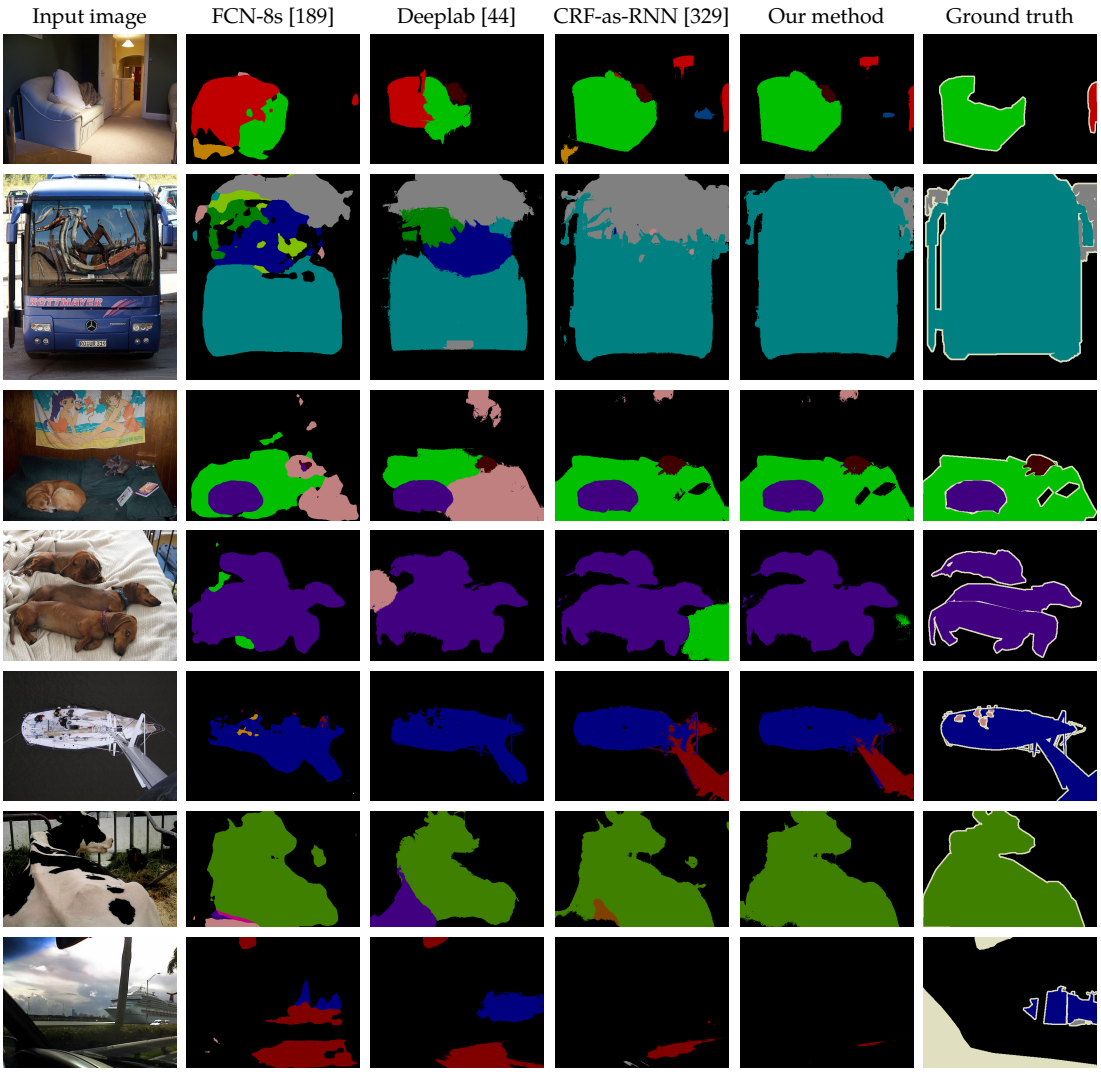


Figure 3.9: Qualitative comparison with other current methods. Sample results of our method compared to other current techniques on VOC 2012. We reproduced the segmentation results of Deeplab from their original publication, whilst we reproduced the results of FCN-8s and CRF-as-RNN from their publicly-available source code.

For completeness, we show a comparison of all the potentials described in our main paper (pairwise, superpixels and detections), for all the images shown in Figures 3.6 and 3.7.

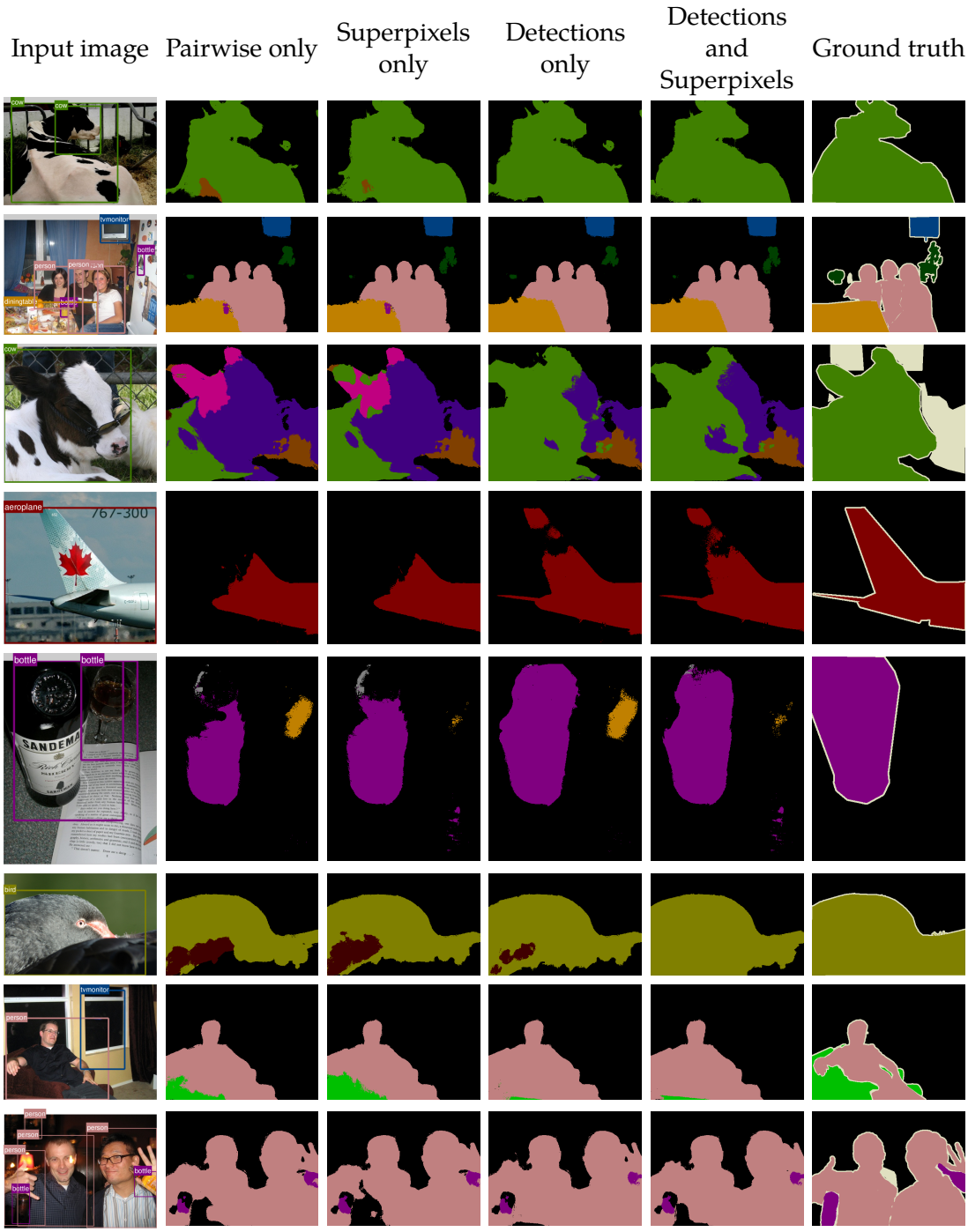


Figure 3.10: Comparison of all potentials on images shown in Figures 3.6 and 3.7

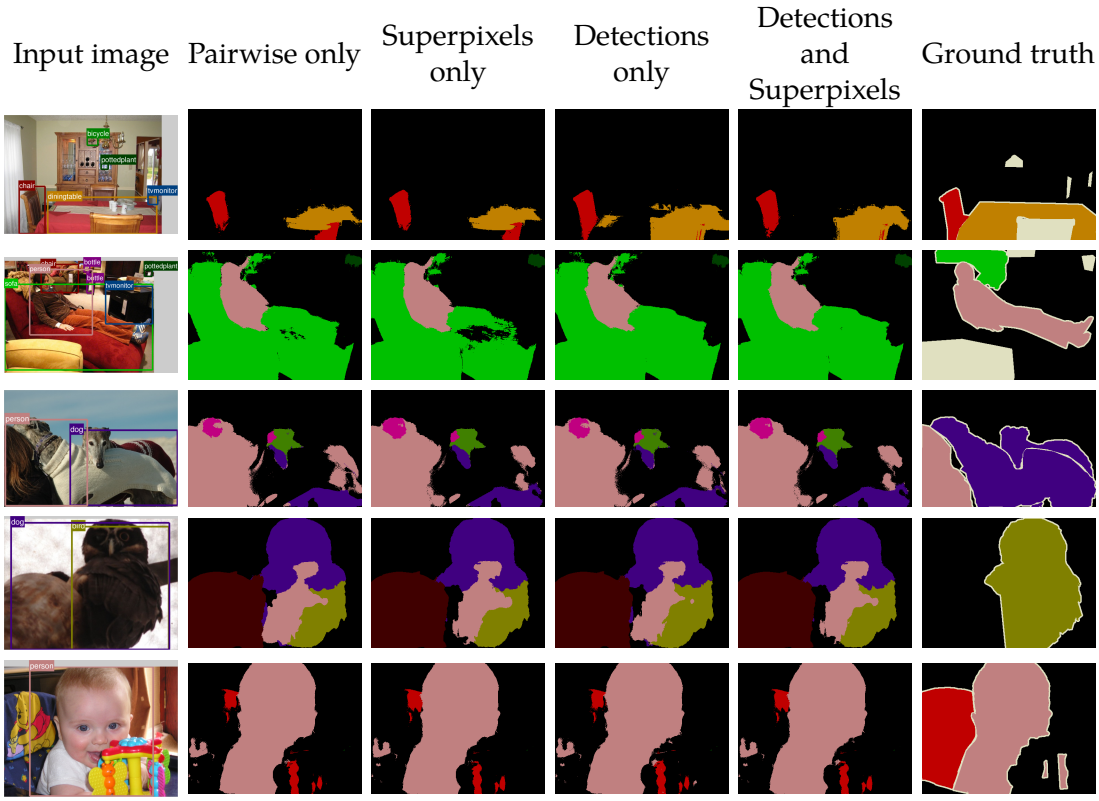


Fig. 3.10 continued: Comparison of all potentials on images shown in Figures 3.6 and 3.7

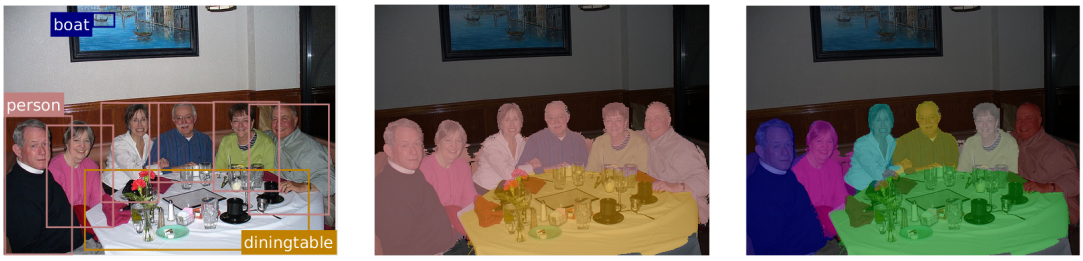
Chapter 4

Pixelwise Instance Segmentation with a Dynamically Instantiated Network

Semantic segmentation and object detection research have recently achieved rapid progress. However, the former task has no notion of different instances of the same object, and the latter operates at a coarse, bounding-box level. We propose an Instance Segmentation system that produces a segmentation map where each pixel is assigned an object class and instance identity label. Most approaches adapt object detectors to produce segments instead of boxes. In contrast, our method is based on an initial semantic segmentation module, which feeds into an instance subnetwork. This subnetwork uses the initial category-level segmentation, along with cues from the output of an object detector, within an end-to-end CRF to predict instances. This part of our model is dynamically instantiated to produce a variable number of instances per image. Our end-to-end approach requires no post-processing and considers the image holistically, instead of processing independent proposals. Therefore, unlike some related work, a pixel cannot belong to multiple instances. Furthermore, far more precise segmentations are achieved, as shown by our substantial improvements at high AP^r thresholds.

4.1 Introduction

Semantic segmentation and object detection are well-studied scene understanding problems, and have recently witnessed great progress due to deep learning [117, 66, 43]. However, semantic segmentation – which labels every pixel in an image with its object class – has no notion of different instances of an object (Fig. 4.1). Object detection does localise different object instances, but does so at a very coarse, bounding-box level. Instance segmentation localises objects at a pixel level, as shown in Fig. 4.1, and can be thought of being at the intersection of these two scene understanding tasks. Unlike the former, it knows about different instances of the same object, and unlike the latter, it operates at a pixel



(a) Object Detection (b) Semantic Segmentation (c) Instance Segmentation

Figure 4.1: Object detection (a) localises the different people, but at a coarse, bounding-box level. Semantic segmentation (b) labels every pixel, but has no notion of instances. Instance segmentation (c) labels each pixel of each person uniquely. Our proposed method jointly produces both semantic and instance segmentations. Our method uses the output of an object detector as a cue to identify instances, but is robust to false positive detections, poor bounding box localisation and occlusions. Best viewed in colour.

level. Accurate recognition and localisation of objects enables many applications, such as autonomous driving [57], image-editing [312] and robotics [108].

Many recent approaches to instance segmentation are based on object detection pipelines where objects are first localised with bounding boxes. Thereafter, each bounding box is refined into a segmentation [112, 111, 174, 184, 169]. Another related approach [65, 321] is to use segment-based region proposals [62, 232, 233] instead of box-based proposals. However, these methods do not consider the entire image, but rather independent proposals. As a result, occlusions between different objects are not handled. Furthermore, many of these methods cannot easily produce segmentation maps of the image, as shown in Fig. 4.1, since they process numerous proposals independently. There are typically far more proposals than actual objects in the image, and these proposals can overlap and be assigned different class labels. Finally, as these methods are based on an initial detection step, they cannot recover from false detections.

Our proposed method is inspired by the fact that instance segmentation can be viewed as a more complex form of semantic segmentation, since we are not only required to label the object class of each pixel, but also its instance identity. We produce a pixelwise segmentation of the image, where each pixel is assigned both a semantic class and instance label. Our end-to-end trained network, which outputs a variable number of instances per input image, begins with an initial semantic segmentation module. The following, dynamic part of the network, then uses information from an object detector and a Conditional Random Field (CRF) model to distinguish different instances. This approach is robust to false-positive detections, as well as poorly localised bounding boxes which do not cover the entire object, in contrast to detection-based methods to instance segmentation. Moreover, as it considers the

entire image when making predictions, it attempts to resolve occlusions between different objects and can produce segmentation maps as in Fig. 4.1 without any post-processing.

Furthermore, we note that the Average Precision (AP) metric [81] used in evaluating object detection systems, and its AP^r variant [112] used for instance segmentation, considers individual, potentially overlapping, object predictions in isolation, as opposed to the entire image. To evaluate methods such as ours, which produce complete segmentation maps and reason about occlusions, we also evaluate using the "Matching Intersection over Union" metric.

Our system, which is based on an initial semantic segmentation subnetwork, produces sharp and accurate instance segmentations. This is reflected by the substantial improvements we achieve over state-of-the-art methods at high AP^r thresholds on the Pascal VOC and Semantic Boundaries datasets. Furthermore, our network improves on the semantic segmentation task while being trained for the related task of instance segmentation.

4.2 Related Work

An early work on instance segmentation was by Winn and Shotton [305]. A per-pixel unary classifier was trained to predict parts of an object. These parts were then encouraged to maintain a spatial ordering, that is characteristic of an instance, using asymmetric pairwise potentials in a Conditional Random Field (CRF). Subsequent work [317], presented another approach where detection outputs of DPM [88], with associated foreground masks, were assigned a depth ordering using a generative, probabilistic model. This depth ordering resolved occlusions.

However, instance segmentation has become more common after the "Simultaneous Detection and Segmentation" (SDS) work of Hariharan *et al.* [112]. This system was based on the R-CNN pipeline [100]: Region proposals, generated by the method of [5], were classified into object categories with a Convolutional Neural Network (CNN) before applying bounding-box regression as post-processing. A class-specific segmentation was then performed in this bounding box to simultaneously detect and segment the object. Numerous works [111, 50, 169] have extended this pipeline. However, approaches that segment instances by refining detections [112, 111, 50, 64, 169] are inherently limited by the quality of the initial proposals. This problem is exacerbated by the fact that this pipeline consists of several different modules trained with different objective functions. Furthermore, numerous post-processing steps such as "superpixel projection" and rescoring are performed. Dai *et al.* [65] addressed some of these issues by designing one end-to-end trained network that generates box-proposals, creates foreground masks from these

proposals and then classifies these masks. This network can be seen as an extension of the end-to-end Faster-RCNN [243] detection framework, which generates box-proposals and classifies them. Additionally, Liu *et al.* [184] formulated an end-to-end version of the SDS network [112], whilst [174] iteratively refined object proposals.

On a separate track, algorithms have also been developed that do not require object detectors. Zhang *et al.* [325, 326] segmented car instances by predicting the depth ordering of each pixel in the image. Unlike the previous detection-based approaches, this method reasoned globally about all instances in the image simultaneously (rather than individual proposals) with an MRF-based formulation. However, inference of this graphical model was not performed end-to-end as shown to be possible in [329, 7, 46, 178]. Furthermore, although this method does not use object detections, it is trained with ground truth depth and assumes a maximum of nine cars in an image. Predicting all the instances in an image simultaneously (rather than classifying individual proposals) requires a model to be able to handle a variable number of output instances per image. As a result, [248] proposed a Recurrent Neural Network (RNN) for this task. However, this model was only for a single object category. Our proposed method not only outputs a variable number of instances, but can also handle multiple object classes.

Liang *et al.* [175] developed another proposal-free method based on the semantic segmentation network of [44]. The category-level segmentation, along with CNN features, was used to predict instance-level bounding boxes. The number of instances of each class was also predicted to enable a final spectral clustering step. However, this additional information predicted by Liang's network could have been obtained from an object detector. Arnab *et al.* [8] also started with an initial semantic segmentation network [7], and combined this with the outputs of an object detector using a CRF to reason about instances. This method was not trained end-to-end though, and could not really recover from errors in bounding-box localisation or occlusion.

Our method also has an initial semantic segmentation subnetwork, and uses the outputs of an object detector. However, in contrast to [8] it is trained end-to-end to improve on both semantic- and instance-segmentation performance (to our knowledge, this is the first work to achieve this). Furthermore, it can handle detector localisation errors and occlusions better due to the energy terms in our end-to-end CRF. In contrast to detection-based approaches [112, 111, 65, 184], our network requires no additional post-processing to create an instance segmentation map as in Fig. 4.1(c) and reasons about the entire image, rather than independent proposals. This global reasoning allows our method to produce more accurate segmentations. Our proposed system also handles a variable number of instances per image, and thus does not assume a maximum number of instances like [325, 326].

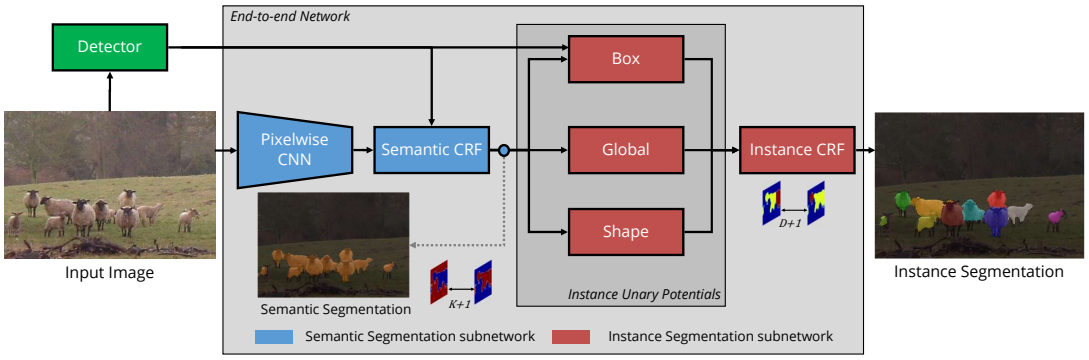


Figure 4.2: Network overview: Our end-to-end trained network consists of semantic- and instance-segmentation modules. The intermediate category-level segmentation, along with the outputs of an object detector, are used to reason about instances. This is done by instance unary terms which use information from the detector's bounding boxes, the initial semantic segmentation and also the object's shape. A final CRF is used to combine all this information together to obtain an instance segmentation. The output of the semantic segmentation module is a fixed size $W \times H \times (K+1)$ tensor where K is the number of object classes, excluding background, in the dataset. The final output, however, is of a variable $W \times H \times (D+1)$ dimensions where D is the number of detected objects (and one background label).

4.3 Proposed Approach

Our network (Fig. 4.2) contains an initial semantic segmentation module. We use the semantic segmentation result, along with the outputs of an object detector, to compute the unary potentials of a Conditional Random Field (CRF) defined over object instances. We perform mean field inference in this random field to obtain the Maximum a Posteriori (MAP) estimate, which is our labelling. Although our network consists of two conceptually different parts – a semantic segmentation module, and an instance segmentation network – the entire pipeline is fully differentiable, given object detections, and trained end-to-end.

4.3.1 Semantic Segmentation subnetwork

Semantic Segmentation assigns each pixel in an image a semantic class label from a given set, $\mathcal{L}$. In our case, this module uses the FCN8s architecture [189] which is based on the VGG [267] ImageNet model. For better segmentation results, we include mean field inference of a Conditional Random Field as the last layer of this module. This CRF contains the densely-connected pairwise potentials described in [154] and is formulated as a recurrent neural network as in [329]. Additionally, we include the Higher Order detection potential described in [7]. This detection potential has two primary benefits: Firstly, it improves semantic segmentation quality by encouraging consistency between object detections and segmentations. Secondly, it also recalibrates detection scores. This detection potential is

similar to the one previously proposed by [164], [275], [306] and [318], but formulated for the differentiable mean field inference algorithm. We employ this potential as we are already using object detection information for identifying object instances in the next stage. We denote the output at the semantic segmentation module of our network as the tensor $\mathbf{Q}$, where $Q_i(l)$ denotes the probability (obtained by applying the softmax function on the network's activations) of pixel i taking on the label $l \in \mathcal{L}$.

4.3.2 Instance Segmentation subnetwork

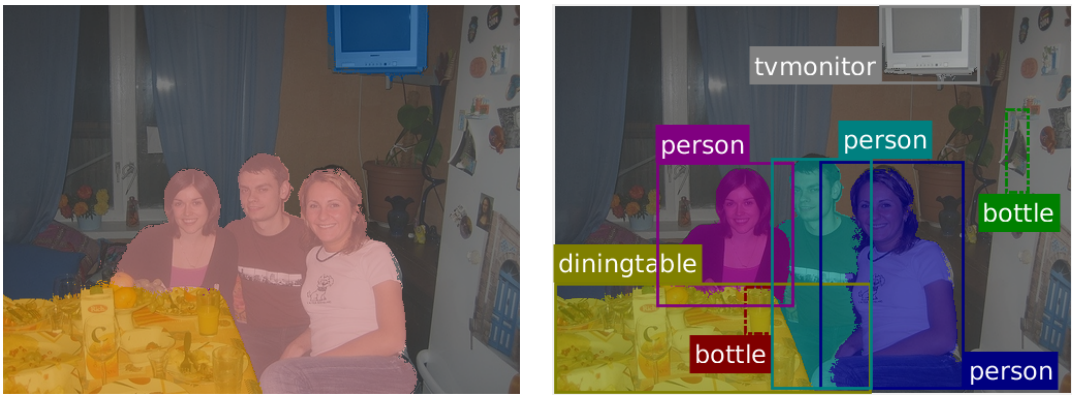
At the input to our instance segmentation subnetwork, we assume that we have two inputs available: The semantic segmentation predictions, $\mathbf{Q}$, for each pixel and label, and a set of object detections. For each input image, we assume that there are D object detections, and that the i^{th} detection is of the form (l_i, s_i, B_i) where $l_i \in \mathcal{L}$ is the detected class label, $s_i \in [0, 1]$ is the confidence score and B_i is the set of indices of the pixels falling within the detector's bounding box. Note that the number D varies for every input image.

The problem of instance segmentation can then be thought of as assigning every pixel to either a particular object detection, or the background label. This is based on the assumption that every object detection specifies a potential object instance. We define a multinomial random variable, V, at each of the N pixels in the image, and $\mathbf{V} = [V_1\ V_2 \dots V_N]^T$. Each variable at pixel i, V_i, is assigned a label corresponding to its instance. This label set, $\{0, 1, 2, ..., D\}$ changes for each image since D, the number of detections, varies for every image (0 is the background label). In the case of instance segmentation of images, the quality of a prediction is invariant to the permutations of the instance labelling. For example, labelling the "blue person" in Fig. 6.1(c) as "1" and the "purple person" as "2" is no different to labelling them as "2" and "1" respectively. This condition is handled by our loss function in Sec. 4.3.4.

Note that unlike works such as [325] and [326] we do not assume a maximum number of possible instances and keep a fixed label set. Furthermore, since we are considering object detection outputs jointly with semantic segmentation predictions, we have some robustness to high-scoring false positive detections unlike methods such as [50, 111, 184] which refine object detections into segmentations.

We formulate a Conditional Random Field over our instance variables, V, which consists of unary and pairwise energies. The energy of the assignment $\mathbf{v}$ to all the variables, $\mathbf{V}$, is

$$E(\mathbf{v}) = \sum_i U(v_i) + \sum_{i<j} P(v_i, v_j). \tag{4.1}$$



(a) Semantic Segmentation (b) Instance Segmentation

Figure 4.3: Instance segmentation using only the "Box" unary potential. This potential is effective when we have a good initial semantic segmentation (a). Occlusions between objects of the same class can be resolved by the pairwise term based on appearance differences. Note that we can ignore the confident, false-positive "bottle" detections (b). This is in contrast to methods such as [50, 112, 111, 169] which cannot recover from detection errors.

The unary energy is a sum of three terms, which take into account the object detection bounding boxes, the initial semantic segmentation and shape information,

$$U(v_i) = -\ln[w_1\psi_{Box}(v_i) + w_2\psi_{Global}(v_i) + w_3\psi_{Shape}(v_i)], \tag{4.2}$$

and are described further in Sections 4.3.2.1 through 4.3.2.3. w_1, w_2 and w_3 are all weighting co-efficients learned via backpropagation.

4.3.2.1 Box Term

This potential encourages a pixel to be assigned to the instance corresponding to the k^{th} detection if it falls within the detection's bounding box. This potential is proportional to the probability of the pixel's semantic class being equal to the detected class $Q_i(l_k)$ and the detection score, s_k.

$$\psi_{Box}(V_i = k) = \begin{cases} Q_i(l_k)s_k & \text{if } i \in B_k \\ 0 & \text{otherwise} \end{cases} \tag{4.3}$$

As shown in Fig. 4.3, this potential performs well when the initial semantic segmentation is good. It is robust to false positive detections, unlike methods which refine bounding boxes [50, 112, 111] since the detections are considered in light of our initial semantic segmentation, **Q**. Together with the pairwise term (Sec. 4.3.2.4), occlusions between objects of the same class can be resolved if there are appearance differences in the different instances.

(a) Only Box term (b) Box and Global terms

Figure 4.4: The "Global" unary potential (b) is particularly effective in cases where the input detection bounding box does not cover the entire extent of the object. Methods which are based on refining bounding-box detections such as [112, 111, 50, 65] cannot cope with poorly localised detections. Note, the overlaid detection boxes are an additional input to our system.

4.3.2.2 Global Term

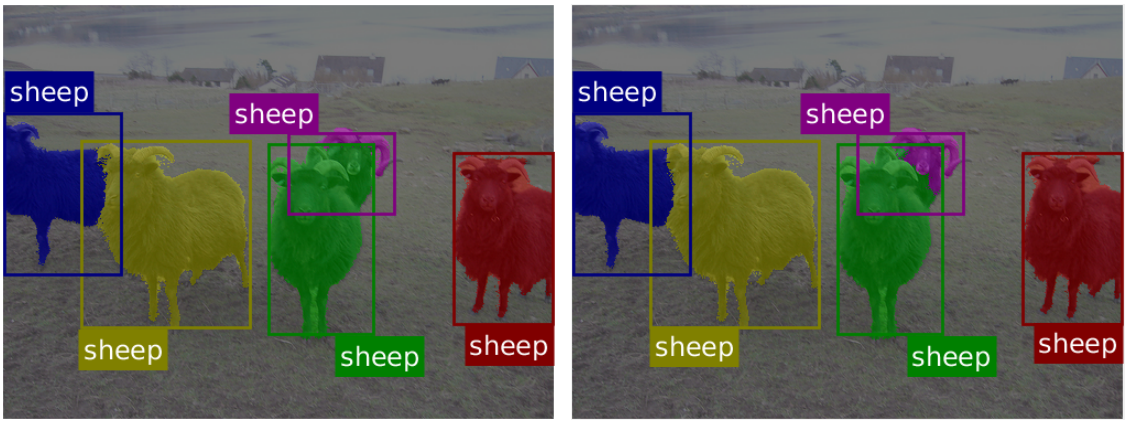
This term does not rely on bounding boxes, but only the segmentation prediction at a particular pixel, Q_i. It encodes the intuition that if we only know there are d possible instances of a particular object class, and have no further localisation information, each instance is equally probable, and this potential is proportional to the semantic segmentation confidence for the detected object class at that pixel:

$$\psi_{Global}(V_i = k) = Q_i(l_k). \tag{4.4}$$

As shown in Fig. 4.4, this potential overcomes cases where the bounding box does not cover the entire extent of the object, as it assigns probability mass to a particular instance label throughout all pixels in the image. This is also beneficial during training, as it ensures that the final output is dependent on the segmentation prediction at all pixels in the image, leading to error gradients that are more stable across batches and thus more amenable to backpropagation.

4.3.2.3 Shape Term

We also incorporate shape priors to help us reason about occlusions involving multiple objects of the same class, which may have minimal appearance variation between them, as shown in Fig. 4.5. In such cases, a prior on the expected shape of an object category can help us to identify the foreground instance within a bounding box. Previous approaches to incorporating shape priors in segmentation [120, 50, 304] have involved generating "shape



(a) Without shape term (b) With Shape term

Figure 4.5: The "Shape" unary potential (b) helps us to distinguish between the green and purple sheep, which the other two unary potentials cannot. Input detections are overlaid on the images.

exemplars" from the training dataset and, at inference time, matching these exemplars to object proposals using the Chamfer distance [265, 181].

We propose a fully differentiable method: Given a set of shape templates, $\mathcal{T}$, we warp each shape template using bilinear interpolation into $\tilde{\mathcal{T}}$ so that it matches the dimensions of the k^{th} bounding box, B_k. We then select the shape prior which matches the segmentation prediction for the detected class within the bounding box, $\mathbf{Q}_{B_k}(l_k)$, the best according to the normalised cross correlation. Our shape prior is then the Hadamard (elementwise) product ($\odot$) between the segmentation unaries and the matched shape prior:

$$t^* = \arg\max_{t \in \tilde{\mathcal{T}}} \frac{\sum \mathbf{Q}_{B_k}(l_k) \odot t}{\|\mathbf{Q}_{B_k}(l_k)\| \, \|t\|} \tag{4.5}$$

$$\psi(\mathbf{V}_{B_k} = k) = \mathbf{Q}_{B_k}(l_k) \odot t^*. \tag{4.6}$$

Equations 4.5 and 4.6 can be seen as a special case of max-pooling, and the numerator of Eq. 4.5 is simply a convolution that produces a scalar output since the two arguments are of equal dimension. Additionally, during training, we can consider the shape priors $\mathcal{T}$ as parameters of our "shape term" layer and backpropagate through to the matched exemplar t^* to update it. In practice, we initialised these parameters with the shape priors described in [304]. This consists of roughly 250 shape templates for each of five different aspect ratios. These were obtained by clustering foreground masks of object instances from the training set.

Here, we have only matched a single shape template to a proposed instance. This method could be extended in future to matching multiple templates to an instance, in which case each shape exemplar would correspond to a part of the object such as in DPM [88].

4.3.2.4 Pairwise term

The pairwise term consists of densely-connected Gaussian potentials [154] and encourages appearance and spatial consistency. The weights governing the importance of these terms are also learnt via backpropagation, as in [329]. We find that these priors are useful in the case of instance segmentation as well, since nearby pixels that have similar appearance often belong to the same object instance. They are often able to resolve occlusions based on appearance differences between objects of the same class (Fig. 4.3).

4.3.3 Inference of our Dynamic Instance CRF

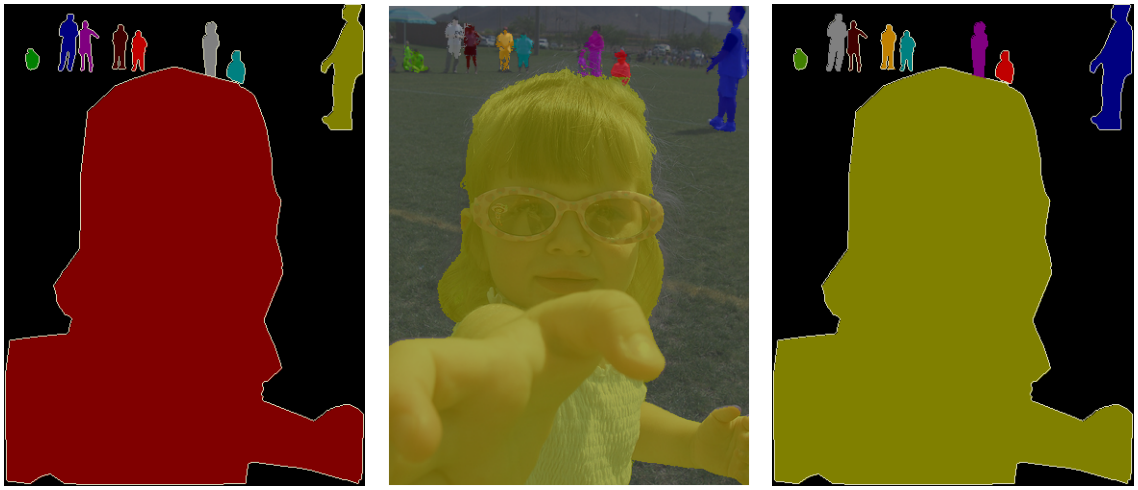
We use mean field inference to approximately minimise the Gibbs Energy in Eq. 4.1 which corresponds to finding the Maximum a Posteriori (MAP) labelling of the corresponding probability distribution, $P(\mathbf{v}) = \frac{1}{Z}\exp\left(-E(\mathbf{v})\right)$ where Z is the normalisation factor. Mean field inference is differentiable, and this iterative algorithm can be unrolled and seen as a recurrent neural network [329]. Following this approach, we can incorporate mean field inference of a CRF as a layer of our neural network. This enables us to train our entire instance segmentation network end-to-end.

Because we deal with a variable number of instances for every image, our CRF needs to be dynamically instantiated to have a different number of labels for every image, as observed in [8]. Therefore, unlike [329], none of our weights are class-specific. This weight-sharing not only allows us to deal with variable length inputs, but class-specific weights also do not make sense in the case of instance segmentation since a class label has no particular semantic meaning.

4.3.4 Loss Function

When training for instance segmentation, we have a single loss function which we back-propagate through our instance- and semantic-segmentation modules to update all the parameters. As discussed previously, we need to deal with different permutations of our final labelling which could have the same final result. The works of [325] and [326] order instances by depth to break this symmetry. However, this requires ground-truth depth maps during training which we do not assume that we have. Proposal-based methods [65, 112, 111, 184] do not have this issue since they consider a single proposal at a time, rather than the entire image. Our approach is similar to [248] in that we match the original ground truth to our instance segmentation prediction based on the Intersection over Union (IoU) [81] of each instance prediction and ground truth, as shown in Fig. 4.6.

More formally, we denote the ground-truth labelling of an image, $\mathcal{G}$, to be a set of r segments, $\{g_1, g_2, \ldots, g_r\}$, where each segment (set of pixels) is an object instance and has



(a) Original ground truth, $\mathcal{G}$ (b) Prediction, $\mathcal{P}$ (c) "Matched" ground truth, $\mathcal{G}^*$

Figure 4.6: Due to the problem of label permutations, we "match" the ground truth with our prediction before computing the loss when training.

an associated semantic class label. Our prediction, which is the output of our network, $\mathcal{P}$, is a set of s segments, $\{p_1, p_2, \ldots, p_s\}$, also where each segment corresponds to an instance label and also has an associated class label. Note that r and s may be different since we may predict greater or fewer instances than actually present. Let $\mathcal{M}$ denote the set of all permutations of the ground-truth, $\mathcal{G}$. As can be seen in Fig. 4.6, different permutations of the ground-truth correspond to the same qualitative result. We define the "matched" ground-truth, $\mathcal{G}^*$, as the permutation of the original ground-truth labelling which maximises the IoU between the prediction, $\mathcal{P}$, and ground truth:

$$\mathcal{G}^* = \arg\max_{m \in \mathcal{M}} \text{IoU}(m, \mathcal{P}). \tag{4.7}$$

Once we have the "matched" ground truth, $\mathcal{G}^*$, (Fig. 4.6) for an image, we can apply any loss function to train our network for segmentation. In our case, we use the common cross-entropy loss function. We found that this performed better than the approximate IoU loss proposed in [155, 248].

Crucially, we do not need to evaluate all permutations of the ground truth to compute Eq. 4.7, since it can be formulated as a maximum-weight bipartite matching problem. The edges in our bipartite graph connect ground-truth and predicted segments. The edge weights are given by the IoU between the ground truth and predicted segments if they share the same semantic class label, and zero otherwise. Leftover segments are matched to "dummy" nodes with zero overlap.

Additionally, the ordering of the instances in our network are actually determined by the object detector, which remains static during training. As a result, the ordering of our predictions does not fluctuate much during training – it only changes in cases where there are multiple detections overlapping an object.

4.3.5 Network Training

We first train a network for semantic segmentation with the standard cross-entropy loss. In our case, this network is FCN8s [189] with a CRF whose inference is unrolled as an RNN and trained end-to-end, as described in [329] and [7]. To this pretrained network, we append our instance segmentation subnetwork, and finetune with instance segmentation annotations and only the loss detailed in Sec. 4.3.4. For the semantic segmentation subnetwork, we train with an initial learning rate of 10^{-8}, momentum of 0.9 and batch size of 20. The learning rate is low since we do not normalise the loss by the number of pixels. This is so that images with more pixels contribute a higher loss. The normalised learning rate is approximately 2×10^{-3}. When training our instance segmentation network as well, we lower the learning rate to 10^{-12} and use a batch size of 1 instead. Decreasing the batch size gave empirically better results. We also clipped gradients (a technique common in training RNNs [226]) with ℓ_2 norms above 10^9. This threshold was set by observing "normal" gradient magnitudes during training. The relatively high magnitude is due to the fact that our loss is not normalised. In our complete network, we have two CRF inference modules which are RNNs (one each in the semantic- and instance-segmentation subnetworks), and gradient clipping facilitated successful training.

4.3.6 Discussion

Our network is able to compute a semantic and instance segmentation of the input image in a single forward pass. We do not require any post-processing, such as the patch aggregation of [184], "mask-voting" of [65], "superpixel projection" of [112, 111, 169] or spectral clustering of [175]. The fact that we compute an initial semantic segmentation means that we have some robustness to errors in the object detector (Fig. 4.3). Furthermore, we are not necessarily limited by poorly localised object detections either (Fig. 4.4). Our CRF model allows us to reason about the entire image at a time, rather than consider independent object proposals, as done in [112, 111, 65, 184, 169]. Although we do not train our object detector jointly with the network, it also means that our segmentation network and object detector do not succumb to the same failure cases. Moreover, it ensures that our instance labelling does not "switch" often during training, which makes learning more stable. Finally, note that

although we perform mean field inference of a CRF within our network, we do not optimise the CRF's likelihood, but rather a cross-entropy loss (Sec 4.3.4).

4.4 Experimental Evaluation

Sections 4.4.1 to 4.4.5 describe our evaluation on the Pascal VOC validation set [81] and the Semantic Boundaries Dataset (SBD) [110] (which provides per-pixel annotations to 11355 previously unlabelled images from Pascal VOC). Section 4.4.6 details results on Cityscapes [57].

4.4.1 Experimental Details

We first train a network for semantic segmentation, thereafter we finetune it to the task of instance segmentation, as described in Sec. 4.3.5. Our training data for the semantic segmentation pretraining consists of images from Pascal VOC [81], SBD [110] and Microsoft COCO [180]. Finally, when finetuning for instance segmentation, we use only training data from either the VOC dataset, or from the SBD dataset. We train separate models for evaluating on the VOC validation set, and the SBD validation set. In each case, we remove validation set images from the initial semantic segmentation pretraining set. We use the publicly available R-FCN object detection framework [66], and ensure that the images used to train the detector do not fall into our test sets for instance segmentation.

4.4.2 Evaluation Metrics

We report the mean Average Precision over regions (AP^r) as defined by [112]. The difference between AP^r and the AP metric used in object detection [81] is that the Intersection over Union (IoU) is computed over predicted and ground-truth regions instead of bounding boxes. Furthermore, the standard AP metric uses an IoU threshold of 0.5 to determine whether a prediction is correct or not. Here, we use a variety of IoU thresholds since larger thresholds require more precise segmentations. Additionally, we report the AP^r_{vol} which is the average of the AP^r for 9 IoU thresholds ranging from 0.1 to 0.9 in increments of 0.1.

However, we also observe that the AP^r metric requires an algorithm to produce a ranked list of segments and their object class. It does not require, nor evaluate, the ability of an algorithm to produce a globally coherent segmentation map of the image, for example Fig. 6.1c. To measure this, we propose the "Matching IoU" which matches the predicted image and ground truth, and then calculates the corresponding IoU as defined in [81]. This matching procedure is the same as described in Sec. 4.3.4. This measure was originally proposed in [317], but has not been used since in evaluating instance segmentation systems.

Table 4.1: The effect of the different CRF unary potentials, and end-to-end training with them, on the VOC 2012 validation set.

	AP^r			AP^r_{vol}	match
	0.5	0.7	0.9		IoU
Box Term (piecewise)	60.0	47.3	21.2	54.9	42.6
Box+Global (piecewise)	59.1	46.1	23.4	54.6	43.0
Box+Global+Shape (piecewise)	59.5	46.4	23.3	55.2	44.8
Box Term (end-to-end)	60.7	47.4	24.6	56.2	46.9
Box+Global (end-to-end)	60.9	48.1	**25.5**	56.7	47.1
Box+Global+Shape (end-to-end)	**61.7**	**48.6**	25.1	**57.5**	**48.3**

4.4.3 Effect of Instance Potentials and End-to-End training

We first perform ablation studies on the VOC 2012 validation set. This dataset, consisting of 1464 training and 1449 validation images has very high-quality annotations with detailed object delineations which makes it the most suited for evaluating pixel-level segmentations.

In Tab. 4.1, we examine the effect of each of our unary potentials in our Instance subnetwork on overall performance. Furthermore, we examine the effect of end-to-end training the entire network as opposed to piecewise training. Piecewise training refers to freezing the pretrained semantic segmentation subnetwork's weights and only optimising the instance segmentation subnetwork's parameters. Note that when training with only the "Box" (Eq. 4.3) unary potential and pairwise term, we also have to add in an additional "Background" detection which encompasses the entire image. Otherwise, we cannot classify the background label.

We can see that each unary potential improves overall instance segmentation results, both in terms of AP^r_{vol} and the Matching IoU. The "Global" term (Eq. 4.4) shows particular improvement over the "Box" term at the high AP^r threshold of 0.9. This is because it can overcome errors in bounding box localisation (Fig. 4.4) and leverage our semantic segmentation network's accurate predictions to produce precise labellings. The "Shape" term's improvement in the AP^r_{vol} is primarily due to an improvement in the AP^r at low thresholds. By using shape priors, we are able to recover instances which were occluded and missed out. End-to-end training also improves results at all AP^r thresholds. Training with just the "Box" term shows a modest improvement in the AP^r_{vol} of 1.3%. Training with

Table 4.2: Comparison of Instance Segmentation performance to recent methods on the VOC 2012 validation set.

Method	AP^r					AP^r_{vol}
	0.5	0.6	0.7	0.8	0.9	
SDS [112]	43.8	34.5	21.3	8.7	0.9	–
Chen *et al.* [50]	46.3	38.2	27.0	13.5	2.6	–
PFN [175]	58.7	51.3	42.5	31.2	15.7	52.3
Arnab *et al.* [8]	58.3	52.4	45.4	34.9	20.1	53.1
MPA 1-scale [184]	60.3	54.6	45.9	34.3	17.3	54.5
MPA 3-scale [184]	**62.1**	**56.6**	47.4	36.1	18.5	56.5
Ours	61.7	55.5	**48.6**	**39.5**	**25.1**	**57.5**

the "Global" and "Shape" terms shows larger improvements of 2.1% and 2.3% respectively. This may be because the "Box" term only considers the semantic segmentation at parts of the image covered by object detections. Once we include the "Global" term, we consider the semantic segmentation over the entire image for the detected class. Training makes more efficient use of images, and error gradients are more stable in this case.

4.4.4 Results on VOC validation Set

We then compare our best instance segmentation model to recent methods on the VOC validation Set in Tab. 4.2. The fact that our algorithm achieves the highest AP^r at thresholds above 0.7 indicates that our method produces more detailed and accurate segmentations.

At an IoU threshold of 0.9, our improvement over the previous state-of-the-art (MPA [184]) is 6.6%, which is a relative improvement of 36%. Unlike [184, 112, 50], our network performs an initial semantic segmentation which may explain our more accurate segmentations. Other segmentation-based approaches, [8, 175] are not fully end-to-end trained. We also achieve the best AP^r_{vol} of 57.5%. The relatively small difference in AP^r_{vol} to MPA [184] despite large improvements at high IoU thresholds indicates that MPA performs better at low IoU thresholds. Proposal-based methods, such as [184, 112] are more likely to perform better at low IoU thresholds since they output more proposals than actual instances in an image (SDS evaluates 2000 proposals per image). Furthermore, note that whilst MPA takes 8.7s to process an image [184], our method requires approximately 1.5s on the same Titan X GPU. More detailed qualitative and quantitative results, including success and failure cases, are included in the supplementary material.

4.4.5 Results on SBD Dataset

We also evaluate our model on the SBD dataset, which consists of 5623 training and 5732 validation images, as shown in Tab. 4.3. Following other works, we only report AP^r

Table 4.3: Comparison of Instance Segmentation performance on the SBD Dataset

Method	AP^r 0.5	AP^r 0.7	AP^r_{vol}	match IoU
SDS [112]	49.7	25.3	41.4	–
MPA 1-scale [184]	55.5	–	48.3	–
Hypercolumn [111]	56.5	37.0	–	–
IIS [169]	60.1	38.7	–	–
CFM [64]	60.7	39.6	–	–
Hypercolumn rescore [111]	60.0	40.4	–	–
MPA 3-scale rescore [184]	61.8	–	52.0	–
MNC [65]	63.5	41.5	–	39.0
MNC, Instance FCN [62]	61.5	43.0	–	–
IIS sp. projection, rescore [169]	**63.6**	43.3	–	–
Ours (piecewise)	59.1	42.1	52.3	41.8
Ours (end-to-end)	62.0	**44.8**	**55.4**	**47.3**

results at IoU thresholds of 0.5 and 0.7. However, we provide more detailed results in our supplementary material. Once again, we show significant improvements over other work at high AP^r thresholds. Here, our AP^r at 0.7 improves by 1.5% over the previous state-of-the-art [169]. Note that [169, 184, 111] perform additional post-processing where their results are rescored using an additional object detector. In contrast, our results are obtained by a single forward pass through our network. We have also improved substantially on the AP^r_{vol} measure (3.4%) compared to other works which have reported it. We also used the publicly available source code[1], model and default parameters of MNC [65] to evaluate the "Matching IoU". Our method improves this by 8.3%. This metric is a stricter measure of segmentation performance, and our method, which is based on an initial semantic segmentation and includes a CRF as part of training therefore performs better.

4.4.6 Results on Cityscapes

Finally, we evaluate our algorithm on the Cityscapes road-scene understanding dataset [57]. This dataset consists of 2975 training images, and the held-out test set consisting of 1525 images are evaluated on an online server. None of the 500 validation images were used for training. We use an initial semantic segmentation subnetwork that is based on the ResNet-101 architecture [328], and all of the instance unary potentials described in Sec. 4.3.2.

As shown in Tab. 4.4, our method sets a new state-of-the-art on Cityscapes, surpassing concurrent work [15] and the best previous published work [287] by significant margins.

[1] https://github.com/daijifeng001/MNC

Table 4.4: Results on Cityscapes test set. Evaluation metrics and results of competing methods obtained from the online server. The "AP" metric of Cityscapes is similar to our AP^r_{vol} metric.

Method	AP	AP at 0.5	AP 100m	AP 50m
Ours	**23.4**	**45.2**	**36.8**	**40.9**
DWT [15]	19.4	35.3	31.4	36.8
SAIS [114]	17.4	36.7	29.3	34.0
InstanceCut [143]	13.0	27.9	22.1	26.1
Graph Decomp. [168]	9.8	23.2	16.8	20.3
RecAttend [242]	9.5	18.9	16.8	20.9
Pixel Encoding [287]	8.9	21.1	15.3	16.7
R-CNN [57]	4.6	12.9	7.7	10.3

4.5 Conclusion and Future Work

We have presented an end-to-end instance segmentation approach that produces intermediate semantic segmentations, and shown that finetuning for instance segmentation improves our network's semantic segmentations. Our approach differs from other methods which derive their architectures from object detection networks [65, 184, 111] in that our approach is more similar to a semantic segmentation network. As a result, our system produces more accurate and detailed segmentations as shown by our substantial improvements at high AP^r thresholds. Moreover, our system produces segmentation maps naturally, and in contrast to other published work, does not require any post-processing. Finally, our network produces a variable number of outputs, depending on the number of instances in the image. Our future work is to incorporate an object detector into the end-to-end training of our system to create a network that performs semantic segmentation, object detection and instance segmentation jointly. Possible techniques for doing this are suggested by by UberNet [150] and the Cross-Stitch units described in [204].

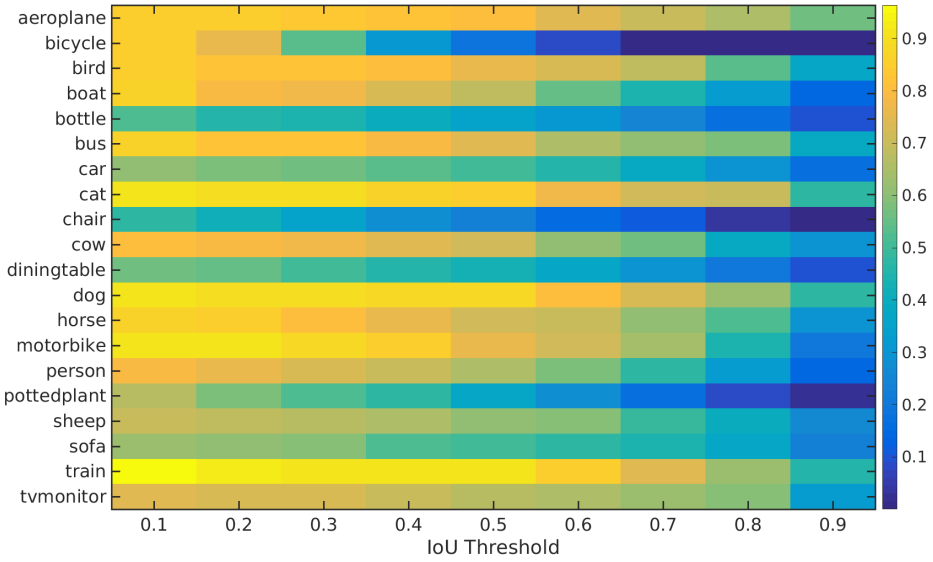


Figure 4.7: A visualisation of the AP^r obtained for each of the 20 classes on the VOC dataset, at nine different IoU thresholds. The x-axis represents the IoU threshold, and the y-axis each of the Pascal classes. Therefore, each "column" of this figure corresponds to the AP^r per class at a particular threshold, and is thus an alternate representation to the results tables. Best viewed in colour.

Appendices

In this appendix, we include more detailed qualitative and quantitative results on the VOC and SBD datasets.

Section 4.A shows more detailed results on the VOC dataset. Figure 4.7 shows a visualisation of our results at different AP^r thresholds, and Tables 4.6 to 4.8 show per-class AP^r results at thresholds of 0.5, 0.7 and 0.9.

Section 4.B shows more detailed results on the SBD dataset. Table 4.5 shows our mean AP^r results at thresholds from 0.5 to 0.9, whilst Tables 4.9 and 4.10 show per-class AP^r results at thresholds of 0.7 and 0.5 respectively.

Figures 4.9 and 4.10 show success and failure cases of our algorithm. Figure 4.11 compares the results of our algorithm to the publicly available model for MNC [65]. Figure 4.12 compares our results to those of FCIS [172], concurrent work which won the COCO 2016 challenge. Figure 4.13 presents some qualitative results on the Cityscapes dataset.

4.A Detailed results on the Pascal VOC dataset

Figure 4.7 shows a visualisation of the AP^r obtained by our method for each class across nine different thresholds. Each "column" of Fig. 4.7 corresponds to the AP^r for each class at a given IoU threshold. It is therefore an alternate representation for the results tables (Tables

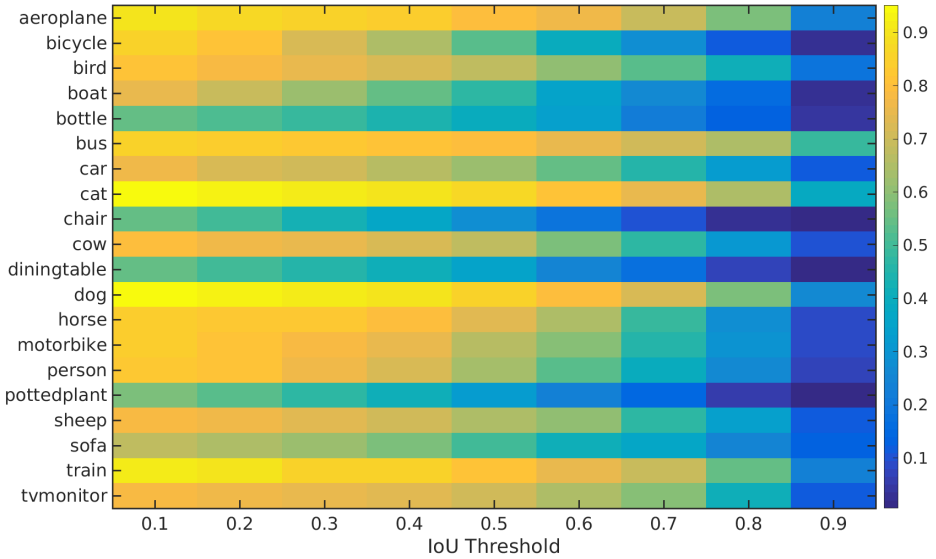


Figure 4.8: A visualisation of the AP^r obtained for each of the 20 classes on the SBD dataset, at nine different IoU thresholds. The x-axis represents the IoU threshold, and the y-axis each of the Pascal classes. Therefore, each "column" of this figure corresponds to the AP^r per class at a particular threshold, and is thus an alternate representation to the results tables. Best viewed in colour.

4.6 to 4.8). We can see that our method struggles with classes such as "bicycle", "chair", "dining table" and "potted plant". This may be explained by the fact that current semantic segmentation systems (including ours) struggle with these classes. All recent methods on the Pascal VOC leaderboard[2] obtain an IoU for these classes which is lower than the mean IoU for all classes. In fact the semantic segmentation IoU for the "chair" class is less than half of the mean IoU for all the classes for 16 out of the 20 most recent submissions on the VOC leaderboard at the time of writing.

Tables 4.6 to 4.8 show per-class instance segmentation results on the VOC dataset, at IoU thresholds of 0.9, 0.7 and 0.5 respectively. At an IoU threshold of 0.9, our method achieves the highest AP^r for 16 of the 20 object classes. At the threshold of 0.7, we achieve the highest AP^r in 15 classes. Finally, at an IoU threshold of 0.5, our method, MPA 3-scale [184] and PFN [175] each achieve the highest AP^r for 6 categories.

4.B Detailed results on the SBD dataset

Once again, we show a visualisation of the AP^r obtained by our method for each class across nine different thresholds (Fig. 4.8). The trend is quite similar to the VOC dataset in that our algorithm struggles on the same object classes ("chair", "dining table", "potted

[2] http://host.robots.ox.ac.uk:8080/leaderboard/displaylb.php?challengeid=11&compid=6

Table 4.5: Comparison of Instance Segmentation performance at multiple AP^r thesholds on the SBD validation set.

Method	AP^r					AP^r_{vol}
	0.5	0.6	0.7	0.8	0.9	
Ours (piecewise)	59.1	51.9	42.1	29.4	12.0	52.3
Ours (end-to-end)	**62.0**	**54.0**	**44.8**	**32.3**	**13.8**	**55.4**

plant", "bottle"). Note that our AP^r for the "bicycle" class has improved compared to the VOC dataset. This is probably because the VOC dataset has more detailed annotations. In the VOC dataset, each spoke of a bicycle's wheel is often labelled, whilst in SBD, the entire wheel is labelled as a single circle with the "bicycle" label. Therefore, the SBD dataset's coarser labelling makes it easier for an algorithm to perform well on objects with fine details.

Table 4.5 shows our mean AP^r over all classes at thresholds ranging from 0.5 to 0.9. Our AP^r at 0.9 is low compared to the result which we obtained on the VOC dataset. This could be for a number of reasons: As the SBD dataset is not as finely annotated as the VOC dataset, it might not be suited for measuring the AP^r at such high thresholds. Additionally, the training data is not as good for training our system which includes a CRF and is therefore able to delineate sharp boundaries. Finally, as the SBD dataset has 5732 validation images (compared to the 1449 in VOC), it leaves less data for pretraining our initial semantic segmentation module. This may hinder our network in being able to produce precise segmentations.

Tables 4.9 and 4.10 show per-class instance segmentation results on the SBD dataset, at IoU thresholds of 0.7 and 0.5 respectively. We can only compare results at these two thresholds since these are the only thresholds which other work has reported.

Table 4.6: Comparison of mean AP^r, achieved by different published methods, at an IoU threshold of **0.9**, for all twenty classes in the VOC dataset.

Method	Mean AP^r(%)	aero-plane	bike	bird	boat	bot-tle	bus	car	cat	chair	cow	ta-ble	dog	horse	mbike	per-son	plant	sheep	sofa	train	tv
Our method	**25.1**	**56.6**	0.03	**36.8**	**14.4**	9.9	39.0	**17.2**	**47.1**	**1.3**	**29.0**	9.5	**47.2**	**29.8**	**20.0**	**14.8**	**2.3**	**25.9**	**23.8**	**45.7**	**32.3**
MPA 3-scale [184]	18.5	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
MPA 1-scale [184]	17.3	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Arnab *et al.* [8]	20.1	43.7	0.03	30.0	13.2	**11.4**	**47.3**	10.9	34.5	0.7	19.6	**12.1**	35.6	24.3	13.3	10.7	0.4	20.7	20.9	35.0	17.4
PFN [175]	15.7	43.9	**0.1**	24.5	7.8	4.1	32.5	6.3	42.0	0.6	25.7	3.2	31.8	13.4	8.1	5.9	1.6	14.8	14.3	25.0	8.5
Chen *et al.* [50]	2.6	0.6	0	0.6	0.5	4.9	9.8	1.1	8.3	0.1	1.1	1.2	1.7	0.3	0.8	0.6	0.3	0.8	7.6	4.3	6.2
SDS [112]	0.9	0	0	0.2	0.3	2.0	3.8	0.2	0.9	0.1	0.2	1.5	0	0	0	0.1	0.1	0	2.3	0.2	5.8

Table 4.7: Comparison of mean AP^r, achieved by different published methods, at an IoU threshold of **0.7**, for all twenty classes in the VOC dataset.

Method	Mean AP^r(%)	aero-plane	bike	bird	boat	bot-tle	bus	car	cat	chair	cow	ta-ble	dog	horse	mbike	per-son	plant	sheep	sofa	train	tv
Our method	**48.6**	**69.6**	1.4	**68.2**	**45.1**	25.2	61.1	**38.7**	72.1	**11.2**	**56.3**	30.0	**73.3**	**60.7**	**64.3**	**46.8**	**17.1**	**49.1**	**44.6**	**75.0**	**62.0**
MPA 3-scale [184]	47.4	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
MPA 1-scale [184]	45.9	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Arnab *et al.* [8]	45.4	68.9	0.84	65.1	38.3	**26.3**	**64.7**	31.8	72.7	6.7	45.4	**32.9**	67.9	60.0	63.7	41.1	13.4	43.9	41.1	74.6	48.1
PFN [175]	42.5	68.5	**5.6**	60.4	34.8	14.9	61.4	19.2	**78.6**	4.2	51.1	28.2	69.6	**60.7**	60.5	26.5	9.8	35.1	43.9	71.2	45.6
Chen *et al.* [50]	27.0	40.8	0.07	40.1	16.2	19.6	56.2	26.5	46.1	2.6	25.2	16.4	36.0	22.1	20.0	22.6	7.7	27.5	19.5	47.7	46.7
SDS [112]	21.3	17.8	0	32.5	7.2	19.2	47.7	22.8	42.3	1.7	18.9	16.9	20.6	14.4	12.0	15.7	5.0	23.7	15.2	40.5	51.4

Table 4.8: Comparison of mean AP^r, achieved by different published methods, at an IoU threshold of **0.5**, for all twenty classes in the VOC dataset.

Method	Mean AP^r(%)	aero-plane	bike	bird	boat	bot-tle	bus	car	cat	chair	cow	ta-ble	dog	horse	mbike	per-son	plant	sheep	sofa	train	tv
Our method	61.7	80.2	**19.3**	**76.4**	**69.0**	35.3	74.5	50.8	84.5	22.8	70.9	43.3	**87.7**	71.3	76.2	65.6	37.2	**61.3**	50.3	**90.5**	67.2
MPA 3-scale [184]	**62.1**	79.7	11.5	71.6	54.6	**44.7**	**80.9**	**62.0**	85.4	**26.5**	64.5	46.6	87.6	71.7	**77.9**	**72.1**	48.8	57.4	48.8	78.9	70.8
MPA 1-scale [184]	60.3	79.2	13.4	71.6	59.0	41.5	73.8	52.3	87.3	23.3	61.2	42.5	83.1	70.0	77.0	67.6	**50.7**	56.0	45.9	80.0	70.5
Arnab *et al.* [8]	58.4	**80.4**	7.9	74.4	59.8	32.7	76.6	39.6	84.6	19.3	62.7	44.1	81.0	74.7	72.0	58.6	32.0	59.6	50.5	87.4	68.4
PFN [175]	58.7	76.4	15.6	74.2	54.1	26.3	73.8	31.4	**92.1**	17.4	**73.7**	**48.1**	82.2	**81.7**	72.0	48.4	23.7	57.7	**64.4**	88.9	**72.3**
Chen *et al.* [50]	46.3	63.6	0.3	61.5	43.9	33.8	67.3	46.9	74.4	8.6	52.3	31.3	63.5	48.8	47.9	48.3	26.3	40.1	33.5	66.7	67.8
SDS [112]	43.8	58.8	0.5	60.1	34.4	29.5	60.6	40.0	73.6	6.5	52.4	31.7	62.0	49.1	45.6	47.9	22.6	43.5	26.9	66.2	66.1

Table 4.9: Comparison of mean AP^r, achieved by different published methods, at an IoU threshold of **0.7**, for all twenty classes in the SBD dataset.

Method	Mean AP^r(%)	aero-plane	bike	bird	boat	bot-tle	bus	car	cat	chair	cow	ta-ble	dog	horse	mbike	per-son	plant	sheep	sofa	train	tv
Our method	**44.8**	**69.0**	27.4	**52.7**	**26.4**	22.4	70.3	46.0	**74.7**	9.6	46.8	**16.9**	**71.6**	48.4	46.3	**40.3**	14.8	47.6	**36.5**	**69.7**	**58.2**
IIS sp, rescore [169]	43.3	61.9	**35.1**	44.4	**26.4**	**29.6**	**74.0**	**48.7**	66.8	**10.9**	**48.4**	13.6	64.0	**53.0**	**46.8**	33.0	**19.0**	**51.0**	23.7	62.2	53.9
IIS raw [169]	38.7	61.8	31.5	42.0	22.0	22.7	72.4	44.8	65.4	7.2	37.6	10.4	60.4	39.6	41.9	32.5	12.0	40.9	19.9	58.8	50.8

Table 4.10: Comparison of mean AP^r, achieved by different published methods, at an IoU threshold of **0.5**, for all twenty classes in the SBD dataset.

Method	Mean AP^r(%)	aero-plane	bike	bird	boat	bot-tle	bus	car	cat	chair	cow	ta-ble	dog	horse	mbike	per-son	plant	sheep	sofa	train	tv
Our method	62.0	**80.3**	52.8	68.5	47.4	39.5	79.1	61.5	**87.0**	28.1	**68.3**	**35.5**	**86.1**	73.9	66.1	63.8	32.9	65.3	**50.4**	**81.4**	**71.4**
IIS sp, rescore [169]	**63.6**	79.2	**67.9**	**70.0**	**47.9**	**45.3**	**81.6**	**68.8**	84.1	**30.4**	65.5	31.8	83.6	**75.5**	**74.5**	**66.6**	**37.7**	**70.6**	44.7	77.7	68.7
IIS raw [169]	60.1	77.3	65.3	65.5	42.5	35.4	80.3	62.2	83.9	27.2	61.6	32.4	82.3	70.9	71.4	63.1	31.3	63.6	44.9	78.3	62.4

Figure 4.9: Success cases of our method. *First and second row:* Our algorithm can leverage good initial semantic segmentations, and detections, to produce an instance segmentation. *Third row:* Notice that we have ignored three false-positive detections. Additionally, the red bounding box does not completely encompass the person, but our algorithm is still able to associate pixels "outside-the-box" with the correct detection (also applies to row 2). *Fourth row:* Our system is able to deal with the heavily occluded sheep, and ignore the false-positive detection. *Fifth row:* We have not been able to identify one bicycle on the left since it was not detected, but otherwise have performed well. *Sixth row:* Although subjective, the train has not been annotated in the dataset, but both our initial semantic segmentation and object detection networks have identified it.
Note that the first three images are from the VOC dataset, and the last three from SBD. Annotations in the VOC dataset are more detailed, and also make more use of the grey "ignore" label to indicate uncertain areas in the image. The first column shows the input image, and the results of our object detector which are another input to our network. Best viewed in colour.

Figure 4.10: Failure cases of our method. *First row:* Both our initial detector, and semantic segmentation system did not identify a car in the background. Additionally, the "brown" person prediction actually consists of two people that have been merged together. This is because the detector did not find the background person. *Second row:* Our initial semantic segmentation identified the table, but it is not there in the Instance Segmentation. This is because there was no "table detection" to associate these pixels with. Using heuristics, we could propose additional detections in cases like these. However, we have not done this in our work. *Third row:* A difficult case where we have segmented most of the people. However, sometimes two people instances are joined together as one person instance. This problem is because we do not have a detection for each person in the image. *Fourth row:* Due to our initial semantic segmentation, we have not been able to segment the green person and table correctly. *Fifth row:* We have failed to segment a bird although it was detected. *Sixth row:* The occluding cows, which all appear similar, pose a challenge, even with our shape priors. The first column shows the input image, and the results of our object detector which are another input to our network. Best viewed in colour.

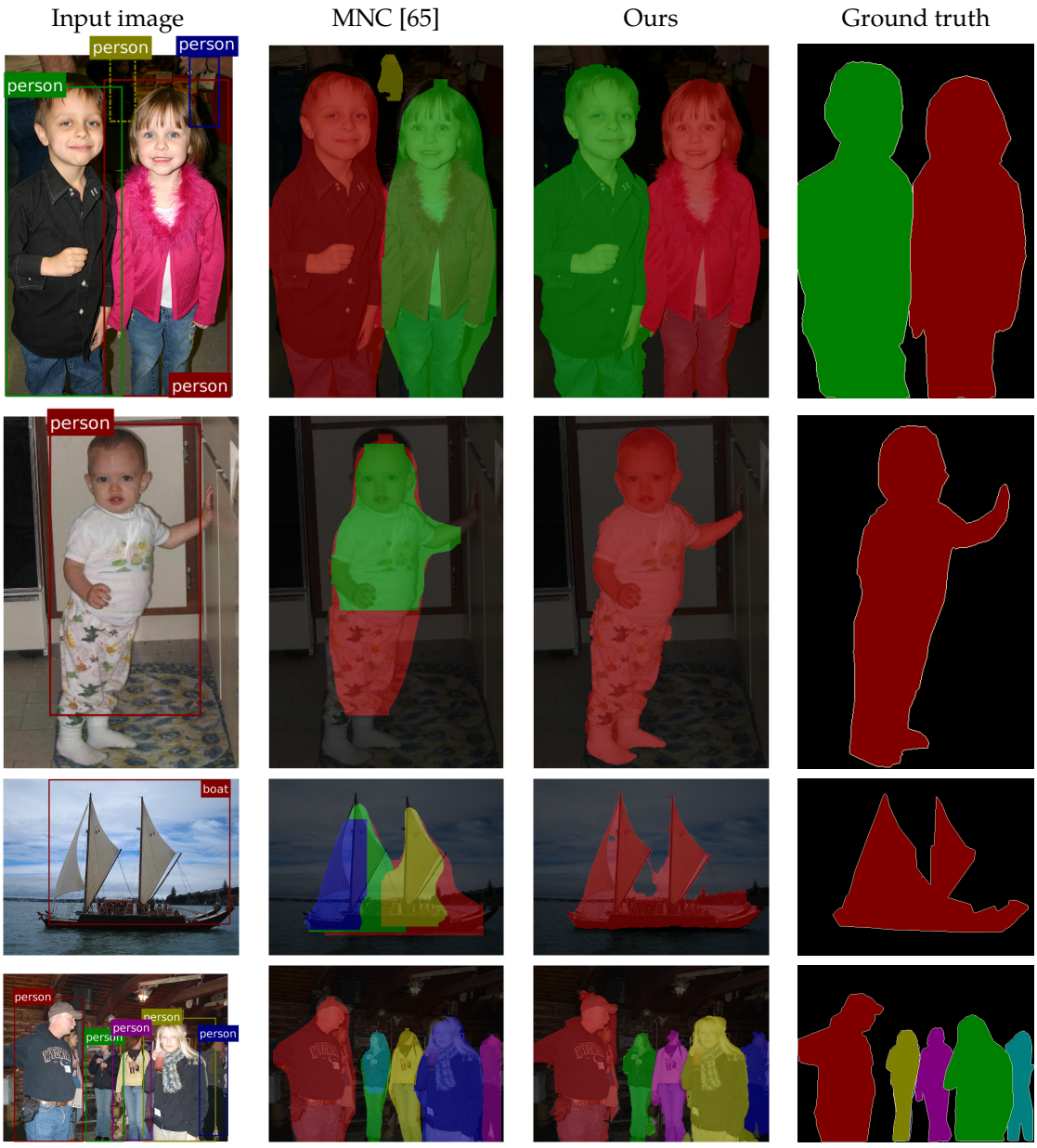


Figure 4.11: Comparison to MNC [65]. The above examples emphasise the advantages in our method over MNC [65]. Unlike proposal-based approaches such as MNC, our method can handle false-positive detections, poor bounding box localisation, reasons globally about the image and also produces more precise segmentations due to the initial semantic segmentation module which includes a differentiable CRF. *Row 1* shows a case where MNC, which scores segment-based proposals, is fooled by a false-positive detection and segments an imaginary human (yellow segment). Our method is robust to false-positive detections due to the initial semantic segmentation module which does not have the same failure modes as the detector. *Rows 2, 3 and 4* show how MNC [65] cannot deal with poorly localised bounding boxes. The horizontal boundaries of the red person in Row 2, and light-blue person in Row 4 correspond to the limits of the proposal processed by MNC. Our method, in contrast, can segment "outside the detection bounding box" due to the global instance unary potential (Eq. 4.4). As MNC does not reason globally about the image, it cannot handle cases of overlapping bounding boxes well, and produces more instances than there actually are. The first column shows the input image, and the results of our object detector which are another input to our network. MNC does not use these detections, but does internally produce box-based proposals which are not shown. Best viewed in colour.

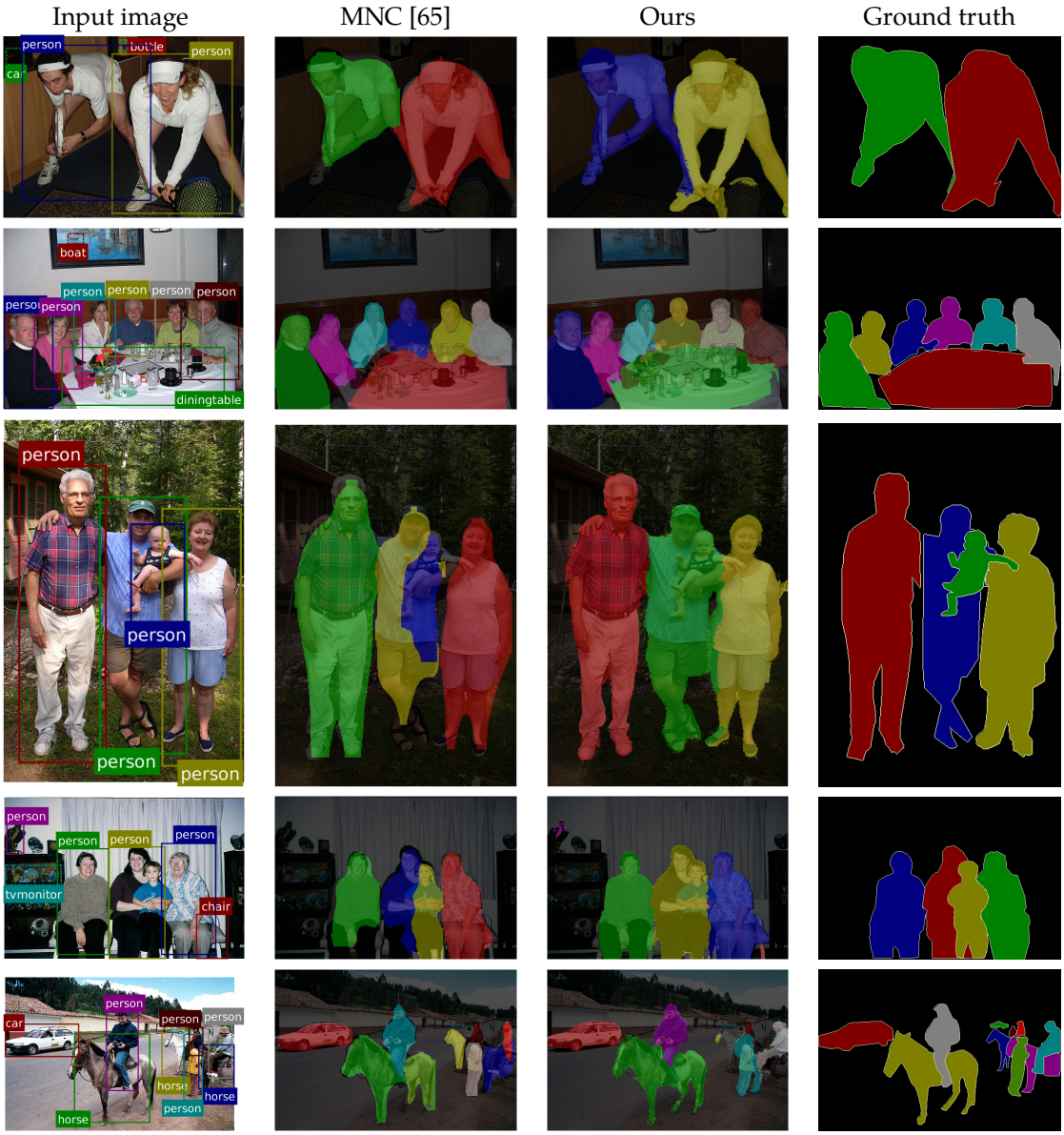


Figure 4.11 continued: Comparison to MNC [65]. The above examples show that our method produces more precise segmentations than MNC, that adhere to the boundaries of the objects. However, in Rows 3, 4 and 5, we see that MNC is able to segment instances that our method misses out. In *Row 3*, our algorithm does not segment the baby, although there is a detection for it. This suggests that our shape prior which was formulated to overcome such occlusions could be better. As MNC processes individual instances, it does not have a problem with dealing with small, occluding instances. In *Row 4*, MNC has again identified a person that our algorithm could not. However, this is because we did not have a detection for this person. In *Row 5*, MNC has segmented the horses on the right better than our method. The first column shows the input image, and the results of our object detector which are another input to our network. MNC does not use these detections, but does internally produce box-based proposals which are not shown. We used the publicly available code, models and default parameters of MNC to produce this figure. Best viewed in colour.

Figure 4.12: Comparison to FCIS [172]. The above images compare our method to the concurrent work, FCIS [172], which was trained on COCO [180] and won the COCO 2016 challenge. Unlike proposal-based methods such as FCIS, our method can handle false-positive detections and poor bounding-box localisation. Furthermore, as our method reasons globally about the image, one pixel can only be assigned to a single instance, which is not the case with FCIS. Our method also produces more precise segmentations, as it includes a differentiable CRF, and it is based off a semantic segmentation network. The results of FCIS are obtained from their publicly available results on the COCO test set (`https://github.com/daijifeng001/TA-FCN`). Note that FCIS is trained on COCO, and our model is trained on Pascal VOC which does not have as many classes as COCO, such as "umbrella" and "suitcase" among others. As a result, we are not able to detect these objects. The first column shows the input image, and the results of our object detector which are another input to our network. FCIS does not use these detections, but does internally produce proposals which are not shown. Best viewed in colour.

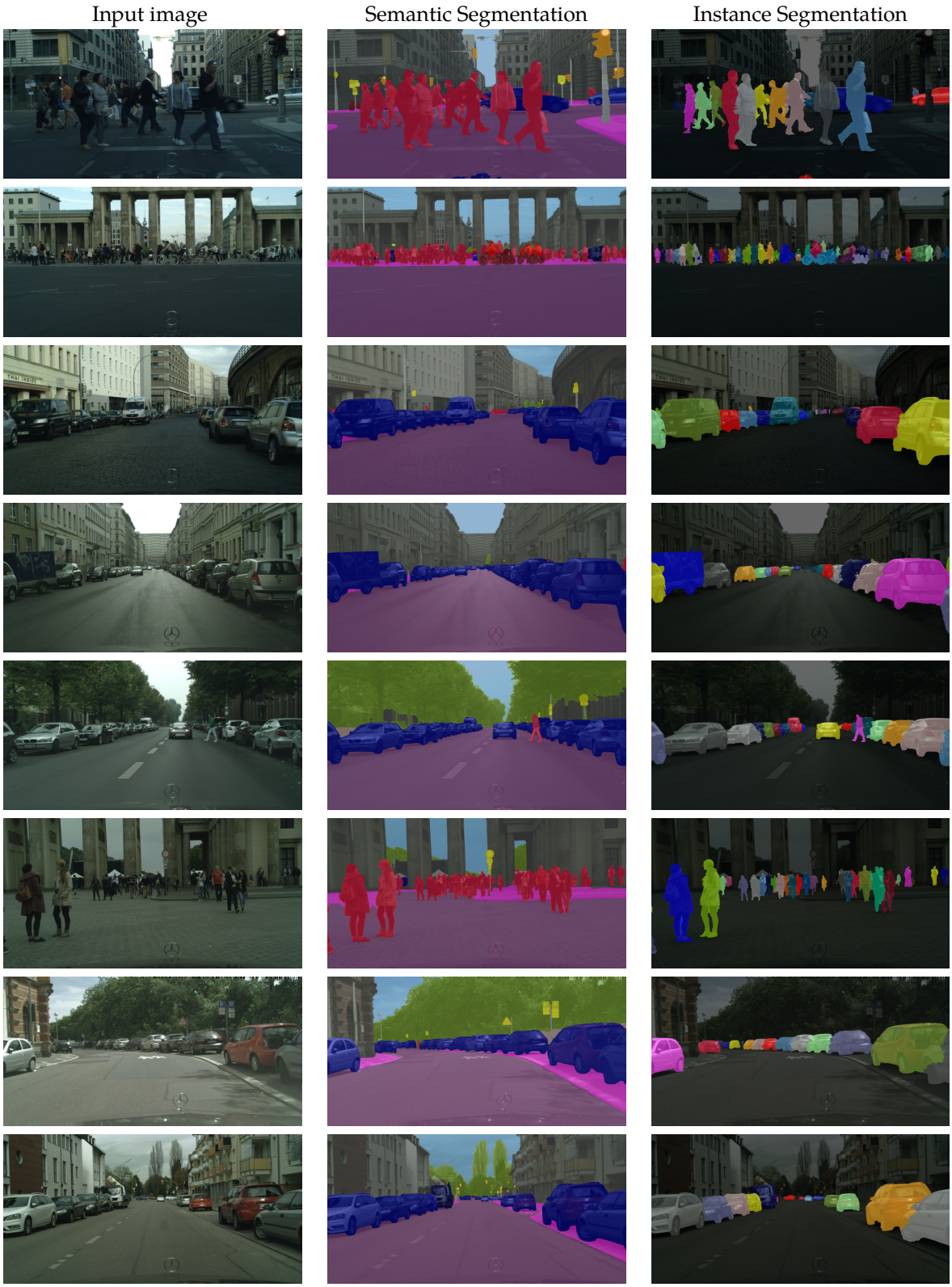


Figure 4.13: Sample results on the Cityscapes dataset. The above images show how our method can handle the large numbers of instances present in the Cityscapes dataset. Unlike other recent approaches, our algorithm can deal with objects that are not continuous – such as the car in the first row which is occluded by a pole. Best viewed in colour.

Chapter 5

Weakly- and Semi-Supervised Panoptic Segmentation

We present a weakly supervised model that jointly performs both semantic- and instance-segmentation – a particularly relevant problem given the substantial cost of obtaining pixel-perfect annotation for these tasks. In contrast to many popular instance segmentation approaches based on object detectors, our method does not predict any overlapping instances. Moreover, we are able to segment both "thing" and "stuff" classes, and thus explain all the pixels in the image. "Thing" classes are weakly-supervised with bounding boxes, and "stuff" with image-level tags. We obtain state-of-the-art results on Pascal VOC, for both full and weak supervision (which achieves about 95% of fully-supervised performance). Furthermore, we present the first weakly-supervised results on Cityscapes for both semantic- and instance-segmentation. Finally, we use our weakly supervised framework to analyse the relationship between annotation quality and predictive performance, which is of interest to dataset creators.

5.1 Introduction

Convolutional Neural Networks (CNNs) excel at a wide array of image recognition tasks [117, 267, 243]. However, their ability to learn effective representations of images requires large amounts of labelled training data [254, 274]. Annotating training data is a particular bottleneck in the case of segmentation, where labelling each pixel in the image by hand is particularly time-consuming. This is illustrated by the Cityscapes dataset where finely annotating a single image took "more than 1.5h on average" [57]. In this paper, we address the problems of semantic- and instance-segmentation using only weak annotations in the form of bounding boxes and image-level tags. Bounding boxes take only 7 seconds to draw using the labelling method of [220], and image-level tags an average of 1 second per class [219]. Using only these weak annotations would correspond to a reduction factor of 30

Figure 5.1: We propose a method to train an instance segmentation network from weak annotations in the form of bounding-boxes and image-level tags. Our network can explain both "thing" and "stuff" classes in the image, and does not produce overlapping instances as common detector-based approaches [116, 65, 172].

in labelling a Cityscapes image which emphasises the importance of cost-effective, weak annotation strategies.

Our work differs from prior art on weakly-supervised segmentation [151, 301, 221, 63, 20] in two primary ways: Firstly, our model jointly produces semantic- and instance-segmentations of the image, whereas the aforementioned works only output instance-agnostic semantic segmentations. Secondly, we consider the segmentation of both "thing" and "stuff" classes [91, 3], in contrast to most existing work in both semantic- and instance-segmentation which only consider "things".

We define the problem of instance segmentation as labelling every pixel in an image with both its object class and an instance identifier [9, 8, 326]. It is thus an extension of semantic segmentation, which only assigns each pixel an object class label. "Thing" classes (such as "person" and "car") are countable and are also studied extensively in object detection [81, 180]. This is because their finite extent makes it possible to annotate tight, well-defined bounding boxes around them. "Stuff" classes (such as "sky" and "vegetation"), on the other hand, are amorphous regions of homogeneous or repetitive textures [91]. As these classes have ambiguous boundaries and no well-defined shape they are not appropriate to annotate with bounding boxes [177]. Since "stuff" classes are not countable, we assume that all pixels of a stuff category belong to the same, single instance. Recently, this task of jointly segmenting "things" and "stuff" at an instance-level has also been named "Panoptic Segmentation" by [142].

Note that many popular instance segmentation algorithms which are based on object detection architectures [116, 65, 172, 183, 184] are not suitable for this task, as also noted by [142]. These methods output a ranked list of proposed instances, where the different proposals are allowed to overlap each other as each proposal is processed independently of

the other. Consequently, these architectures are not suitable where each pixel in the image has to be explained, and assigned a unique label of either a "thing" or "stuff" class as shown in Fig. 5.1. This is in contrast to other instance segmentation methods such as [9, 15, 69, 143, 182].

In this work, we use weak bounding box annotations for "thing" classes, and image-level tags for "stuff" classes. Whilst there are many previous works on semantic segmentation from image-level labels, the best performing ones [301, 302, 216, 42] used a saliency prior. The salient parts of an image are "thing" classes in popular saliency datasets [51, 315, 264] and this prior therefore does not help at all in segmenting "stuff" as in our case. We also consider the "semi-supervised" case where we have a mixture of weak- and fully-labelled annotations.

To our knowledge, this is the first work which performs weakly-supervised, non-overlapping instance segmentation, allowing our model to explain all "thing" and "stuff" pixels in the image (Fig. 5.1). Furthermore, our model jointly produces semantic- and instance-segmentations of the image, which to our knowledge is the first time such a model has been trained in a weakly-supervised manner. Moreover, to our knowledge, this is the first work to perform either weakly supervised semantic- or instance-segmentation on the Cityscapes dataset. On Pascal VOC, our method achieves about 95% of fully-supervised accuracy on both semantic- and instance-segmentation. Furthermore, we surpass the state-of-the-art on fully-supervised instance segmentation as well. Finally, we use our weakly- and semi-supervised framework to examine how model performance varies with the number of examples in the training set and the annotation quality of each example, with the aim of helping dataset creators better understand the trade-offs they face in this context.

5.2 Related Work

Instance segmentation is a popular area of scene understanding research. Most top-performing algorithms modify object detection networks to output a ranked list of segments instead of boxes [116, 65, 172, 183, 184, 112]. However, all of these methods process each instance independently and thus overlapping instances are produced – one pixel can be assigned to multiple instances simultaneously. Additionally, object detection based architectures are not suitable for labelling "stuff" classes which cannot be described well by bounding boxes [177]. These limitations, common to all of these methods, have also recently been raised by Kirillov *et al.* [142]. We observe, however, that there are other instance segmentation approaches based on initial semantic segmentation networks [9, 15, 69, 143]

which do not produce overlapping instances and can naturally handle "stuff" classes. Our proposed approach extends methods of this type to work with weaker supervision.

Although prior work on weakly-supervised instance segmentation is limited, there are many previous papers on weakly-supervised semantic segmentation, which is also relevant to our task. Early work in weakly-supervised semantic segmentation considered cases where images were only partially labelled using methods based on Conditional Random Fields (CRFs) [295, 121]. Subsequently, many approaches have achieved high accuracy using only image-level labels [151, 301, 231, 228], bounding boxes [139, 221, 63], scribbles [177] and points [20]. Additionally, the motion cues in videos (which were only annotated with video-level tags) have also been exploited to learn semantic segmentation models of moving objects [281, 127]. A popular paradigm for these approaches is "self-training" [257]: a model is trained in a fully-supervised manner by generating the necessary ground truth with the model itself in an iterative, Expectation-Maximisation (EM)-like procedure [221, 63, 177, 228]. Such approaches are sensitive to the initial, approximate ground truth which is used to bootstrap training of the model. To this end, Khoreva *et al.* [139] showed how, given bounding box annotations, carefully chosen unsupervised foreground-background and segmentation-proposal algorithms could be used to generate high-quality approximate ground truth such that iterative updates to it were not required thereafter.

Our work builds on the "self-training" approach to perform instance segmentation. To our knowledge, only Khoreva *et al.* [139] have published results on weakly-supervised instance segmentation. However, the model used by [139] was not competitive with the existing instance segmentation literature in a fully-supervised setting. Moreover, [139] only considered bounding-box supervision, whilst we consider image-level labels as well. Recent work by [128] modifies Mask-RCNN [116] to train it using fully-labelled examples of some classes, and only bounding box annotations of others. Our proposed method can also be used in a semi-supervised scenario (with a mixture of fully- and weakly-labelled training examples), but unlike [128], our approach works with only weak supervision as well. Furthermore, in contrast to [139] and [128], our method does not produce overlapping instances, handles "stuff" classes and can thus explain every pixel in an image as shown in Fig. 5.1.

5.3 Proposed Approach

We first describe how we generate approximate ground truth data to train semantic- and instance-segmentation models with in Sec. 5.3.1 through 5.3.4. Thereafter, in Sec. 5.3.5, we discuss the network architecture that we use. To demonstrate our method and ensure the

reproducibility of our results, we release our approximate ground truth and the code to generate it[1].

5.3.1 Training with weaker supervision

In a fully-supervised setting, semantic segmentation models are typically trained by performing multinomial logistic regression independently for each pixel in the image. The loss function, the cross entropy between the ground-truth distribution and the prediction, can be written as

$$L = -\sum_{i \in \Omega} \log p(l_i|\mathbf{I}) \tag{5.1}$$

where l_i is the ground-truth label at pixel i, $p(l_i|\mathbf{I})$ is the probability (obtained from a softmax activation) predicted by the neural network for the correct label at pixel i of image $\mathbf{I}$ and Ω is the set of pixels in the image.

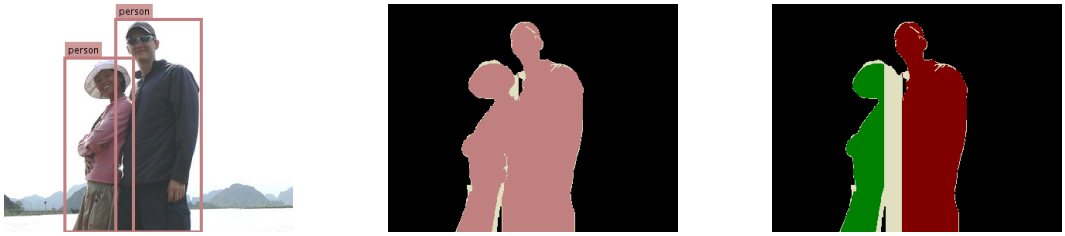
In the weakly-supervised scenarios considered in this paper, we do not have reliable annotations for all pixels in Ω. Following recent work [139, 151, 20, 228], we use our weak supervision and image priors to approximate the ground-truth for a subset $\Omega' \subset \Omega$ of the pixels in the image. We then train our network using the estimated labels of this smaller subset of pixels. Section 5.3.2 describes how we estimate Ω' and the corresponding labels for images with only bounding-box annotations, and Sec. 5.3.3 for image-level tags.

Our approach to approximating the ground truth is based on the principle of only assigning labels to pixels which we are confident about, and marking the remaining set of pixels, $\Omega \setminus \Omega'$, as "ignore" regions over which the loss is not computed. This is motivated by Bansal *et al.* [16] who observed that sampling only 4% of the pixels in the image for computing the loss during fully-supervised training yielded about the same results as sampling all pixels, as traditionally done. This supported their hypothesis that most of the training data for a pixel-level task is statistically correlated within an image, and that randomly sampling a much smaller set of pixels is sufficient. Moreover, [234] and [170] showed improved results by respectively sampling only 6% and 12% of the hardest pixels, instead of all of them, in fully-supervised training.

5.3.2 Approximate ground truth from bounding box annotations

We use GrabCut [252] (a classic foreground segmentation technique given a bounding-box prior) and MCG [5] (a segment-proposal algorithm) to obtain a foreground mask from a bounding-box annotation, following [139]. To achieve high precision in this approximate

[1] https://github.com/qizhuli/Weakly-Supervised-Panoptic-Segmentation



(a) Input image (b) Semantic segmentation approximate ground truth (c) Instance segmentation approximate ground truth

Figure 5.2: An example of generating approximate ground truth from bounding box annotations for an image (a). A pixel is labelled the with the bounding-box label if it belongs to the foreground masks of both GrabCut [252] and MCG [5] (b). Approximate instance segmentation ground truth is generated using the fact that each bounding box corresponds to an instance (c). Grey regions are "ignore" labels over which the loss is not computed due to ambiguities in label assignment.

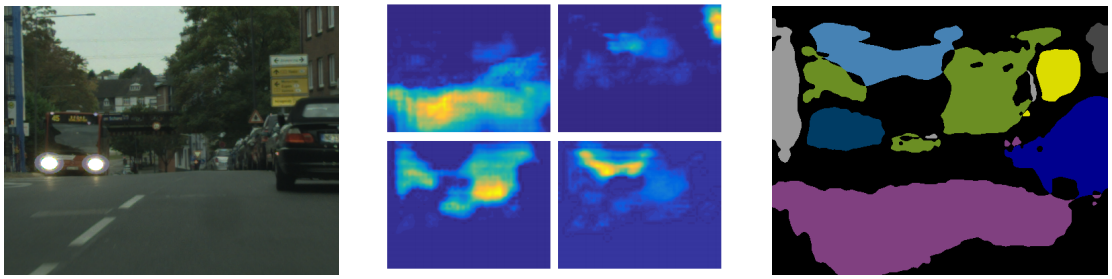
labelling, a pixel is only assigned to the object class represented by the bounding box if both GrabCut and MCG agree (Fig. 5.2).

Note that the final stage of MCG uses a random forest trained with pixel-level supervision on Pascal VOC to rank all the proposed segments. We do not perform this ranking step, and obtain a foreground mask from MCG by selecting the proposal that has the highest Intersection over Union (IoU) with the bounding box annotation.

This approach is used to obtain labels for both semantic- and instance-segmentation as shown in Fig. 5.2. As each bounding box corresponds to an instance, the foreground for each box is the annotation for that instance. If the foreground of two bounding boxes of the same class overlap, the region is marked as "ignore" as we do not have enough information to attribute it to either instance.

5.3.3 Approximate ground-truth from image-level annotations

When only image-level tags are available, we leverage the fact that CNNs trained for image classification still have localisation information present in their convolutional layers [330]. Consequently, when presented with a dataset of only images and their tags, we first train a network to perform multi-label classification. Thereafter, we extract weak localisation cues for all the object classes that are present in the image (according to the image-level tags). These localisation heatmaps (as shown in Fig. 5.3) are thresholded to obtain the approximate ground-truth for a particular class. It is possible for localisation heatmaps for different classes to overlap. In this case, thresholded heatmaps occupying a smaller area are given precedence. We found this rule, like [151], to be effective in preventing small or thin objects from being missed.



Input image | Localisation heatmaps for road, building, vegetation and sky | Approximate ground truth generated from image tags

Figure 5.3: Approximate ground truth generated from image-level tags using weak localisation cues from a multi-label classification network. Cluttered scenes from Cityscapes with full "stuff" annotations makes weak localisation more challenging than Pascal VOC and ImageNet that only have "things" labels. Black regions are labelled "ignore". Colours follow Cityscapes convention.

Input Image | Iteration 0 | Iteration 2 | Iteration 5 | Ground truth

Figure 5.4: By using the output of the trained network, the initial approximate ground truth produced according to Sec. 5.3.2 and 5.3.3 (Iteration 0) can be improved. Black regions are "ignore" labels over which the loss is not computed in training. Note for instance segmentation, permutations of instance labels of the same class are equivalent.

Though this approach is independent of the weak localisation method used, we used Grad-CAM [258]. Grad-CAM is agnostic to the network architecture unlike CAM [330] and also achieves better performance than Excitation BP [324] on the ImageNet localisation task [254].

We cannot differentiate different instances of the same class from only image tags as the number of instances is unknown. This form of weak supervision is thus appropriate for "stuff" classes which cannot have multiple instances. Note that saliency priors, used by many works such as [301, 302, 216] on Pascal VOC, are not suitable for "stuff" classes as popular saliency datasets [51, 315, 264] only consider "things" to be salient.

5.3.4 Iterative ground truth approximation

The ground truth approximated in Sec. 5.3.2 and 5.3.3 can be used to train a network from random initialisation. However, the ground truth can subsequently be iteratively refined by using the outputs of the network on the training set as the new approximate ground

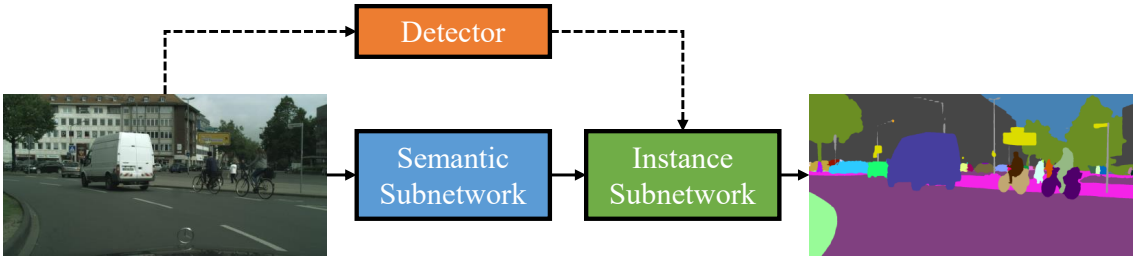


Figure 5.5: Overview of the network architecture. An initial semantic segmentation is partitioned into an instance segmentation, using the output of an object detector as a cue. Dashed lines indicate paths which are not backpropagated through during training.

truth as shown in Fig 5.4. The network's output is also post-processed with DenseCRF [154] using the parameters of Deeplab [44] (as also done by [151, 139]) to improve the predictions at boundaries. Moreover, any pixel labelled a "thing" class that is outside the bounding-box of the "thing" class is set to "ignore" as we are certain that a pixel for a thing class cannot be outside its bounding box. For a dataset such as Pascal VOC, we can set these pixels to be "background" rather than "ignore". This is because "background" is the only "stuff" class in the dataset.

5.3.5 Network Architecture

Using the approximate ground truth generation method described in this section, we can train a variety of segmentation models. Moreover, we can trivially combine this with full human-annotations to operate in a semi-supervised setting. We use the architecture of Arnab *et al.* [9] as it produces both semantic- and instance-segmentations, and can be trained end-to-end, given object detections. This network consists of a semantic segmentation subnetwork, followed by an instance subnetwork which partitions the initial semantic segmentation into an instance segmentation with the aid of object detections, as shown in Fig. 5.5.

We denote the output of the first module, which can be any semantic segmentation network, as $\mathbf{Q}$ where $Q_i(l)$ is the probability of pixel i of being assigned semantic label l. The instance subnetwork has two inputs – $\mathbf{Q}$ and a set of object detections for the image. There are D detections, each of the form (l_d, s_d, B_d) where l_d is the detected class label, $s_d \in [0, 1]$ the score and B_d the set of pixels lying within the bounding box of the d^{th} detection. This model assumes that each object detection represents a possible instance, and it assigns every pixel in the initial semantic segmentation an instance label using a Conditional Random Field (CRF). This is done by defining a multinomial random variable, X_i, at each of the N pixels in the image, with $\mathbf{X} = [X_1, X_2 \ldots, X_N]^\top$. This variable takes on a label from the set $\{1, \ldots, D\}$ where D is the number of detections. This formulation ensures that each pixel

can only be assigned one label. The energy of the assignment $\mathbf{x}$ to all instance variables $\mathbf{X}$ is then defined as

$$E(\mathbf{x}) = -\sum_{i}^{N} \ln\left(w_1 \psi_{Box}(x_i) + w_2 \psi_{Global}(x_i) + \epsilon\right) + \sum_{i<j}^{N} \psi_{Pairwise}(x_i, x_j). \tag{5.2}$$

The first unary term, the box term, encourages a pixel to be assigned to the instance represented by a detection if it falls within its bounding box,

$$\psi_{Box}(X_i = k) = \begin{cases} s_k Q_i(l_k) & \text{if } i \in B_k \\ 0 & \text{otherwise.} \end{cases} \tag{5.3}$$

Note that this term is robust to false-positive detections [9] since it is low if the semantic segmentation at pixel i, $Q_i(l_k)$ does not agree with the detected label, l_k. The global term,

$$\psi_{Global}(X_i = k) = Q_i(l_k), \tag{5.4}$$

is independent of bounding boxes and can thus overcome errors in mislocalised bounding boxes not covering the whole instance. Finally, the pairwise term is the common densely-connected Gaussian and bilateral filter [154] encouraging appearance and spatial consistency.

In contrast to [9], we also consider stuff classes (which object detectors are not trained for), by simply adding "dummy" detections covering the whole image with a score of 1 for all stuff classes in the dataset. This allows our network to jointly segment all "things" and "stuff" classes at an instance level. As mentioned before, the box and global unary terms are not affected by false-positive detections arising from detections for classes that do not correspond to the initial semantic segmentation $\mathbf{Q}$. The Maximum-a-Posteriori (MAP) estimate of the CRF is the final labelling, and this is obtained by using mean-field inference, which is formulated as a differentiable, recurrent network [329, 6].

We first train the semantic segmentation subnetwork using a standard cross-entropy loss with the approximate ground truth described in Sec 5.3.2 and 5.3.3. Thereafter, we append the instance subnetwork and finetune the entire network end-to-end. For the instance subnetwork, the loss function must take into account that different permutations of the same instance labelling are equivalent. As a result, the ground truth is "matched" to the prediction before the cross-entropy loss is computed as described in [9].

5.4 Experimental Evaluation

5.4.1 Experimental Set-up

Datasets and weak supervision We evaluate on two standard segmentation datasets, Pascal VOC [81] and Cityscapes [57]. Our weakly- and fully-supervised experiments

are trained with the same images, but in the former case, pixel-level ground truth is approximated as described in Sec. 5.3.1 through 5.3.4.

Pascal VOC has 20 "thing" classes annotated, for which we use bounding box supervision. There is a single "background" class for all other object classes. Following common practice on this dataset, we utilise additional images from the SBD dataset [110] to obtain a training set of 10582 images. In some of our experiments, we also use 54000 images from Microsoft COCO [180] only for the initial pretraining of the semantic subnetwork. We evaluate on the validation set, of 1449 images, as the evaluation server is not available for instance segmentation.

Cityscapes has 8 "thing" classes, for which we use bounding box annotations, and 11 "stuff" class labels for which we use image-level tags. We train our initial semantic segmentation model with the images for which 19998 coarse and 2975 fine annotations are available. Thereafter, we train our instance segmentation network using the 2975 images with fine annotations available as these have instance ground truth labelled. Details of the multi-label classification network we trained in order to obtain weak localisation cues from image-level tags (Sec. 5.3.3) are described in the supplementary. When using Grad-CAM, the original authors originally used a threshold of 15% of the maximum value for weak localisation on ImageNet. However, we increased the threshold to 50% to obtain higher precision on this more cluttered dataset.

Network training Our underlying segmentation network is a reimplementation of PSPNet [328]. For fair comparison to our weakly-supervised model, we train a fully-supervised model ourselves, using the same training hyperparameters (detailed in the supplementary) instead of using the authors' public, fully-supervised model. The original PSPNet implementation [328] used a large batch size synchronised over 16 GPUs, as larger batch sizes give better estimates of batch statistics used for batch normalisation [328, 45]. In contrast, our experiments are performed on a single GPU with a batch size of one 521×521 image crop. As a small batch size gives noisy estimates of batch statistics, our batch statistics are "frozen" to the values from the ImageNet-pretrained model as common practice [43, 129]. Our instance subnetwork requires object detections, and we train Faster-RCNN [243] for this task. All our networks use a ResNet-101 [117] backbone.

Evaluation Metrics We use the AP^r metric [112], commonly used in evaluating instance segmentation. It extends the AP, a ranking metric used in object detection [81], to segmentation where a predicted instance is considered correct if its Intersection over Union (IoU) with the ground truth instance is more than a certain threshold. We also report the

Table 5.1: Comparison of semantic segmentation performance to recent methods using only weak, bounding-box supervision on Pascal VOC. Note that [63] and [221] use the less accurate VGG network, whilst we and [139] use ResNet-101. "FS%" denotes the percentage of fully-supervised performance.

Method	Validation set			Test set		
	IoU (weak)	IoU (full)	FS%	IoU (weak)	IoU (full)	FS%
Without COCO annotations						
BoxSup [63]	62.0	63.8	**97.2**	64.6	–	–
Deeplab WSSL [221]	60.6	67.6	89.6	62.2	70.3	88.5
SDI [139]	69.4	74.5	93.2	–	–	–
Ours	**74.3**	**77.3**	96.1	**75.5**	**78.6**	**96.3**
With COCO annotations						
SDI [139]	74.2	77.7	95.5	–	–	–
Ours	**75.7**	**79.0**	**95.8**	**76.7**	**79.4**	**96.6**

AP^r_{vol} which is the mean AP^r across a range of IoU thresholds. Following the literature, we use a range of 0.1 to 0.9 in increments of 0.1 on VOC, and 0.5 to 0.95 in increments of 0.05 on Cityscapes.

However, as noted by several authors [317, 9, 15, 142], the AP^r is a ranking metric that does not penalise methods which predict more instances than there actually are in the image as long as they are ranked correctly. Moreover, as it considers each instance independently, it does not penalise overlapping instances. As a result, we also report the Panoptic Quality (PQ) recently proposed by [142],

$$\text{PQ} = \underbrace{\frac{\sum_{(p,g)\in TP} \text{IoU}(p,g)}{|TP|}}_{\text{Segmentation Quality (SQ)}} \times \underbrace{\frac{|TP|}{|TP| + \frac{1}{2}|FP| + \frac{1}{2}|FN|}}_{\text{Detection Quality (DQ)}}, \tag{5.5}$$

where p and g are the predicted and ground truth segments, and TP, FP and FN respectively denote the set of true positives, false positives and false negatives.

5.4.2 Results on Pascal VOC

Tables 5.1 and 5.2 show the state-of-art results of our method for semantic- and instance-segmentation respectively. For both semantic- and instance-segmentation, our weakly supervised model obtains about 95% of the performance of its fully-supervised counterpart, emphasising that accurate models can be learned from only bounding box annotations, which are significantly quicker and cheaper to obtain than pixelwise annotations. Table 5.2 also shows that our weakly-supervised model outperforms some recent fully supervised instance segmentation methods such as [8] and [175]. Moreover, our fully-supervised instance segmentation model outperforms all previous work on this dataset. The main

Table 5.2: Comparison of instance segmentation performance to recent (fully- and weakly-supervised) methods on the VOC 2012 validation set.

Method	AP^r 0.5	0.6	0.7	0.8	0.9	AP^r_{vol}	PQ
Weakly supervised without COCO							
SDI [139]	44.8	–	–	–	–	–	–
Ours	**60.5**	**55.2**	**47.8**	**37.6**	**21.6**	**55.6**	**59.0**
Fully supervised without COCO							
SDS [112]	43.8	34.5	21.3	8.7	0.9	–	–
Chen *et al.* [50]	46.3	38.2	27.0	13.5	2.6	–	–
PFN [175]	58.7	51.3	42.5	31.2	15.7	52.3	–
Ours (fully supervised)	**63.6**	**59.5**	**53.8**	**44.7**	**30.2**	**59.2**	**62.7**
Weakly supervised with COCO							
SDI [139]	46.4	–	–	–	–	–	–
Ours	**60.9**	**55.9**	**48.0**	**37.2**	**21.7**	**55.5**	**59.5**
Fully supervised with COCO							
Arnab *et al.* [8]	58.3	52.4	45.4	34.9	20.1	53.1	–
MPA [184]	62.1	56.6	47.4	36.1	18.5	56.5	–
Arnab *et al.* [9]	61.7	55.5	48.6	39.5	25.1	57.5	–
SGN [182]	61.4	55.9	49.9	42.1	26.9	–	–
Ours (fully supervised)	**63.9**	**59.3**	**54.3**	**45.4**	**30.2**	**59.5**	**63.1**

difference of our model to [9] is that our network is based on the PSPNet architecture using ResNet-101, whilst [9] used the network of [7] based on VGG [267].

We can obtain semantic segmentations from the output of our semantic subnetwork, or from the final instance segmentation (as we produce non-overlapping instances) by taking the union of all instances which have the same semantic label. We find that the IoU obtained from the final instance segmentation, and the initial pretrained semantic subnetwork to be very similar, and report the latter in Tab. 5.1. Further qualitative and quantitative results, including success and failure cases, are included in the supplement.

End-to-end training of instance subnetwork Our instance subnetwork can be trained in a piecewise fashion, or the entire network including the semantic subnetwork can be trained end-to-end. End-to-end training was shown to obtain higher performance by [9] for full supervision. We also observe this effect for weak supervision from bounding box annotations. A weakly supervised model, trained with COCO annotations improves from an AP^r_{vol} of 53.3 to 55.5. When not using COCO for training the initial semantic subnetwork, a slightly higher increase by 3.9 from 51.7 is observed. This emphasises that our training strategy (Sec. 5.3.1) is effective for both semantic- and instance-segmentation.

Table 5.3: Semantic- and instance-segmentation performance on Pascal VOC with varying levels of supervision from the Pascal and COCO datasets. The former is measured by the IoU, and latter by the AP^r_{vol} and PQ.

Dataset		IoU	AP^r_{vol}	PQ
VOC	COCO			
Weak	Weak	75.7	55.5	59.5
Weak	Full	75.8	56.1	59.8
Full	Weak	77.5	58.9	62.7
Full	Full	79.0	59.5	63.1

Table 5.4: Semantic segmentation performance on the Cityscapes validation set. We use more informative, bounding-box annotations for "thing" classes, and this is evident from the higher IoU than on "stuff" classes for which we only have image-level tags.

Method	IoU (weak)	IoU (full)	FS%
Ours (thing classes)	68.2	70.4	96.9
Ours (stuff classes)	60.2	72.4	83.1
Ours (overall)	63.6	71.6	88.8

Iterative training The approximate ground truth used to train our model can also be generated in an iterative manner, as discussed in Sec. 5.3.4. However, as the results from a single iteration (Tab. 5.1 and 5.2) are already very close to fully-supervised performance, this offers negligible benefit. Iterative training is, however, crucial for obtaining good results on Cityscapes as discussed in Sec. 5.4.3.

Semi-Supervision We also consider the case where we have a combination of weak and full annotations. As shown in Tab. 5.3, we consider all combinations of weak- and full-supervision of the training data from Pascal VOC and COCO. Table 5.3 shows that training with fully-supervised data from COCO and weakly-supervised data from VOC performs about the same as weak supervision from both datasets for both semantic- and instance-segmentation. Furthermore, training with fully annotated VOC data and weakly labelled COCO data obtains similar results to full supervision from both datasets. We have qualitatively observed that the annotations in Pascal VOC are of higher quality than those of Microsoft COCO (random samples from both datasets are shown in the supplementary). And this intuition is evident in the fact that there is not much difference between training with weak or full annotations from COCO. This suggests that in the case of segmentation, per-pixel labelling of additional images is not particularly useful if they are not labelled to a high standard, and that labelling fewer images at a higher quality (Pascal VOC) is more beneficial than labelling many images at a lower quality (COCO). This is because Tab. 5.3 demonstrates how both semantic- and instance-segmentation networks can be trained to achieve similar performance by using only bounding box labels instead of low-quality segmentation masks. The average annotation time can be considered a proxy for segmentation quality. While a COCO instance took an average of 79 seconds to segment [180], this figure is not mentioned for Pascal VOC [81, 80].

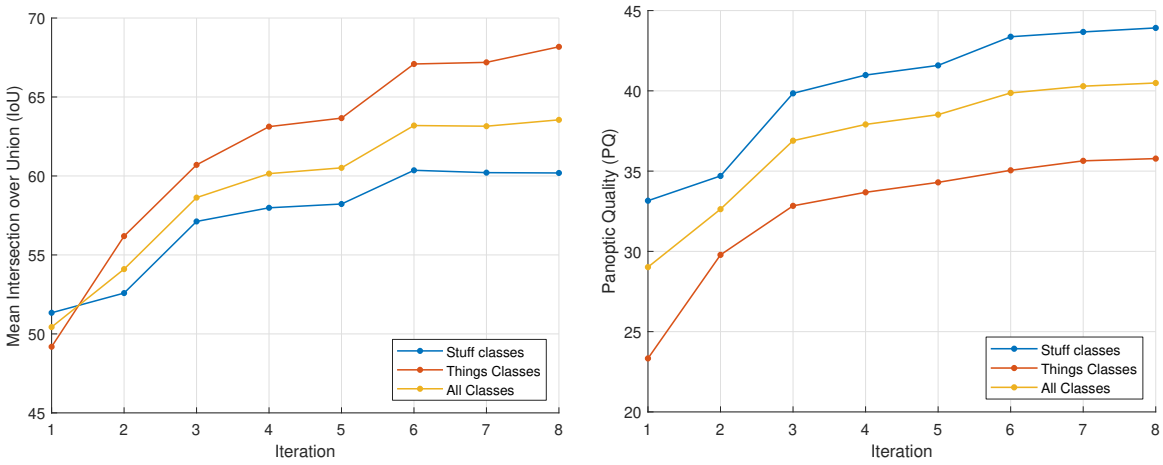
Table 5.5: Instance-level segmentation results on Cityscapes. On the validation set, we report results for both "thing" (th.) and "stuff" (st.) classes. The online server, which evaluates the test set, only computes the AP^r for "thing" classes. We compare to other fully-supervised methods which produce non-overlapping instances. To our knowledge, no published work has evaluated on both "thing" and "stuff" classes. Our fully supervised model, initialised from the public PSPNet model [328] is equivalent to our previous work [9], and competitive with the state-of-art. Note that we cannot use the public PSPNet pretrained model in a weakly-supervised setting.

	Validation									Test
	AP^r_{vol}			PQ			IoU			AP^r_{vol}
Method	th.	st.	all	th.	st.	all	th.	st.	all	th.
Ours (weak, ImageNet init.)	17.0	33.1	26.3	35.8	43.9	40.5	68.2	60.2	63.6	12.8
Ours (full, ImageNet init.)	24.3	42.6	34.9	39.6	52.9	47.3	70.4	72.4	71.6	18.8
Ours (full, PSPNet init.) [9]	28.6	52.6	42.5	42.5	62.1	53.8	80.1	79.5	79.8	23.4
Pixel Encoding [287]	9.9	–	–	–	–	–	–	–	–	8.9
RecAttend [242]	–	–	–	–	–	–	–	–	–	9.5
InstanceCut [143]	–	–	–	–	–	–	–	–	–	13.0
DWT [15]	21.2	–	–	–	–	–	–	–	–	19.4
SGN [182]	29.2	–	–	–	–	–	–	–	–	25.0

5.4.3 Results on Cityscapes

Tables 5.4 and 5.5 present, what to our knowledge is, the first weakly supervised results for either semantic or instance segmentation on Cityscapes. Table 5.4 shows that, as expected for semantic segmentation, our weakly supervised model performs better, relative to the fully-supervised model, for "thing" classes compared to "stuff" classes. This is because we have more informative bounding box labels for "things", compared to only image-level tags for "stuff". For semantic segmentation, we obtain about 97% of fully-supervised performance for "things" (similar to our results on Pascal VOC) and 83% for "stuff". Note that we evaluate images at a single-scale, and higher absolute scores could be obtained by multi-scale ensembling [328, 43].

For instance-level segmentation, the fully-supervised ratios for the PQ are similar to the IoU ratio for semantic segmentation. In Tab. 5.5, we report the AP^r_{vol} and PQ for both thing and stuff classes, assuming that there is only one instance of a "stuff" class in the image if it is present. Here, the AP^r_{vol} for "stuff" classes is higher than that for "things". This is because there can only be one instance of a "stuff" class, which makes instances easier to detect, particularly for classes such as "road" which typically occupy a large portion of the image. The Cityscapes evaluation server, and previous work on this dataset, only report the AP^r_{vol} for "thing" classes. As a result, we report results for "stuff" classes only on the validation set. Table 5.5 also compares our results to existing work which produces non-overlapping



(a) Semantic segmentation (IoU) (b) Instance segmentation (PQ)

Figure 5.6: Iteratively refining our approximate ground truth during training improves both semantic and instance segmentation on the Cityscapes validation set.

instances on this dataset, and shows that both our fully- and weakly-supervised models are competitive with recently published work on this dataset. We also include the results of our fully-supervised model, initialised from the public PSPNet model [328] released by the authors, and show that this is competitive with the state-of-art [182] among methods producing non-overlapping segmentations (note that [182] also uses the same PSPNet model). Further quantitative and qualitative results are in the supplementary.

Iterative training Iteratively refining our approximate ground truth during training, as described in Sec. 5.3.4, greatly improves our performance on both semantic- and instance-segmentation as shown in Fig. 5.6. We trained the network for 150 000 iterations before regenerating the approximate ground truth using the network's own output on the training set. Unlike on Pascal VOC, iterative training is necessary to obtain good performance on Cityscapes as the approximate ground truth generated on the first iteration is not sufficient to obtain high accuracy. This was expected for "stuff" classes, since we began from weak localisation cues derived from the image-level tags. However, as shown in Fig. 5.6, "thing" classes also improved substantially with iterative training, unlike on Pascal VOC where there was no difference. Compared to VOC, Cityscapes is a more cluttered dataset, and has large scale variations as the distance of an object from the car-mounted camera changes. These dataset differences may explain why the image priors employed by the methods we used (GrabCut [252] and MCG [5]) to obtain approximate ground truth annotations from bounding boxes are less effective. Furthermore, in contrast to Pascal VOC, Cityscapes has frequent co-occurences of the same objects in many different images, making it more

Table 5.6: The effect of different instance ranking methods on the AP^r_{vol} of our weakly supervised model computed on the Cityscapes validation set.

Ranking Method	AP^r_{vol} th.	AP^r_{vol} st.	PQ all
Detection score	17.0	26.7	40.5
Mean seg. confidence	14.6	33.1	40.5
Oracle	21.6	37.0	40.5

challenging for weakly supervised methods.

Effect of ranking methods on the AP^r The AP^r metric is a ranking metric derived from object detection. It thus requires predicted instances to be scored such that they are ranked in the correct relative order. As our network uses object detections as an additional input and each detection represents a possible instance, we set the score of a predicted instance to be equal to the object detection score. For the case of stuff classes, which object detectors are not trained for, we use a constant detection score of 1 as described in Sec. 5.3.5. An alternative to using a constant score for "stuff" classes is to take the mean of the softmax-probability of all pixels within the segmentation mask. Table 5.6 shows that this latter method improves the AP^r for stuff classes. For "things", ranking with the detection score performs better and comes closer to oracle performance which is the maximum AP^r that could be obtained with the predicted instances.

Changing the score of a segmented instance does not change the quality of the actual segmentation, but does impact the AP^r greatly as shown in Tab. 5.6. The PQ, which does not use scores, is unaffected by different ranking methods, and this suggests that it is a better metric for evaluating non-overlapping instance segmentation where each pixel in the image is explained.

5.5 Conclusion and Future Work

We have presented, to our knowledge, the first weakly-supervised method that jointly produces non-overlapping instance and semantic segmentation for both "thing" and "stuff" classes. Using only bounding boxes, we are able to achieve 95% of state-of-art fully-supervised performance on Pascal VOC. On Cityscapes, we use image-level annotations for "stuff" classes and obtain 88.8% of fully-supervised performance for semantic segmentation and 85.6% for instance segmentation (measured with the PQ). Crucially, the weak annotations we use incur only about 3% of the time of full labelling. As annotating pixel-level segmentation is time consuming, there is a dilemma between labelling few images with high quality or many images with low quality. Our semi-supervised experiment suggests

that the latter is not an effective use of annotation budgets as similar performance can be obtained from only bounding-box annotations.

Future work is to perform instance segmentation using only image-level tags and the number of instances of each object present in the image as supervision. This will require a network architecture that does not use object detections as an additional input.

Appendices

Section 5.A presents further qualitative and quantitative results of our experiments on Cityscapes and Pascal VOC. Section 5.B describes the training of the networks described in the main paper. Section 5.4.2 mentioned that the annotation quality of Pascal VOC [81] is better than COCO [180]. Some randomly drawn images from these datasets are presented to illustrate this point in Sec. 5.C. Finally, Sec. 5.D shows our calculation of how much the overall annotation time is reduced by using weak annotations, in comparison to full annotations, on the Cityscapes dataset.

5.A Additional Qualitative and Quantitative Results

Figure 5.7 and Tab. 5.7 present additional qualitative and quantitative results on the Cityscapes dataset. Similarly, Fig. 5.8 and Tab. 5.8 show additional results on the Pascal VOC dataset.

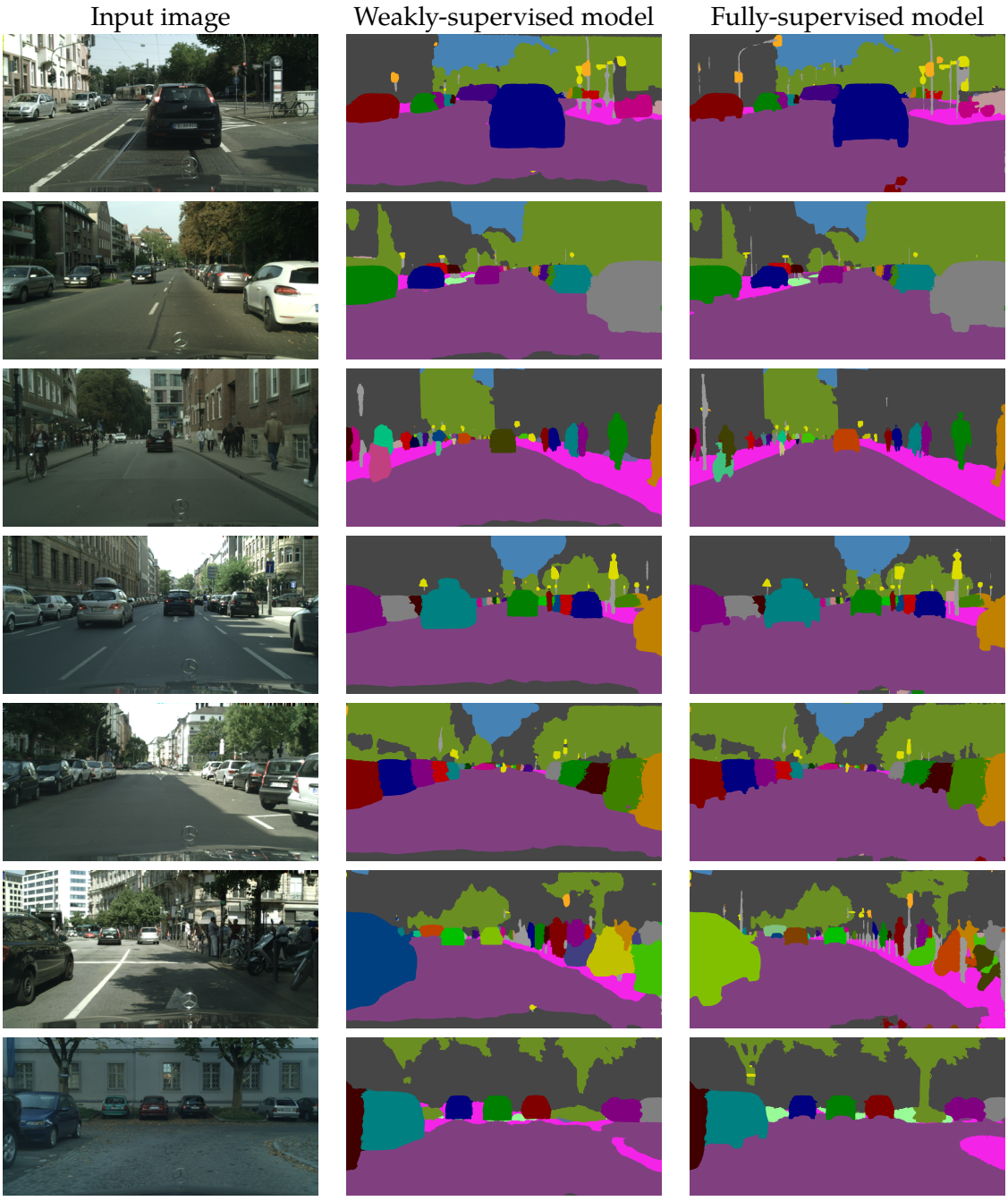


Figure 5.7: Comparison of our weakly- and fully-supervised instance segmentation models on the Cityscapes dataset. The fully-supervised model produces more precise segmentations, as seen by its sharper boundaries. The last row also shows how the fully-supervised model segments "stuff" classes such as "vegetation" and "sidewalk" more accurately. Both of these were expected, as the weakly-supervised model is trained only with bounding box and image tag annotations. Rows 3 and 6 also show some instances with different colouring. Each colour represents an instance ID, and a discrepancy between the two indicates that a different number of instances were segmented.

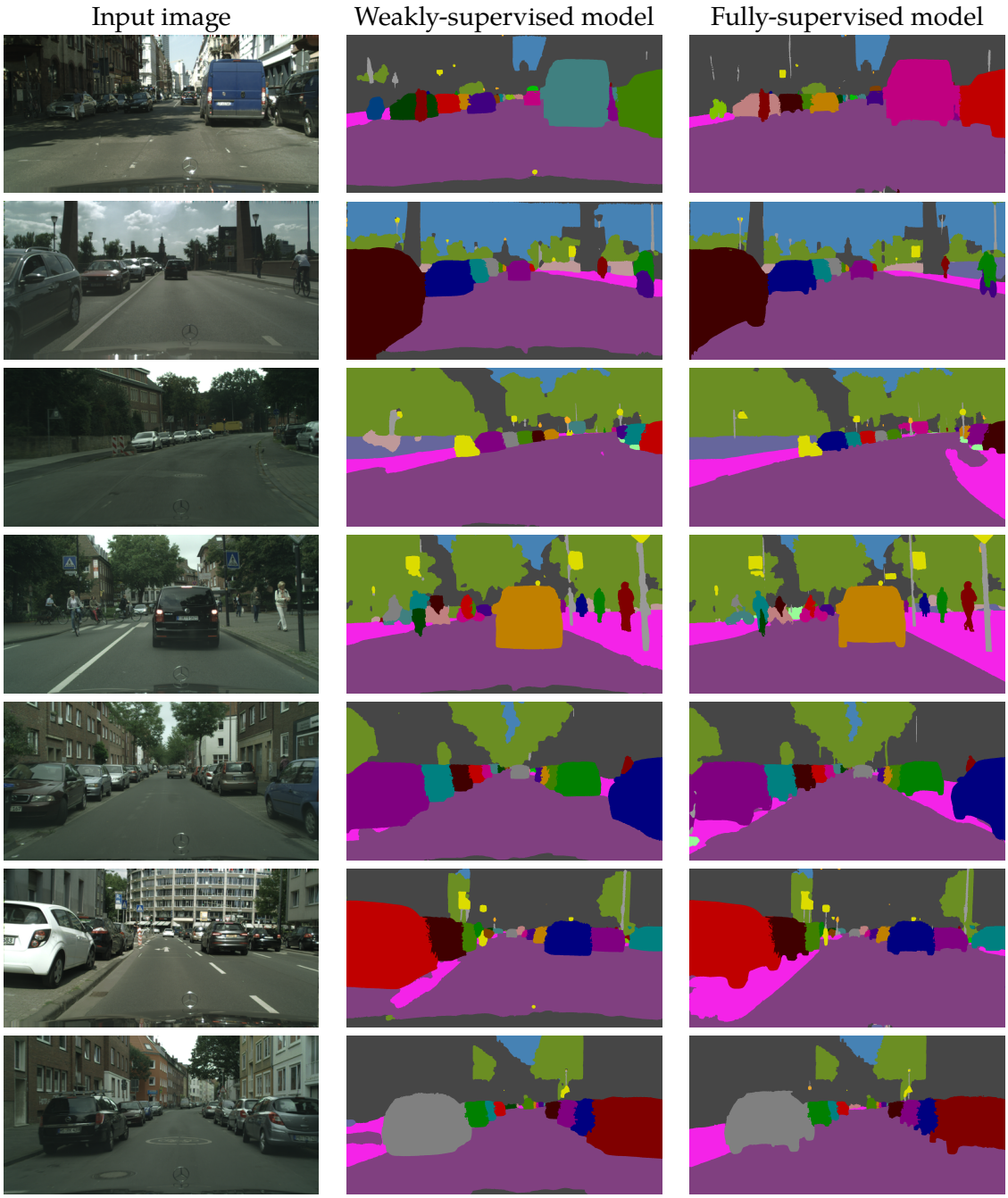


Figure 5.7 continued. Comparison of our weakly- and fully-supervised instance segmentation models on the Cityscapes dataset. The last three rows show how the fully-supervised model is also able to segment "stuff" classes such as "sidewalk" more accurately. This was expected since the weakly-supervised model is only trained with image-level tags for "stuff" classes, which provides very little localisation information.

Figure 5.8: Comparison of our weakly- and fully-supervised instance segmentation models on the Pascal VOC validation set. The weakly-supervised model typically obtains results similar to its state-of-the-art, fully-supervised counterpart. However, the fully-supervised model produces more accurate and precise segmentations, as seen in the last two rows.

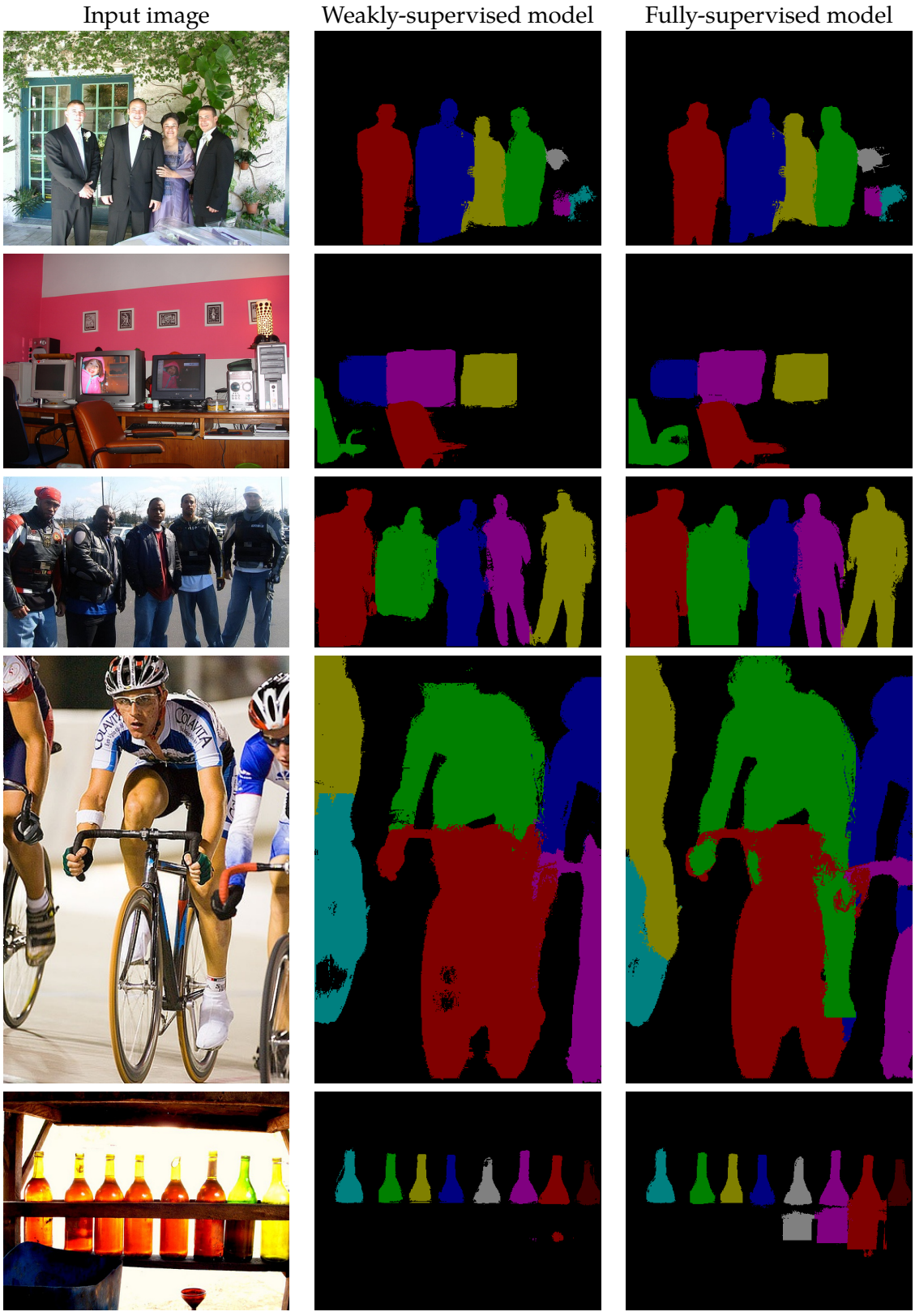


Figure 5.8 continued. The first and second rows show examples where the results of the two models are similar. In the third and fourth rows, the weakly-supervised model does not segment the "green person" as well as the fully-supervised model. In the last row, both weakly- and fully-supervised models have made an error in not completely segmenting each of the bottles.

Table 5.7: Per-class results of our weakly- and fully-supervised models for both semantic and instance segmentation on the Cityscapes validation set. The IoU measures semantic segmentation performance, whilst the AP^r_{vol} and PQ measure instance segmentation performance.

Metric	Mean	road	side-walk	build-ing	wall	fence	pole	traffic light	traffic sign	vege-tation	terrain	sky	person	rider	car	truck	bus	train	motor-cycle	bi-cycle
Weakly supervised model																				
IoU	63.6	93.3	59.3	86.6	38.7	29.6	32.0	44.0	59.2	88.7	39.1	91.7	69.4	48.4	87.4	68.0	80.7	68.0	56.0	67.5
AP^r_{vol}	26.3	82.7	27.6	68.1	5.9	5.2	0.6	3.0	16.6	74.1	4.7	76.1	11.7	5.0	27.7	17.4	36.3	23.0	9.0	5.9
PQ	40.5	91.2	47.0	79.6	14.8	12.7	5.5	13.2	37.3	83.3	16.2	82.3	30.6	25.7	46.9	33.7	55.5	37.0	31.8	24.9
Fully supervised model																				
IoU	71.6	97.6	81.9	90.4	42.2	52.3	54.5	61.1	71.8	90.5	61.1	93.5	76.6	53.2	93.4	68.3	77.8	70.6	50.7	72.3
AP^r_{vol}	34.9	94.8	56.2	73.6	10.5	7.4	11.9	10.7	31.9	77.3	16.2	78.2	21.2	15.0	32.6	25.5	41.4	30.5	15.3	12.6
PQ	47.3	95.5	67.9	83.4	17.2	15.5	38.0	22.2	54.7	84.7	21.7	80.4	40.4	37.1	49.8	31.8	54.1	36.4	34.3	32.5

Table 5.8: Per-class results of our weakly- and fully-supervised models for both semantic and instance segmentation on the Pascal VOC validation set. The IoU measures semantic segmentation performance, whilst the AP^r_{vol} and PQ measure instance segmentation performance.

Metric	Mean	aero-plane	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	motor-bike	per-son	plant	sheep	sofa	train	tv
Weakly supervised model																					
IoU	75.7	85.0	35.9	88.6	70.3	77.9	91.9	83.6	90.5	39.2	84.5	59.4	86.5	82.4	81.5	84.3	57.0	85.9	55.8	85.8	70.4
AP^r_{vol}	55.5	68.8	26.4	74.4	50.4	37.9	70.0	49.4	78.6	22.0	57.1	37.4	78.7	61.6	61.7	50.8	42.2	54.6	46.9	74.9	66.5
PQ	59.5	69.7	18.0	76.8	55.1	48.2	75.4	54.9	77.8	26.4	65.8	43.6	73.8	62.9	68.9	60.8	48.7	62.9	53.7	75.9	71.4
Fully supervised model																					
IoU	79.0	92.0	42.2	90.6	71.1	80.7	95.0	88.5	91.9	41.5	90.6	60.3	86.5	88.3	85.4	86.9	61.7	91.6	53.3	89.2	76.8
AP^r_{vol}	59.5	77.1	31.7	78.1	50.9	40.2	72.4	52.6	82.9	27.0	60.3	35.4	83.1	65.4	72.3	57.3	45.6	56.4	49.7	80.1	71.3
PQ	63.1	77.8	29.1	79.0	57.2	48.9	75.5	59.8	81.7	31.8	67.3	46.2	77.3	69.0	75.3	64.8	52.2	62.0	54.6	79.8	73.7

5.B Experimental Details

5.B.1 Network architecture and training

The underlying semantic segmentation network is a reimplementation of PSPNet [328] as described in Sec. 5.3.5, using a ResNet-101 backbone. This network has an output stride of 8, meaning that the result of the network has to be upsampled by a factor of 8 to obtain the final prediction at the original resolution.

We used most of the same training hyperparameters for training both our fully- and weakly-supervised networks. A batch size of a single 521×521 image crop, momentum of 0.9, and a weight decay of 5×10^{-4} were used in all our experiments.

We trained the semantic segmentation module first, and finetuned the entire instance segmentation network afterwards. For training the semantic segmentation module, the fully supervised models were trained with an initial learning rate of 1×10^{-4}, which was then reduced to 1×10^{-5} when the training loss converged. We used the same learning rate schedule for our weakly-supervised model on Pascal VOC where we did not do any iterative training. In total, about 400k iterations of training were performed. When training our weakly-supervised model iteratively on Cityscapes, we used an initial learning rate of 1×10^{-4} which was then halved for each subsequent stage of iterative training. Each of these iterative training stages were 150k iterations long. Both of the weakly- and fully-supervised models were initialised with ImageNet-pretrained weights and batch normalisation statistics.

In the instance training stage, we fixed the learning rate to 1×10^{-5} for both weakly- and fully-supervised experiments on the VOC and Cityscapes datasets. We observed that a total of 400k iterations were required for the models' training losses to converge.

When training the Faster-RCNN object detector [243], we used all the default training hyperparameters in the publicly available code.

5.B.2 Multi-label classification network

We obtained weak localisation cues, as described in Sec. 5.3.3 of the main paper, by first training a network to perform multi-label classification on the Cityscapes dataset.

We adapted the same PSPNet [328] architecture for segmentation for the classification task: The output of the last convolutional layer (`conv5_4`) is followed by a global average pooling layer to aggregate all the spatial information. Thereafter, a fully-connected layer with 19 outputs (the number of classes in the Cityscapes dataset) is appended. This network was then trained with a binary cross entropy loss for each of the 19 labels in the dataset.

The loss for a single image is

$$L = \frac{1}{N}\sum_{i=1}^{N} -y_i \log(\text{sigmoid}(z_i)) - (1 - y_i)\log(1 - \text{sigmoid}(z_i)), \tag{5.6}$$

where $\mathbf{y}$ is the ground truth image-level label vector and $y_i = 1$ if the i^{th} class is present in the image and 0 otherwise. z_i is the logit for the i^{th} class output by the final fully-connected layer in the network.

It is not possible to fit an entire 2048×1024 Cityscapes image in memory to perform multi-label classification. Using the PSPNet architecture described above (with an output stride of 8), it would take 48.8 GB of memory to train a network with a batch size of 1. Even the standard ResNet-101 architecture [117] (which has a higher output stride of 32, and thus sixteen times less spatial resolution) would take 21.7 GB of memory, which is still almost double the 12GB available in our Titan X GPU. Consequently, we took 15 fixed crops of size 500×400 from the original 2048×1024 image and trained with these crops instead. We were careful not to take random crops during training, as this could be a form of extra supervision. Instead, as we took 15 fixed crops which tile the image and derived image-level labels from them, it effectively means that in a real-world scenario annotators would be asked to annotate image-level labels for fifteen 500×400 images rather than a single 2048×1024 image.

This multi-label classification network was trained with a batch size of 1 and a fixed learning rate of 1×10^{-4} until the training loss converged. We found that this occurred after 50k iterations of training. At this point, the mean Average Precision (mAP) on the validation set was 78.8. The mAP is also used by the Pascal VOC dataset to benchmark multi-label classification [81].

5.C Comparison of Pascal VOC and Microsoft COCO annotation quality

Section 5.4.2 mentioned that images in Pascal VOC [81] are annotated at a higher quality than those in Microsoft COCO [180]. Figure 5.9 illustrates this observation. Images were randomly drawn from Microsoft COCO, and then images from Pascal VOC with the same semantic classes present are shown alongside for comparison. The polygons used to annotate the objects in COCO are evident, and the annotations at the boundaries of objects are often incorrect.

5.D Calculation of reduction factor in annotation time if only weak labels are used

The Cityscapes dataset has 11 "stuff" classes, and 8 "thing" classes annotated. Over the training and validation sets, there are an average of 17.9 instances of "thing" classes per full-resolution, 2048×1024 image.

For the calculation in Sec. 5.1, we assumed that each instance of a "thing" class is labelled with a bounding box, and that image-level tags are annotated for all present "stuff" classes. We assumed that a bounding box takes 7 seconds per instance to draw [220] and that an image-level tag takes 1 second to label [219].

Therefore the average time to annotate "thing" classes with a bounding-box is $17.9 \times 7 = 125.3$ seconds. As we took 15 fixed crops per image (as described in Sec. 5.B.2) and there are an average of 3.8 "stuff" tags per crop, the average time to annotate stuff classes is $15 \times 3.8 = 57$ seconds. This totals 182.3 seconds $= 3.0$ minutes per image. Thus the annotation time is reduced by a factor of 29.6 (since the images originally required 90 minutes to label at a pixel-level by hand [57]) if weak annotations in the form of bounding boxes and image-level tags are used.

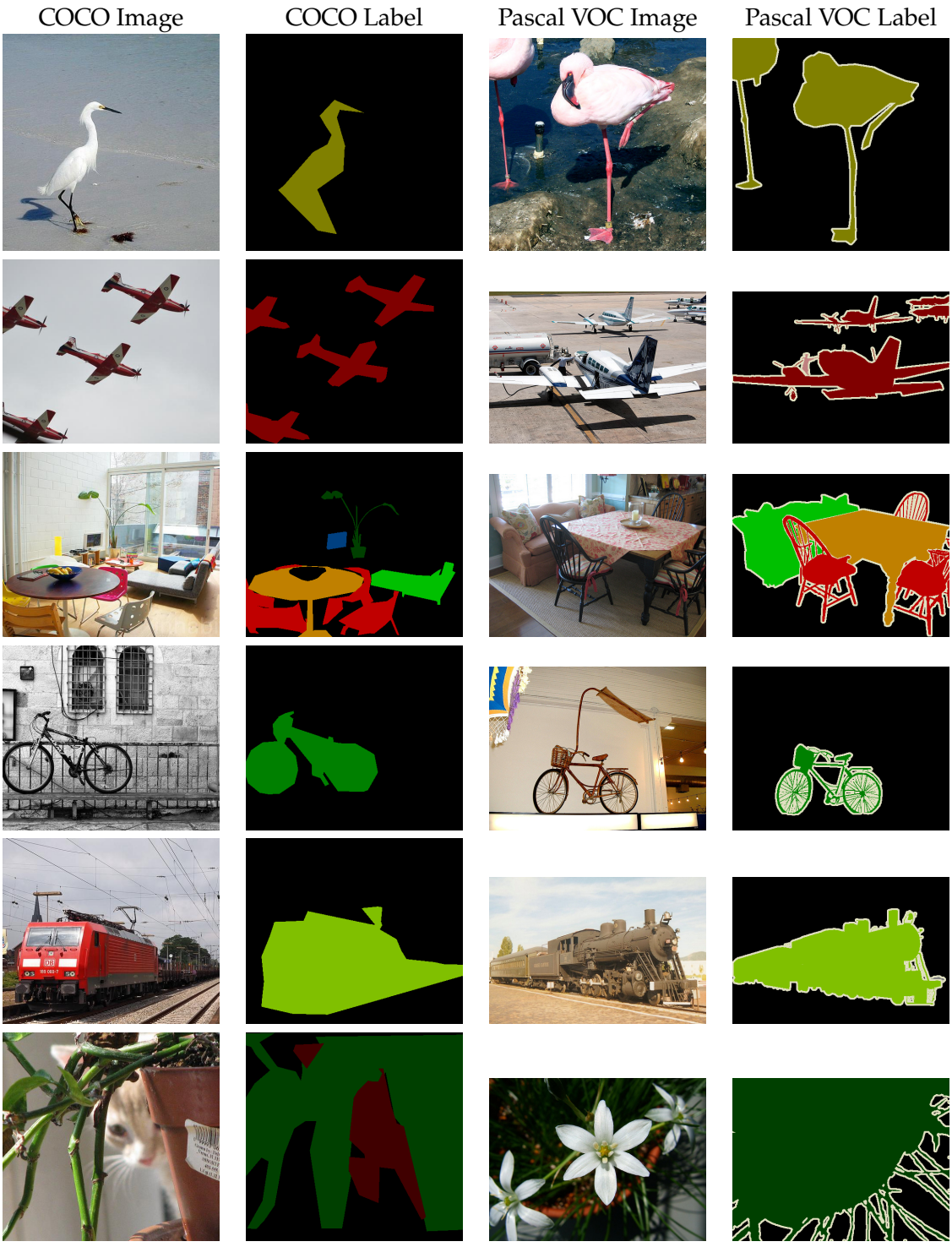


Figure 5.9: Comparison of the annotation quality of images in the Microsoft COCO and Pascal VOC datasets. An image was randomly drawn from COCO, and an image from Pascal VOC with similar content is shown alongside it. The polygons used to annotate the objects in COCO are evident, and the annotations at the boundaries of objects are often incorrect. Grey regions in the Pascal images indicate "void" regions where the annotator was unsure of the correct label.

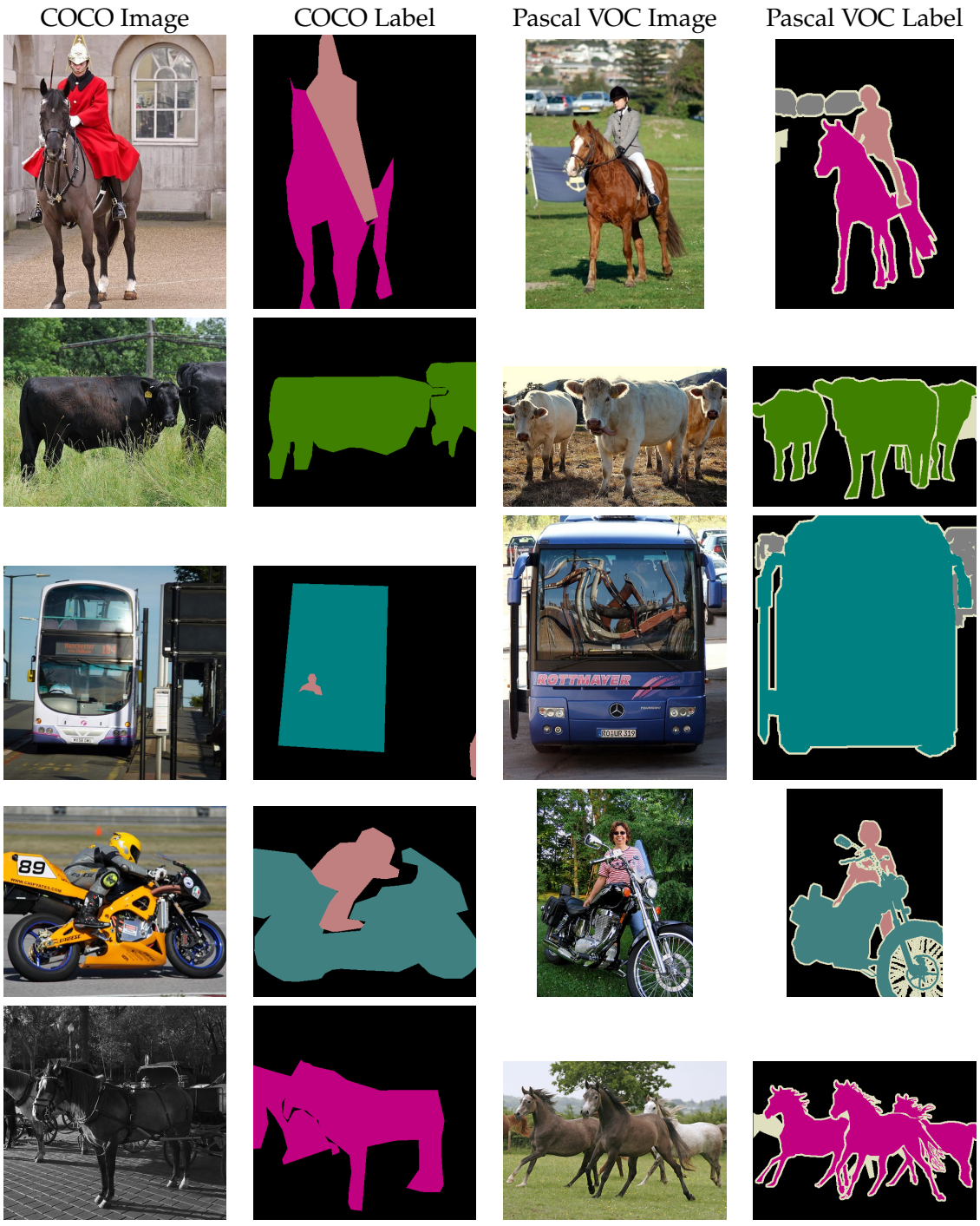


Figure 5.9 continued. Comparison of the annotation quality of images in the Microsoft COCO and Pascal VOC datasets. An image was randomly drawn from COCO, and an image from Pascal VOC with similar content is shown alongside it. The polygons used to annotate the objects in COCO are evident, and the annotations at the boundaries of objects are often incorrect. Grey regions in the Pascal images indicate "void" regions where the annotator was unsure of the correct label.

Chapter 6

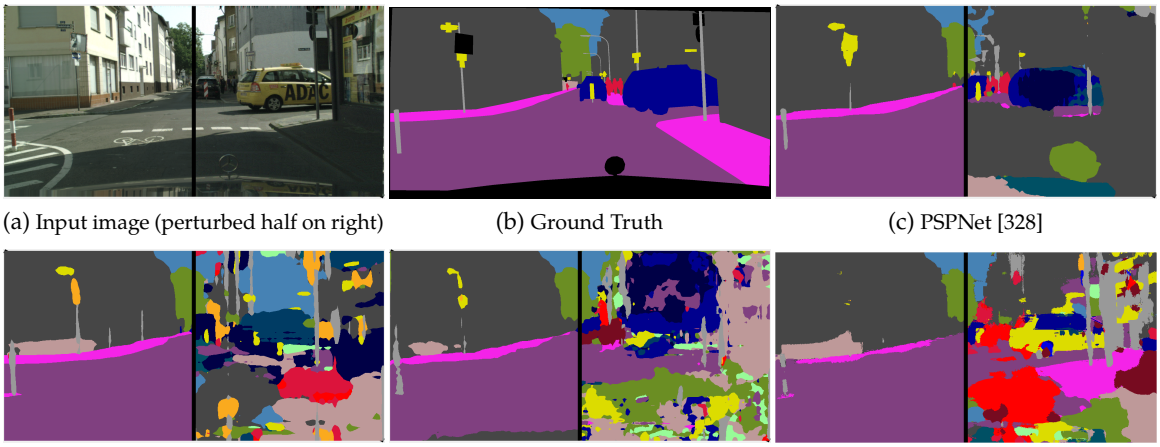
On the Robustness of Semantic Segmentation Models to Adversarial Attacks

Deep Neural Networks (DNNs) have demonstrated exceptional performance on most recognition tasks such as image classification and segmentation. However, they have also been shown to be vulnerable to adversarial examples. This phenomenon has recently attracted a lot of attention but it has not been extensively studied on multiple, large-scale datasets and structured prediction tasks such as semantic segmentation which often require more specialised networks with additional components such as CRFs, dilated convolutions, skip-connections and multiscale processing.

In this paper, we present what to our knowledge is the first rigorous evaluation of adversarial attacks on modern semantic segmentation models, using two large-scale datasets. We analyse the effect of different network architectures, model capacity and multiscale processing, and show that many observations made on the task of classification do not always transfer to this more complex task. Furthermore, we show how mean-field inference in deep structured models, multiscale processing (and more generally, input transformations) naturally implement recently proposed adversarial defenses. Our observations will aid future efforts in understanding and defending against adversarial examples. Moreover, in the shorter term, we show how to effectively benchmark robustness and show which segmentation models should currently be preferred in safety-critical applications due to their inherent robustness.

6.1 Introduction

Computer vision has progressed to the point where Deep Neural Network (DNN) models for most recognition tasks such as classification or segmentation have become a widely



(a) Input image (perturbed half on right) (b) Ground Truth (c) PSPNet [328]

(d) DilatedNet [319] (e) ICNet [327] (f) CRF-RNN [329]

Figure 6.1: The left hand side shows the original image, and the right the output when modified with imperceptible adversarial perturbations. There is a large variance in how each network's performance is degraded, even though the perturbations are created individually for each network with the same ℓ_∞ norm of 4. We rigorously analyse a diverse range of state-of-the-art segmentation networks, observing how architectural properties, such as residual connections, multiscale processing and CRFs, and input transformations, all influence adversarial robustness. These observations will help future efforts to understand and defend against adversarial examples, whilst in the short term they suggest which networks should currently be preferred in safety-critical applications.

available commodity. State-of-the-art performance on various datasets has increased at an unprecedented pace, and as a result, these models are now being deployed in more and more complex systems. However, despite DNNs performing exceptionally well in absolute performance scores, they have also been shown to be vulnerable to *adversarial examples* – images which are classified incorrectly (often with high confidence), although there is only a minimal perceptual difference with correctly classified inputs [68, 23, 278].

This raises doubts about DNNs being used in safety-critical applications such as driverless vehicles [132] or medical diagnosis [79] since the networks could inexplicably classify a natural input incorrectly although it is almost identical to examples it has classified correctly before (Fig. 6.1). Moreover, it allows for the possibility of malicious agents attacking systems that use neural networks [160, 222, 259, 82]. Hence, the robustness of networks perturbed by adversarial noise may be as important as the predictive accuracy on clean inputs. And if multiple models achieve comparable performance, we should always consider deploying the one which is inherently most robust to adversarial examples in (safety-critical) production settings.

This phenomenon has recently attracted a lot of attention and numerous strategies have been proposed to train DNNs to be more robust to adversarial examples [104, 161, 225, 196].

However, these defenses are not universal; they have frequently been found to be vulnerable to other types of attacks [36, 35, 34, 119] and/or come at the cost of performance penalties on clean inputs [39, 106, 196]. To the best of our knowledge, adversarial examples have not been extensively analysed beyond standard image classification models, and often on small datasets such as MNIST or CIFAR-10 [196, 106, 225]. Hence, the vulnerability of modern DNNs to adversarial attacks on more complex tasks such as semantic segmentation in the context of real-world datasets covering different domains remains unclear.

In this paper, we present what to our knowledge is the first rigorous evaluation of the robustness of semantic segmentation models to adversarial attacks. We focus on semantic segmentation, since it is a significantly more complex task than image classification [19]. This has also been witnessed by the fact that state-of-the-art semantic segmentation models are typically based on standard image classification architectures [157, 267, 117], extended by additional components such as dilated convolutions [44, 319], specialised pooling [43, 328], skip-connections [189], Conditional Random Fields (CRFs) [329, 8] and/or multiscale processing [43, 40] whose impact on the robustness has never been thoroughly studied.

First, we analyse the robustness of various DNN architectures to adversarial examples and show that the Deeplab v2 network [43] is significantly more robust than approaches which achieve better prediction scores on public benchmarks [328]. Thereafter, we show that adversarial examples are less effective when processed at different scales. Furthermore, multiscale networks are more robust to multiple different attacks and white-box attacks on them produce more transferable perturbations. Inspired by the effect of multiscale processing, we examine other input transformations which neural networks are not invariant to and show that they are markedly more robust to transformed adversarial examples. However, we also show that this is true only when the attack generation process does not take knowledge of these input transformations into account; otherwise, the robustness improvements are rather marginal. These observations have important implications on producing effective physical adversarial examples in the real world. On a separate track, we also show that structured prediction models have a similar effect as "gradient-masking" defense strategies [223, 225]. As such, mean field CRF inference increases robustness to untargeted adversarial attacks, but in contrast to the gradient masking defense, it also improves the network's predictive accuracy. Another of our contributions shows that some widely accepted observations about robustness and model size or iterative attacks, which were made in the context of image classification [161, 196] do not transfer to semantic segmentation and different, real-world datasets. Moreover, we also show that proposed adversarial defenses should be evaluated prudently by using knowledge of the defense mechanism in the white-box attack to test it, which was not done in previously [107, 308,

176, 55]. Finally, in contrast to the prior art [161, 187], our experiments are carried out on two large-scale, real-world datasets and (most of) our observations remained consistent across them.

We believe our findings will facilitate future efforts in understanding and defending against adversarial examples without compromising predictive accuracy.

6.2 Adversarial Examples

Adversarial perturbations cause a classifier to change its original prediction, when added to the original input $\mathbf{x}$. For a classifier f parametrised by θ that maps $\mathbf{x} \in \mathbb{R}^m$ to y, a target class from $\mathcal{C} = \{1, 2, \ldots, C\}$, a targeted adversarial attack causes the classifier to predict y_t instead, where y_t is chosen by the attacker and $y_t \neq y$. An untargeted adversarial attack causes the classifier to predict any label besides the original prediction (from the label set $\mathcal{C} \setminus \{y\}$).

This phenomenon was initially studied in the context of malware detection and spam classification [23, 68], and has recently become popular in the context of computer vision. Szegedy *et al.* [278] defined an adversarial perturbation $\mathbf{r}$ as the solution to the optimisation problem defining a targeted attack

$$\arg\min \quad \|\mathbf{r}\|_2 \qquad \text{subject to} \quad f(\mathbf{x} + \mathbf{r}; \theta) = y_t, \tag{6.1}$$

where y_t is the target label of the adversarial example $\mathbf{x}^{adv} = \mathbf{x} + \mathbf{r}$. For clarity of exposition, we consider only a single label y. This naturally generalises to the case of semantic segmentation where networks are trained with an independent cross-entropy loss at each pixel.

Constraining the neural network to output y is difficult to optimise. Hence, [278] added an additional term to the objective based on the loss function used to train the network

$$\arg\min_{\mathbf{r}} \quad \lambda \|\mathbf{r}\|_2 + L(f(\mathbf{x} + \mathbf{r}; \theta), y_t). \tag{6.2}$$

Here, L is the loss function between the network prediction and desired target, and λ is a positive scalar. Szegedy *et al.* [278] solved this using L-BFGS, and [36] and [54] have proposed further advances using surrogate loss functions. However, this method is computationally very expensive as it requires several minutes to produce a single attack. Hence, the following methods are used in practice:

Fast Gradient Sign Method (FGSM) [104]. FGSM produces adversarial examples by increasing the loss (usually the cross-entropy) of the network on the input $\mathbf{x}$ as

$$\mathbf{x}^{adv} = \mathbf{x} + \epsilon \cdot \text{sign}(\nabla_{\mathbf{x}} L(f(\mathbf{x}; \theta), y)). \tag{6.3}$$

This is a single-step, untargeted attack, which approximately minimises the ℓ_∞ norm of the perturbation bounded by the parameter ϵ.

FGSM ll [161]. This single-step attack encourages the network to classify the adversarial example as y_t by assigning

$$\mathbf{x}^{adv} = \mathbf{x} - \epsilon \cdot \text{sign}(\nabla_{\mathbf{x}} L(f(\mathbf{x}; \theta), y_t)). \tag{6.4}$$

We follow the convention of choosing the target class as the least likely class predicted by the network [161].

Iterative FGSM [161, 196]. This attack extends FGSM by applying it in an iterative manner, which increases the chance of fooling the original network. Using the subscript to denote the iteration number, this can be written as

$$\begin{aligned} \mathbf{x}_0^{adv} &= \mathbf{x} \\ \mathbf{x}_{t+1}^{adv} &= \text{clip}(\mathbf{x}_t^{adv} + \alpha \cdot \text{sign}(\nabla_{\mathbf{x}_t^{adv}} L(f(\mathbf{x}_t^{adv}; \theta), y)), \epsilon) \end{aligned} \tag{6.5}$$

The clip$(\mathbf{a}, \epsilon)$ function makes sure that each element a_i of $\mathbf{a}$ is in the range $[a_i - \epsilon, a_i + \epsilon]$. This ensures that the max-norm constraint of each component of the perturbation $\mathbf{r}$, being no greater than ϵ is maintained. It thus corresponds to projected gradient descent [196], with step-size α, into an ℓ_∞ ball of radius ϵ around the input $\mathbf{x}$.

Iterative FGSM ll [161]. This is a stronger version of FGSM ll. This attack sets the target to be the least likely class predicted by the network, y_{ll}, in each iteration

$$\mathbf{x}_{t+1}^{adv} = \text{clip}(\mathbf{x}_t^{adv} - \alpha \cdot \text{sign}(\nabla_{\mathbf{x}_t^{adv}} L(f(\mathbf{x}_t^{adv}; \theta), y_{ll})), \epsilon). \tag{6.6}$$

The aforementioned attacks were all proposed in the context of image classification, but they have been adapted to the problems of semantic segmentation [90, 54], object detection [309] and visual question answering [314]. Similar, gradient-based attacks have also been proposed to minimise the ℓ_2 norm of the adversarial perturbation, $\mathbf{r}$, [206, 36], and also to attack other classification algorithms such as SVMs [23]. Methods to optimise the non-differentiable ℓ_0 norm of the perturbation have also been proposed [273, 224, 213].

6.3 Adversarial Defenses and Evaluations

Liu *et al.* [187] have thoroughly evaluated the transferability of adversarial examples generated on one network and tested on another unknown model, *i.e.* only as "black-box"

attacks [278, 223, 201, 205]. Kurakin *et al.* [161], contrastingly, studied the adversarial training defense, which generates adversarial examples online and adds them into the training set [104, 196, 284]. They found that training with adversarial examples generated by single-step methods conferred robustness to other single-step attacks with negligible performance difference to normally trained networks on clean inputs. However, the adversarially trained network was still as vulnerable to iterative attacks as standard models. Madry *et al.* [196], conversely, found robustness to iterative attacks by adversarial training with them. However, this was only on the small MNIST dataset. The defense was not effective on CIFAR-10, underlining the importance of testing on multiple datasets. Tramer *et al.* [284] also found that adversarially trained models were still susceptible to black-box, single-step attacks generated from other networks. Other adversarial defenses based on detecting the perturbation in the input [200, 105, 87, 313, 270] or pre-processing the input [107, 176, 308, 237] have also all been subverted [11, 10, 34, 119, 35, 286]. Recently, progress has been made on formal verification of neural networks [135, 31] which can provably compute the adversarial perturbation with the minimum norm for a network. However, as these methods are limited to certain architectures, and do not scale to large networks, they cannot be used on the state-of-the-art networks we consider in this work.

Currently, no effective defense to all adversarial attacks exist. This motivates us, for the first time to our knowledge, to study the properties of state-of-the-art segmentation networks and how they affect robustness to various adversarial attacks. Previous evaluations have only considered standard classification networks (Inception in [161], and GoogleNet, VGG and ResNet in [187]). We consider the more complex task of semantic segmentation, and evaluate eight different architectures, some of them with multiple classification backbones, and show that some features of semantic segmentation models (such as CRFs and multi-scale processing) naturally implement recently proposed adversarial defenses. Moreover, our evaluation is carried out on two large-scale datasets instead of only ImageNet as [161, 187]. This allows us to show that not all previously observed empirical results on classification transfer to segmentation.

The conclusions from our evaluations may thus aid future efforts to develop defenses to adversarial attacks that preserve predictive accuracy. Moreover, our results suggests which state-of-the-art models for semantic segmentation should currently be preferred in (safety-critical) settings where both accuracy and robustness are a priority.

Note that adversarial examples have been shown to exist for semantic segmentation before by [309, 201, 54]. However, our work is complementary, as we thoroughly study the properties of semantic segmentation networks and how they affect robustness to adversarial attacks. Previous works were not as systematic as they only considered one particular

network, did not limit the norm of the adversarial perturbation and did not show how different architectural components impact adversarial robustness. Moreover, although [309] propose a gradient-based attack algorithm which considers each pixel independently, we show that similar and more common FGSM-based methods [161, 196, 104] (which [309] did not use as a baseline) are still effective.

6.4 Experimental Set-up

We describe the datasets, DNN models, adversarial attacks and evaluation metrics used for our evaluation in this section. Exhaustive details are included in the supplementary. We have also released our code[1] to aid reproducibility.

Datasets. We use the Pascal VOC [81] and Cityscapes [57] validation sets, the two most widely used semantic segmentation benchmarks. Pascal VOC consists of internet-images labelled with 21 different classes. The reduced validation [329, 189] set contains 346 images, and the training set has about 70000 images when combined with additional annotations from [110] and [180]. Cityscapes consists of road-scenes captured from car-mounted cameras and has 19 classes. The validation set has 500 images, and the training set totals about 23000 images. As this dataset provides high-resolution imagery (2048×1024 pixels) which require too much memory for some models, we have resized all images to 1024×512 when evaluating.

Models. We use a wide variety of current or previous state-of-the-art models, ranging from lightweight networks suitable for embedded applications to complex models which explicitly enforce structural constraints. Whenever possible, we have used publicly available code or trained models. The models we had to retrain achieve similar performance to the ones trained by the original authors.

We used the public models of CRF-RNN [329], DilatedNet [319], PSPNet [328] on Cityscapes, ICNet [328] and SegNet [13]. We retrained FCN [189] and E-Net [227], as well as Deeplab v2 [43] and PSPNet for VOC as the public models are trained with the validation set. Our selection of networks are based on both VGG [267] and ResNet [117] backbones, whilst E-Net and ICNet employ custom architectures for real-time applications whose parameters measure only 1.5MB and 30.1MB in 32-bit floats, respectively. Furthermore, the models we evaluate use a variety of unique approaches including dilated convolutions [319, 43], skip-connections [189], specialised pooling [328, 43], encoder-decoder architecture [13, 227], multiscale processing [43] and CRFs [329]. In all our experiments, we evaluate the model

[1] `www.robots.ox.ac.uk/~aarnab/adversarial_robustness.html`

in the same manner it was trained – CRF post-processing or multiscale ensembling is not performed unless the network incorporated CRFs [329] or multiscale averaging [43] as network layers whilst training.

Adversarial attacks. We use the FGSM, FGSM ll, Iterative FGSM and Iterative FGSM ll attacks described in Sec. 6.2. Kurakin *et al.* [161] set the number of iterations of iterative attacks to $\min(\epsilon + 4, \lceil 1.25\epsilon \rceil)$. However, we found that attacks did not always converge with this setting, and instead used $\max(\epsilon + 4, \lceil 5\epsilon \rceil)$. We set our step-size $\alpha = \min(1, \epsilon)$ meaning that the value of each pixel is changed by α (if it is not clipped due to the max-norm constraint) every iteration. The Iterative FGSM (untargeted) and FGSM ll (targeted) attacks are only reported in the supplementary as we observed similar trends on FGSM and Iterative FGSM ll. We evaluated these attacks when setting the ℓ_∞ norm of the perturbations ϵ to each value from $\{0.25, 0.5, 1, 2, 4, 8, 16, 32\}$. Even small values such as $\epsilon = 0.25$ introduce errors among all the models we evaluated. The maximum value of ϵ was chosen as 32 since the perturbation was conspicuous at this point. Qualitative examples of these attacks are shown in the supplementary.

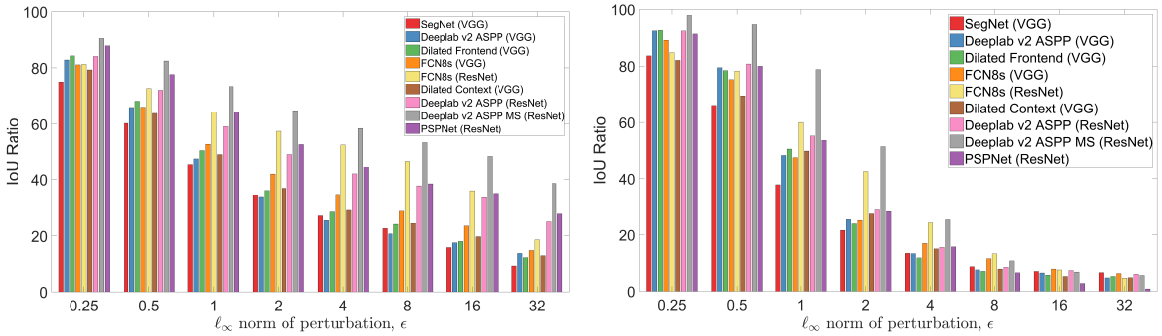
Evaluation metric. The Intersection over Union (IoU) is the primary metric used in evaluating semantic segmentation [81, 57]. However, as the accuracy of each model varies, we adapt the relative metric used by [161] for image classification and measure adversarial robustness using the *IoU Ratio* – the ratio of the network's IoU on adversarial examples to that on clean images computed over the entire dataset. As the relative ranking between models for the IoU Ratio and absolute IoU is typically the same, we report the latter only in the supplementary.

6.5 The robustness of different architectures

In this section, we evaluate the robustness of different network architectures. Our experiments show a more nuanced relationship between model capacity and adversarial robustness, by considering a different setting to the previous findings of [196, 161]. Additionally, our results also support why JPEG compression as a pre-processing step mitigates small perturbations [76].

6.5.1 The robustness of different networks

Fig. 6.2 shows the robustness of several state-of-the-art models on the VOC dataset. In general, ResNet-based models not only achieve higher accuracy on clean inputs but are also more robust to adversarial inputs. This is particularly the case for the single-step FGSM



(a) Untargeted FGSM attack

(b) Targeted Iterative FGSM ll. attack

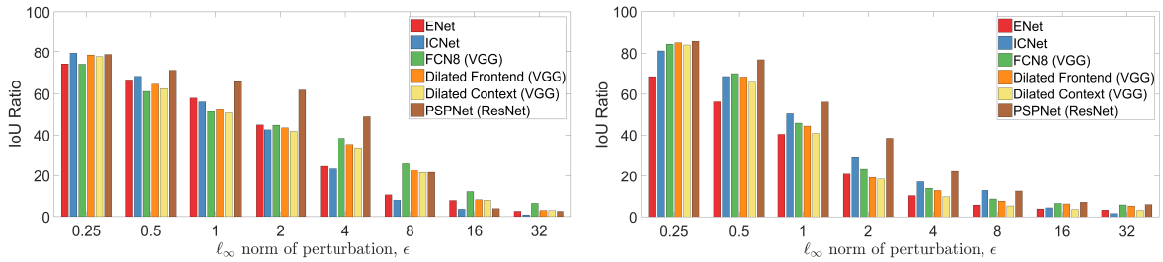
Figure 6.2: Adversarial robustness of state-of-the-art models on Pascal VOC. Models based on the ResNet backbone tend to be more robust. For instance, FCN8s and Deeplab v2 ASPP with a ResNet-101 backbone are more robust than with the VGG backbone. Moreover, as expected, the Iterative FGSM ll attack is more powerful at fooling networks than single-step FGSM. Models are ordered by increasing IoU on clean inputs. Results on additional attacks are in the supplementary.

attack (Fig. 6.2a). On the more effective Iterative FGSM ll attack, the margin between the most and least robust network is smaller as none of them perform well (Fig. 6.2b). However, we note that iterative attacks tend not to transfer to other models [161] (Sec. 6.6.2). Thus, they are less useful in practical, black-box attacks.

In particular, we have evaluated the FCN8s [189] and Deeplab-v2 with ASPP [43] models based on the popular VGG-16 [267] and ResNet-101 [117] backbones. In both cases, the ResNet variant shows greater robustness. We also observe that most of the networks achieve similar scores on clean inputs. As a result, the relative rankings of models in Fig. 6.2 for the IoU Ratio is about the same as their ranking on clean inputs. Furthermore, the best performing model on clean inputs, PSPNet [328] is actually less robust than Deeplab v2 with Multiscale ASPP [43]: For all ϵ values we tested, the absolute IoU score of Deeplab v2 was higher than PSPNet. These observations as well as results on FGSM ll and Iterative FGSM showing that the relative ranking of robustness for the different networks is similar, are detailed in the supplementary material.

6.5.2 Model capacity and residual connections

Madry *et al.* [196] and Kurakin *et al.* [161] have studied the effect of model capacity on adversarial robustness by changing the number of filters at each convolutional layer in their network, since they used the parameter count as a proxy for the model capacity. Madry *et al.* [196] observed, on MNIST and CIFAR-10, that networks trained on clean examples with a small number of parameters are the most vulnerable to adversarial examples. This observation, which suggests that small networks with few parameters are



(a) Untargeted FGSM attack

(b) Targeted Iterative FGSM ll. attack

Figure 6.3: Adversarial robustness of state-of-the-art models on the Cityscapes dataset. We observe that lightweight networks such as E-Net [227] and ICNet [327] are often about as robust as Dilated-Net [319] ($341\times$ more parameters than E-Net). Dilated-Net without its "Context" module is also slightly more robust than the full network (these findings regarding parameter count are contrary to Madry *et al.* [196] who however did not evaluate different architectures). Both attacks are very effective after $\epsilon \geq 16$, with performance of all networks degraded considerably. As with the VOC dataset, ResNet (PSPNet) architectures are more robust than VGG (Dilated-Net and FCN8).

the most vulnerable to adversarial examples, would have serious safety implications on the deployment of lightweight models, typically required in robotics, autonomous vehicles and embedded system applications. Here, we instead analyse different network structures that are used in practice (unlike [196] and [161] who used the same architecture with a different number of filters) and show in Fig. 6.3 that lightweight networks such as E-Net [227] (only 1.5 MB) and IC-Net [327] (only 30.1 MB) are affected by adversarial examples similarly as Dilated-Net [319] which has 512.6 MB in parameters (using 32-bit floats). Dilated-Net is only more robust than both of these lightweight networks for FGSM and FGSM-ll with $\epsilon \geq 4$ (which is also when perturbations become visible to the naked eye). Note that both E-Net and IC-Net have custom backbones and heavily use residual connections.

Fig. 6.3 also shows that adding the "Context" module of Dilated-Net onto the "Front-end" slightly reduces robustness across all ϵ values on both attacks on Cityscapes. Fig. 6.2 shows that this is observed for most ϵ values on VOC as well. This is even though the additional parameters of the "Context" module increases accuracy on clean inputs. Whilst models with higher capacity may be more resistant to adversarial attacks (as posited by Madry *et al.* [196]), one cannot compare the capacities of different networks, given that neither the most accurate network (PSPNet) or the network with the most parameters (Dilated-Net) are actually the most robust.

6.5.3 The unexpected effectiveness of single-step methods on Cityscapes

The single-step FGSM and FGSM ll attacks are significantly more effective on Cityscapes than on Pascal VOC. The IoU ratio for FGSM at $\epsilon = 32$ for PSPNet, Dilated Context and

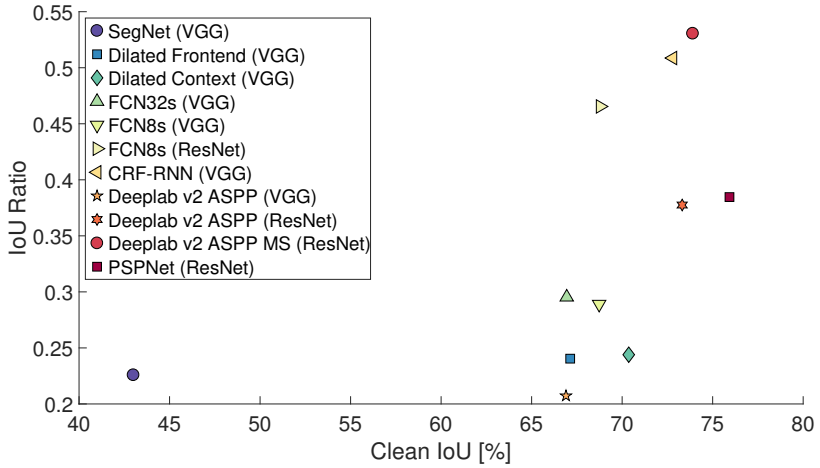


Figure 6.4: The IoU Ratio compared to the IoU on clean inputs on the Pascal VOC dataset, for the FGSM attack with $\epsilon = 8$. The relative ordering of the models is the same if we plot the absolute IoU on adversarial inputs, with the exception of SegNet which is then ranked the lowest.

FCN8s is 2.5%, 2.8% and 8.0%, respectively, on Cityscapes. On Pascal VOC, it is substantially higher at 27.9%, 12.2% and 15.0%. As expected, the iterative methods still significantly outperform single-step methods across both datasets.

Thus, it may be a dataset property that causes the network to learn weights more susceptible to single-step attacks. Cityscapes has, subjectively, less variability than VOC and it also labels "stuff" classes [91]. The effect of the training set on adversarial attacks has not been considered before, and most prior work used MNIST [278, 104, 196] or ImageNet [161, 284, 187]. However, [24] and [146], showed that the test error of an SVM and neural network could respectively be increased by inserting "poisonous" examples into its training set. Results from the FGSM ll attack, which shows the same trend as FGSM, are in the supplementary.

6.5.4 Imperceptible perturbations

With $\epsilon = 0.25$, the perturbation is so small that the RGB values of the image pixels (assuming integers $\in [0, 255]$) are usually unchanged. Nevertheless, Fig. 6.2 and 6.3 show that the performance of all analysed models were degraded by at least 3% relative IoU for each attack. The observation of [76], that lossy JPEG as a pre-processing step helps to mitigate FGSM for small ϵ is thus not surprising as JPEG does not entirely preserve these small, high-frequency perturbations and the result is also finally rounded to integers.

6.5.5 Relation with concurrent work

Our results are also corroborated by the concurrent work of Cubuk *et al.* [61] who performed Neural Architecture Search to find architectures that are more robust to adversarial examples. Cubuk *et al.* [61] found that their best architecture had more identity connections and depth than their baseline. This agrees with our observation that models based on ResNet typically have higher robustness and accuracy on clean inputs.

The authors also observed a correlation between accuracy on clean data and robustness. We also observed this correlation (Fig. 6.4), although the most accurate model on clean inputs (PSPNet) is not the most robust (Deeplab v2 Multiscale). Figure 6.4 shows the results on the FGSM attack at $\epsilon = 8$, for consistency with [61].

6.5.6 Discussion

We have shown that models with residual connections (ResNet, E-Net, ICNet) are inherently more robust than chain-like VGG-based networks, even if the number of parameters of the VGG model is orders of magnitude larger. Moreover, Dilated-Net, without its "Context" module is more robust than its more performant, full version. This is contrary to the observations regarding parameter count of [196], who noted that smaller networks were less adversarially robust on MNIST and CIFAR-10. However, a key difference between our experiments and [196, 161] is that we have considered different network architectures whilst [196] only changed the number of filters at each DNN layer. Our results in this regard are more in line with Kurakin *et al.* [161] who reported with Inception-v3 [277] based architectures on ImageNet that models that were too large or too small were less adversarially robust. The most robust model was Deeplab v2 with Multiscale ASPP, outperforming the current state-of-the-art PSPNet [328], in absolute IoU on adversarial inputs.

We also found that perturbations that do not even change the image's integral RGB values still degraded performance of all models, and that single-step attacks are significantly more effective on Cityscapes than VOC, achieving as low as 0.8% relative IoU, raising questions about how the training data of a network affects its decision boundaries. Also, explaining the effect of residual connections on adversarial robustness remains an open research question. As Deeplab v2 showed a significant increase in robustness over its single-scale variant, we analyse the effects of multiscale processing next in Sec. 6.6. Thereafter, we study CRFs, a common component in semantic segmentation models.

Table 6.1: Transferability of adversarial examples generated from different scales of Deeplab v2 (columns) and evaluated on different networks (rows). The underlined diagonals for each attack show white-box attacks. Off-diagonals, show transfer (black-box) attacks. The most effective one in bold, is typically from the multiscale version of Deeplab v2. The IoU ratio is reported.

Network evaluated	FGSM ($\epsilon = 8$)				Iterative FGSM ll ($\epsilon = 8$)			
	50%	75%	100%	Multiscale	50%	75%	100%	Multiscale
Deeplab v2 50% scale (ResNet)	<u>37.3</u>	70.5	84.8	**60.3**	<u>18.0</u>	92.0	96.9	**20.0**
Deeplab v2 75% scale (ResNet)	85.5	<u>39.7</u>	62.2	**50.8**	99.5	<u>17.9</u>	89.9	**20.4**
Deeplab v2 100% scale (ResNet)	93.6	57.9	<u>37.7</u>	**37.2**	100.0	79.0	<u>15.5</u>	**16.8**
Deeplab v2 Multiscale (ResNet)	83.7	**57.6**	62.3	<u>53.1</u>	99.6	**90.2**	91.9	<u>21.5</u>
Deeplab v2 100% scale (VGG)	94.3	70.6	66.9	**66.5**	98.9	88.4	86.3	**80.9**
FCN8 (VGG)	94.7	67.2	65.8	**65.4**	98.4	85.2	84.9	**78.5**
FCN8 (ResNet)	94.0	66.3	63.5	**63.1**	99.4	82.6	80.3	**74.1**

6.6 Multiscale Processing and Transferability of Adversarial Examples

Deeplab v2 with Multiscale ASPP was the most robust model to various attacks in Sec. 6.5, with a significant difference to its single-scale variant. In this section, we first examine the effect of multiscale processing and then relate our observations to concurrent work.

6.6.1 Multiscale processing

The Deeplab v2 network processes images at three different resolutions, 50%, 75% and 100% where the weights are shared among each of the scale branches. The results from each scale are upsampled to a common resolution, and then max-pooled such that the most confident prediction at each pixel from each of the scale branches is chosen [43]. This network is trained in this multiscale manner, although it is possible to perform this multiscale ensembling as a post-processing step at test-time only [44, 63, 178, 328].

We hypothesise that adversarial attacks, when generated at a single scale, are no longer as malignant when processed at another. This is because CNNs are not invariant to scale, and a range of other transformations [85, 229, 123]. And although it is possible to generate adversarial attacks from multiple different scales of the input, these examples may not be as effective at a single scale, making networks which process images at multiple scales more robust. We investigate the transferability of adversarial perturbations generated at one scale and evaluated at another in Sec. 6.6.2, and the robustness and transferability of multiscale networks in Sec. 6.6.3. Thereafter, we relate our findings to concurrent work.

6.6.2 The transferability of adversarial examples at different scales

Table 6.1 shows results for the FGSM and Iterative FGSM ll attacks. The diagonals show "white-box" attacks where the adversarial examples are generated from the attacked network.

These attacks typically result in the greatest performance degradation, as expected. The off-diagonals show the transferability of perturbations generated from other networks. In constrast to Iterative FGSM ll, FGSM attacks transfer well to other networks, which confirms the observations [161] made in the context of image classification.

The attack produced from 50% resolution inputs transfers poorly to other scales of Deeplab v2 and other architectures, and vice versa. This is seen by looking across the columns and rows of Tab. 6.1 respectively. All other models, FCN (VGG and ResNet) and Deeplab v2 VGG were trained at 100% resolution, and Tab. 6.1 shows that perturbations generated from the multiscale and 100% resolutions of Deeplab v2 transfer the best. This supports the hypothesis that adversarial attacks produced at one scale are not as effective when evaluated at another since CNNs are not scale invariant (the network activations change considerably).

6.6.3 Multiscale networks and adversarial examples

The multiscale version of Deeplab v2 is the most robust to white-box attacks (Tab. 6.1, Fig. 6.2) as well as perturbations generated from single-scale networks. Moreover, attacks produced from it transfer the best to other networks as well, as shown by the bolded entries. This is probably because attacks generated from this model are produced from multiple input resolutions simultaneously. For the Iterative FGSM ll attack, only the perturbations from the multiscale version of Deeplab v2 transfer well to other networks, achieving a similar IoU ratio as a white-box attack. However, this is only the case when attacking a different scale of Deeplab. Whilst perturbations from multiscale Deeplab v2 transfer better on FCN than from single-scale inputs, they are still far from the efficacy of a white-box attack (which has an IoU ratio of 15.2% on FCN-VGG and 26.4% on FCN-ResNet).

Adversarial perturbations generated from multiscale inputs to FCN8 (which has only been trained at a single scale) behave in a similar way: FCN8 with multiscale inputs is more robust to white-box attacks, and its perturbations transfer better to other networks. This suggests that the observations seen in Tab. 6.1 are not properties of training the network, but rather the fact that CNNs are not scale invariant. Furthermore, an alternative to max-pooling the predictions at each scale is to average them. Average-pooling produces similar results to max-pooling. Details of these experiments, along with results using different attacks and l_∞ norms (ϵ values), are presented in the supplementary.

6.6.4 Relation to other defenses

Our observations relate to the "random resizing" defense of [308] in concurrent work. Here, the input image is randomly resized and then classified. This defense exploits (but does

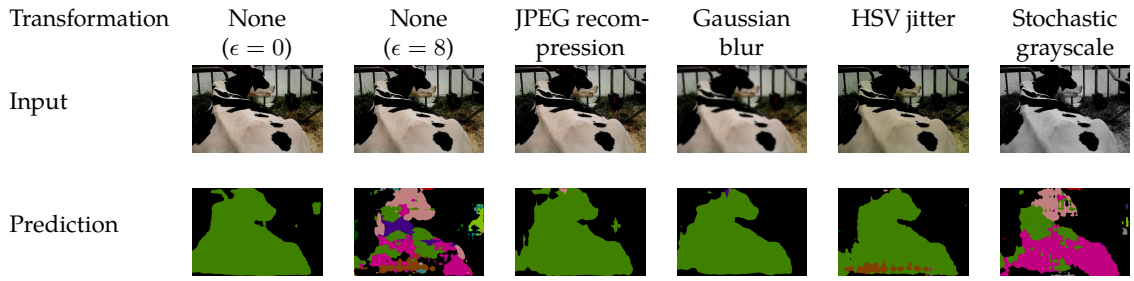


Figure 6.5: Input transformations of adversarial examples generated by Iterative FGSM ll (Eq. 6.6) significantly change the prediction of the Deeplab v2 network. These input transformations, however, barely change the output when they are applied to clean images. The l_∞ norm of the perturbation, $\epsilon = 8$, is visible when looking carefully on screen.

not attribute its efficacy to) the fact that CNNs are not scale invariant and that adversarial examples were only generated at the original scale. Our findings suggest that this defense (which is very similar to the multiscale processing performed naturally by Deeplab v2) could be defeated by creating adversarial attacks from multiple scales, as done in this work, and this has indeed been verified [11, 286].

6.7 Image transformations and adversarial examples

In Sec. 6.6, we posited that adversarial examples are less malicious when processed at different scales since CNNs are not scale invariant. Scale changes are used in segmentation architectures to recognise objects at different resolutions, however, this is not the only commonly used image transformation. In this section, we consider a number of other common input transformations, and examine their effect on adversarial robustness of CNNs for semantic segmentation.

In the following, each transformation is applied to the input image before it is processed by the neural network and we examine how it affects the robustness to adversarial examples. Following on from Sec. 6.6, we use the Deeplab v2 MS network, which we found to be the most robust in Sec. 6.5, and consider the following four transformations (illustrated in Fig. 6.5) which are ubiquitous in computer vision and image processing:

JPEG recompression. The image is compressed using JPEG with a "quality" parameter drawn randomly between 50% and 100%. The image is then reconstructed and processed by the network.

Gaussian blur. The input image is blurred by a Gaussian filter with a bandwidth uniformly drawn from $[0, 2]$, which ensures that all objects in the image are still recognisable and can be segmented precisely.

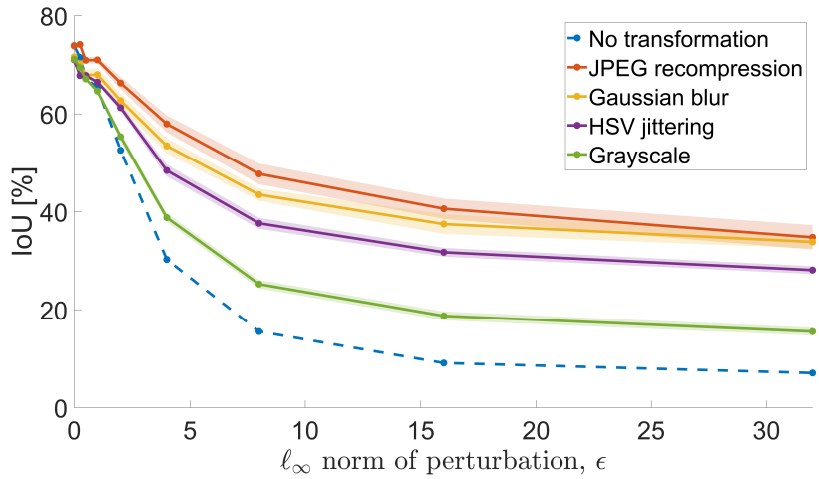


Figure 6.6: The adversarial examples originally generated by Iterative FGSM ll on Deeplab v2, are less malignant when the adversarial image is first pre-processed with a randomised transformation. The shaded regions correspond to two standard deviations computed from nine random trials of the randomised transformation.

HSV jitter. The image is converted to the HSV colour space (which is more perceptually similar than the RGB space). Next, each pixel is perturbed by a value drawn uniformly between $[-30, 30]$ and then converted back to RGB space for processing.

Grayscale. The input image is converted to grayscale by setting all three image channels to have the same value at each pixel. This was performed using a convex combination of each of the three RGB channels, with each of the co-efficients sampled from a flat Dirichlet distribution.

Note that none of the transformations affect the image spatial co-ordinates, which means that it is suitable for using with semantic segmentation models without any additional post-processing. These transformations, though quite disparate, all have a similar effect on adversarial robustness as described in the next subsection.

6.7.1 Robustness conferred by randomised input transformations

Figure 6.6 shows that each type of input transformation substantially increases the robustness of Deeplab v2 to the Iterative FGSM ll attack on the VOC dataset, with "JPEG recompression" and "Gaussian blur" providing substantial benefits. Converting the image to grayscale with random channel coefficients provides a smaller, but still sizeable, improvement. These findings are consistent and show little variance over 9 different trials, since each input transformation is randomised. The IoU of the transformed images at $\epsilon = 0$ (*i.e.* corresponding to no attack) is similar to the original image with the largest difference about

2%. Therefore, the network is more sensitive to input transformations on adversarial images than it is on clean ones.

These results, in addition to Tab. 6.1, show that as neural networks are not invariant to many classes of transformations of the input, their predictions on adversarial examples subject to these transformations change. Consequently, predictions on transformed adversarial inputs are different to the original adversarial example, and this typically results in the adversarial example becoming less malignant. These findings are consistent across a broad range of geometric and photometric transformations.

Dziugaite *et al.* [76] previously observed that JPEG recompression improved adversarial robustness for small ϵ values in the context of image classification. However, the authors hypothesised that a special property of the JPEG algorithm (*i.e.* mapping images back onto the manifold of natural images) was the reason it conferred additional robustness. In contrast, our study of various different transformations suggest that JPEG recompression is just one instance of the numerous input transformations which neural networks are not invariant to. As a result, JPEG recompression, along with other image transformations, increases robustness to adversarial examples that were generated by attacks which did not take it into account.

6.7.2 Subverting randomised, non-differentiable input transformations

The results shown in Fig. 6.6 suggest that randomised input transformations serve as an effective defense to adversarial attacks. They significantly reduced the effectiveness of the Iterative FGSM ll attack, which has been the most powerful attack in our experiments, and the result for $\epsilon = 0$ also shows that this method has minimal performance penalties on clean inputs. This reasoning has been exploited by the concurrent work of [107], where the authors showed how several different input transformations increased the robustness of image classification models to adversarial attacks.

However, the results in Fig. 6.6 and [107] assume that knowledge of the defence mechanism (randomised input transformations in this case) is not exploited in generating the adversarial attack. This methodology goes against Kerckhoffs' principle [138] – the basis of modern cryptographic systems – which states that a system should be secure if everything about it barring the key is public knowledge.

Consequently, to confirm if randomised input transformations really confer adversarial robustness, we modify the Iterative FGSM ll update (Eq. 6.6) to compute the expected gradient over the distribution of transformations which could be applied at inference time,

$$\mathbf{x}^{\text{adv}}_{t+1} = \text{clip}\left(\mathbf{x}^{\text{adv}}_{t} - \alpha \cdot \text{sign}(\mathbb{E}_{t\sim\mathcal{T}}\nabla_{\mathbf{x}^{\text{adv}}_{t}} L(f(t(\mathbf{x}^{\text{adv}}_{t});\theta), y_{ll}), \epsilon)\right), \tag{6.7}$$

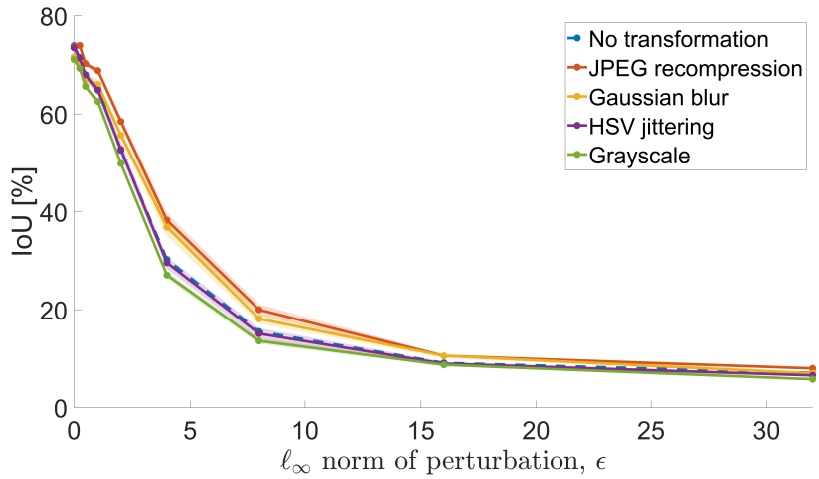
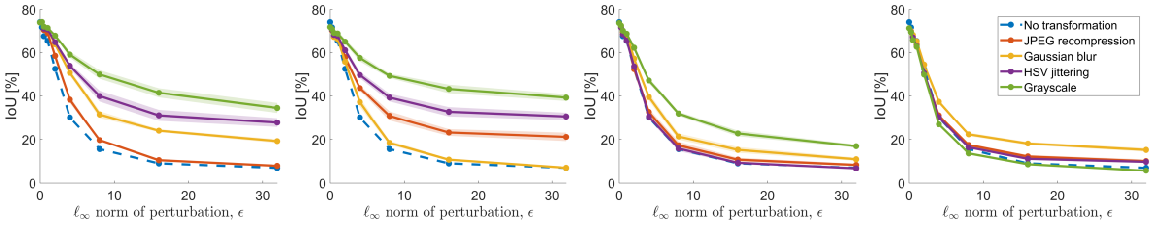


Figure 6.7: The randomised input transformations no longer increase the robustness of the network when the expected gradient over the distribution of the transformation functions is used in the Iterative FGSM ll attack. The shaded regions correspond to two standard deviations computed from nine random trials of the randomised transformation. The dashed blue line shows the original Iterative FGSM ll attack on non-transformed images.

where $\mathcal{T}$ is the distribution of transformation functions t. This method uses the fact that $\nabla_{\mathbf{x}} \mathbb{E}_{t\sim\mathcal{T}} f(t(x)) = \mathbb{E}_{t\sim\mathcal{T}} \nabla_{\mathbf{x}} f(t(x))$. It has also been used by [11] to estimate the gradient of neural networks with randomised non-differentiable adversarial defences [308]. This variant of the FGSM attack corresponds to sampling from the distribution of transformations, computing the loss and gradient of the image with respect to the loss, and averaging this gradient over many samples before performing the update.

Note that some transformations, such as JPEG recompression, are not differentiable. In this case, we use the straight-through estimator [21] which assumes, when computing the gradient using backpropagation, that the transformation is the identity function.

Figure 6.7 shows the results of the Expectation over Transformations (EOT) attack (Eq. 6.7) on the Deeplab v2 model on the Pascal VOC dataset, with the expectation computed over 10 samples. The randomised JPEG and Gaussian blur input transformations increase the robustness of the model marginally, whilst jittering pixel values in the HSV space and grayscale conversion provide no additional robustness. The final IoU is similar to the original model that did not use randomised input transformations and was attacked with the standard Iterative FGSM ll attack. To our knowledge, we are the first who show that neural networks can easily be attacked with both non-differentiable and randomised input transformations. However, we point out that [11] have attacked numerous recent defenses, some which were either non-differentiable or randomised, but not both.



(a) JPEG recompression (b) Gaussian blur (c) HSV jitter (d) Stochastic grayscale

Figure 6.8: The effectiveness of adversarial examples generated with one distribution of input transformations, and evaluated with another. The title of each graph shows the input transformation the adversarial examples were generated with. Each graph is effectively a column of Tab. 6.2 for multiple ϵ values. The dotted blue line shows the Iterative FGSM ll attack when input transformations are not used at either inference or attack generation time.

6.7.3 Transferability of input transformations

The previous two parts have shown that using input transformations reduces the malignancy of an adversarial perturbation (Sec. 6.7.1). Our second observation, however, showed that whenever we exploit knowledge about the input transformation during attack generation, the perturbation can become as malignant as the attack on the image with no input transformation (Sec. 6.7.2).

In this section, we examine the transferability of the perturbations generated from different transformations as described in Sec. 6.7.2. For example, we consider the efficacy of a perturbation created using the "JPEG recompression" transformation when the network's input is pre-processed with "Gaussian blur" instead. This has important implications on the robustness and security of neural networks; if the perturbations do not transfer across different input transformations, it would suggest that a "security-through-obscurity" approach could be used, as a defender could secure their system by ensuring that the attacker does not know the input transformations they are using. It also has implications on our ability to produce malicious physical adversarial examples [160, 259], as physical objects in the real world can be viewed from a diverse range of illumination conditions, camera viewpoints and other transformations of an original canonical view.

Table 6.2 and Fig. 6.8 show our results when the adversarial perturbation generated using one distribution of transformations is applied on a network using another randomised transformation as pre-processing. Table 6.2 shows the absolute IoU (to account for the fact that input transformations cause slight changes on the accuracy of clean inputs) for $\epsilon = 8$, which is when the adversarial perturbations become conspicuous to the human eye, whilst Fig. 6.8 summarises the results for all ϵ values. Perturbations generated to target "JPEG recompression" or "Gaussian blur" input pre-processing (the two transformations which confer the most robustness to standard attacks generated without transformations (Fig. 6.6)),

Table 6.2: Transferability of adversarial attacks generated with different input transformation distributions. The left column indicates the distribution of transformations (as described in Sec. 6.7) that was used at inference time, and the other columns show the input transformations used when generating the attack. This table shows the mean absolute IoU scores of the Deeplab v2 network on the VOC dataset for the Iterative FGSM ll attack with $\epsilon = 8$. The diagonals show "white-box" entries where the input transformation distribution used at inference time is used to generate the attack as well. The bold entries off the diagonals show the strongest attack when a different transformation distribution is used at inference time.

Input transformation at inference time	Input transformation to generate attack				
	JPEG recompression	Gaussian blur	HSV jittering	Stochastic grayscale	None
JPEG recompression	19.7	30.9	17.2	**17.4**	47.7
Gaussian blur	31.6	18.4	**21.3**	22.4	43.5
HSV jitter	39.9	39.2	15.7	**16.3**	33.5
Stochastic grayscale	50.0	49.3	32.0	13.6	**25.2**
None	**11.6**	14.4	12.0	24.4	15.5

show poor transferability when the "Grayscale" or "HSV jitter" input transformation is used instead. In contrast, perturbations generated to target the "Grayscale" input transformation transfer the best to the other input transformations that we have considered in our experiments. Additionally, the last row of Tab. 6.2 shows that when no input transformation is used at inference time, attacks generated to target a particular input transformation are more effective with the exception of the "Grayscale" transformation. This corresponds with our results in Sec. 6.6 where adversarial attacks generated at multiple scales transferred better to other models.

There are clearly a myriad of input transformations that could be performed as input pre-processing to a neural network, of which we have considered only a handful. Nevertheless, it is evident that targeting some input transformations (such as grayscale conversion) appears to produce perturbations that are more transferable to other input transformations in comparison to others (JPEG recompression). This raises an important research question about why including certain input transformations into the attack generation process transfer better to other input transformations. It also suggests another critical and open question, whether it is possible to produce adversarial perturbations that are malignant across all input transformations without modelling all of these transformations explicitly when generating the attack.

6.7.4 Relation to concurrent work

Our findings corroborate with concurrent work discussing producing physical adversarial attacks. Lu *et al.* [192] created adversarial traffic signs by capturing images of road signs from 0.5m and 1.5m away, generating attacks from these images on a computer, and then

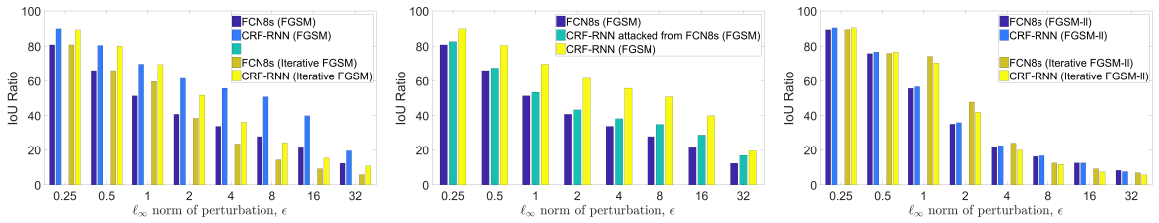
printing out the adversarial image onto paper. Whilst the printed image taken from 0.5m away fooled an object detector viewing the adversarial image from 0.5m, it did not when viewed from 1.5m and vice versa. This result is corroborated by Tab. 6.1 which shows that adversarial examples transfer poorly across different scales. Subsequent work [12, 82] has shown that it is possible to construct adversarial examples that are malignant across multiple different scales by incorporating scale changes into the attack generation process. This is again supported by our results in Tab. 6.1, and Sec. 6.7.2 which also show this effect for a number of other input transformations. When producing physical adversarial attacks, it is difficult to model all the transformations that the original image could be subject to, and as reflected by Sec. 6.7.3, adversarial examples generated to target a particular transformation do not always transfer well to other input transformations. This may explain why the adversarial traffic signs generated by [82] have not been able to fool the detectors subsequently evaluated by Lu *et al.* [193]. Our observation that input transformations that were not explicitly modelled in the attack generation process mitigate the effectiveness of adversarial attacks also suggest that future work on physical adversarial attacks requires much more robust evaluation than initial work in this area [160, 82, 192, 29]. This is to ascertain whether the proposed attacks are still effective in the diverse environmental conditions that images of the adversarial object may be acquired from.

Our study of the effect of input transformations on adversarial robustness also emphasises the importance of incorporating knowledge of the proposed adversarial defence into the attack used to validate it (Kerckhoff's principle [138]). This is not the case for many recently proposed defenses [107, 308, 30, 176] which have all subsequently been defeated [34, 11, 286, 10].

6.8 Effect of CRFs on Adversarial Robustness

Conditional Random Fields (CRFs) are commonly used in semantic segmentation to enforce structural constraints [6]. The most common formulation is DenseCRF [154], which encourages nearby (in terms of position or appearance) pixels to take on the same label and hence prefers smooth labelling. This is done by a pairwise potential function, defined between each pair of pixels, which takes the form of a weighted sum of a bilateral and Gaussian filter.

Intuitively, one may observe that adversarial perturbations typically appear as a high-frequency noise, and thus the pairwise terms of DenseCRF which act as a low-pass filter, may provide resistance to adversarial examples. To verify this hypothesis, we consider



(a) (b) (c)

Figure 6.9: (a) On untargetted attacks on Pascal VOC, CRF-RNN is noticably more robust than FCN8s. (b) CRF-RNN is more vulnerable to black-box attacks from FCN8, due to its "gradient masking" effect which results in ineffective white-box attacks. (c) However, the CRF does not "mask" the gradient for targeted attacks and it is no more robust than FCN8s.

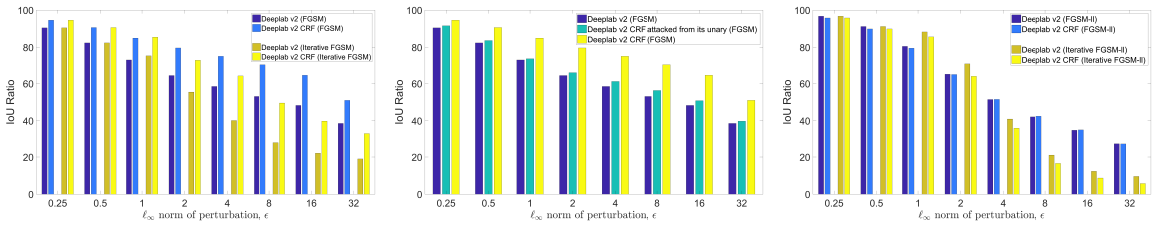
CRF-RNN [329]. This approach formulates mean-field inference of DenseCRF as an RNN which is appended to the FCN8s network [189], enabling end-to-end training.

6.8.1 CRFs confer robustness to untargeted attacks

Fig. 6.9a shows that CRF-RNN is markedly more robust than FCN8s to the untargeted FGSM and Iterative FGSM attacks. To verify the hypothesis that the smoothing effect of the pairwise terms increases the robustness to adversarial attacks, we evaluated various values of the bandwidth hyperparameters defining the pairwise potentials (not learned; in Fig. 6.9a, we used the values of the public model).

Higher bandwidth values (increasing smoothness) do not actually lead to greater robustness. Instead, we observed a correlation between the final confidence of the predictions (from different hyperparameter settings) and robustness to adversarial examples. We measured confidence according to the probability of the highest-scoring label at each pixel, as well as the entropy of the marginal distribution over all labels at each pixel. The mean confidence and entropy for CRF-RNN (with original hyperparameters) is 99.1% and 0.025 nats respectively, whilst it is 95.2% and 0.13 nats for FCN8s (additional details in supplementary). The fact that mean-field inference tends to produce overconfident predictions has also been noted previously by [210] and [33].

More confident predictions lead to a smaller loss, making attacks which use the gradient of the loss with respect to the input less effective. The "Defensive Distillation" approach of [225] made use of a similar fact by increasing the confidence of the model's predictions, resulting in gradients of smaller norm. The key difference is that CRFs increase the confidence as a by-product of a technique generally used to improve accuracy on numerous pixel-wise labelling tasks, while the effect of [225] on accuracy is unknown, as it was only tested on the saturated MNIST and CIFAR10 datasets.



(a) (b) (c)

Figure 6.10: Similar trends are observed for Deeplab v2, which uses the DenseCRF model as post-processing, as CRF-RNN (Fig. 6.9) which integrates the CRF as part of the deep network. (a) On untargetted attacks, Deeplab v2 with a CRF is noticably more robust than just the Deeplab v2 network. (b) Attacks created from the base Deeplab v2 network using FGSM are more effective than those created from Deeplab v2 with CRF. This is due to the "gradient masking" effect of mean-field inference of CRFs. (c) However, the CRF does not "mask" the gradient for targeted attacks. As a result, Deeplab v2 with a CRF is no more robust than just the Deeplab v2 network.

6.8.2 Circumventing the CRF

Although CRFs are more resistant to untargeted attacks, they can still be subverted in two ways. CRF-RNN is effectively FCN8s with an appended mean-field layer. Fig. 6.9b shows, that adversarial examples generated via FGSM from FCN8s ("unary" potentials) are more effective on CRF-RNN than attacks from the output layer of CRF-RNN.

Also, targeted attacks with FGSM ll and Iterative FGSM ll are more effective since the label used to compute the loss for generating the adversarial example is not the network's (highly confident) prediction but rather the least likely label. Consequently, the loss is high and there is a strong gradient signal from which to compute the adversarial example. Fig. 6.9c shows that CRF-RNN and FCN8s barely differ in their adversarial robustness to targeted attacks.

Finally, Fig. 6.10 shows that the same observations hold on the DeepLab v2 network, where the DenseCRF model is used as post-processing, and is not part of the neural network. This confirms that end-to-end training of the CRF, as done in CRF-RNN [329], does not influence adversarial robustness.

6.8.3 Discussion

The smoothing effect of CRFs, perhaps counter-intuitively, has no impact on the adversarial robustness of a DNN. However, mean-field inference produces confident marginals, making untargeted attacks less effective since they rely on the gradient of the final loss with respect to the prediction. Black-box attacks generated from models without a CRF transfer well to networks with a CRF, and are actually more effective. This is the case for both CRFs

trained end-to-end [329] and used as post-processing [43], as shown in the supplementary. Finally, CRFs confer no robustness to untargeted attacks. Our investigation of the CRF also underlines the importance of testing thoroughly with black-box attacks and multiple attack algorithms, which is not the case for numerous proposed defenses [55, 94, 104, 225].

6.9 Conclusion

We have presented what to our knowledge is the first rigorous evaluation of the robustness of semantic segmentation models to adversarial attacks. We believe our main observations will facilitate future efforts to understand and defend against these attacks without compromising accuracy:

Networks with *residual connections* are inherently more robust than chain-like networks. This extends to the case of models with very few parameters, contrary to the prior observations of [161, 196] (we stress that these were however made in a different context, as they did not consider different network architectures, but only varied the number of filters per DNN layer). *Multiscale* processing makes CNNs more robust since adversarial inputs are not as malignant when processed at a different scale from which they were generated at, probably as CNNs are not invariant to scale. Using other *input transformations* that CNNs are not invariant make them markedly more robust to transformed adversarial examples but only when the attack generation does not take knowledge of these input transformations into account. This holds even when the input transformations are randomised. However, when this knowledge is taken into account during attack generation, only marginal improvements in robustness are observed. The fact that adversarial attacks generated to target particular input transformation do not always transfer well to other input transformations also suggests that producing physical adversarial attacks in varying environmental conditions is difficult.

Mean-field inference for Dense CRFs, which increases the confidence of predictions confers robustness to untargeted attacks, as it naturally performs "gradient masking" [223, 225]. There are no robustness benefits from the smoothness priors enforced by the DenseCRF model.

In the shorter term, our observations suggest that networks such as Deeplab v2, which is based on ResNet and performs multiscale processing, should be preferred in safety-critical applications due to their inherent robustness. As the most accurate network on clean inputs is not necessarily the most robust network, we recommend evaluating robustness to a variety of adversarial attacks as done in this paper to find the best combination of accuracy and robustness before deploying models in practice. We also emphasize that it is crucial to

evaluate proposed defenses judiciously, *e.g.* using the white-box attacks which exploit knowledge of the proposed defense to assess the real efficacy of such a defense.

Adversarial attacks are arguably the greatest challenge affecting DNNs. The recent interest of our field into this phenomenon is only the start of an important longer-term effort, and we should also study the influence of other factors such as training regimes and attacks tailored to evaluation metrics. In this paper, we have made numerous observations and raised questions that will aid future work in understanding adversarial examples and developing more effective defenses.

Appendices

This appendix details the DNN models we analysed, and experiments we omitted from the main paper since they follow similar trends. Section 6.A provides further details about the experimental set-up, including the various DNNs used in the experiments. Section 6.B shows qualitative examples of the adversarial attacks we studied. Section 6.C presents further experimental results about "The robustness of different networks" (Sec. 6.5). Similarly, Section 6.D shows more experimental results about "Multiscale Processing and Transferability of Adversarial Examples" (Sec. 6.6). Finally, Section 6.E presents further experimental results on the "Effect of CRFs on Adversarial Robustness" (Sec. 6.8).

6.A Experimental setup

This section details the DNN models, additional information about the Cityscapes dataset and the software and hardware used in the experiments.

6.A.1 Software and hardware setup

We use the Caffe [133] deep learning framework for all experiments, since most publicly available segmentation models are implemented using this library. Our experiments are performed on either a Nvidia M40 or P100 GPU which have 12GB and 16GB of memory respectively.

6.A.2 Description of models

We detail each model in this section. Tab. 6.3 shows the performance of publicly available models on the Pascal VOC validation set. Table 6.4 compares the Intersection over Union (IoU) obtained by models that we have retrained compared to the original author's performance where available. Table 6.5 shows the performance of publicly available models on the Cityscapes validation set. Finally, Tab. 6.6 lists the number of parameters in each of the models.

FCN8s [189]. We retrained the FCN8s (VGG) network on Pascal VOC with additional annotations from SBD [110] and MS-COCO [180]. The publicly available model of FCN8s is not trained with MS-COCO, which is why we retrained it ourselves. As shown in Tab. 6.4, we obtain an IoU of 68.7% on the VOC validation set, whilst the original authors who did not train on MS-COCO obtained 65.5% [260].

Table 6.3: Networks with public models, evaluated on the VOC validation set

Model Name	IoU [%]
CRF-RNN [329]	72.8
Dilated Frontend [319]	67.1
Dilated Context [319]	70.4
SegNet [13]	43.0

Table 6.4: Performance of retrained models on VOC validation set. Details about FCN8, Deeplab v2 and PSPNet can be found in Sec. 6.A.2.

Model Name	IoU [%]	IoU of authors [%]
FCN8s (VGG) [189]	68.7	–
FCN8s (ResNet) [189]	68.8	–
Deeplab v2 ASPP (VGG) [43]	66.9	68.9
Deeplab v2 ASPP (ResNet) [43]	73.3	–
Deeplab v2 Multiscale ASPP (ResNet) [43]	73.9	76.3
Deeplab v2 Multiscale ASPP (ResNet) + CRF post-processing [43]	74.9	77.7
PSPNet [328]	75.9	–
PSPNet [328] (test set)	79.0	85.4

For the Cityscapes dataset, we used the publicly available VGG model[2] from [261].

We trained FCN8s with a ResNet-101 backbone on Pascal VOC since no publicly available model was available. As shown in Tab. 6.4, the IoU on clean inputs of this version is close to the VGG version. We are not aware of any other published work to compare this number to.

Deeplab v2 [43]. We cannot use the publicly released models for the Pascal VOC dataset, since they have been trained on the entire validation set as well. Hence, we use the authors' publicly released training code[3] to retrain their networks without the VOC validation set.

We retrained the Deeplab v2 network with ResNet-101 and VGG backbones on Pascal VOC, achieving similar performance to the original authors as shown in Tab. 6.4. Note that the authors [43] reported results from ablation experiments on the VOC validation set, which we compare to in Tab. 6.4. However, these models have never been released.

For CRF post-processing, we used the hyperparameters used by the original authors. As the weights of our trained model are different to the authors, it is possible that different CRF hyperparameters that obtain a higher IoU on the validation set exist.

[2] `https://github.com/shelhamer/clockwork-fcn`
MD5 checksum of Caffe model: fcae4fdc759f9f461fffc7cc3baa96c6

[3] `https://bitbucket.org/aquariusjay/deeplab-public-ver2.git`

Table 6.5: Networks with public models on Cityscapes validation set. We have reported the IoU at 1024×512 inputs, as well as the original 2048×1024 if the network was trained using full-resolution crops.

Model name	IoU at 1024×512	IoU at 2048×1024
E-Net [227]	53.4	–
ICNet [327]	56.5	67.2
FCN8s (VGG) [261]	62.1	66.4
Dilated Frontend [319]	59.0	64.6
Dilated Context [319]	62.3	68.6
PSPNet [328]	74.4	79.7

PSPNet [328]. We used the publicly available model[4] for our experiments on Cityscapes. As the public VOC model has been trained on the entire validation set, we cannot use it for our experiments. Consequently, we retrained this model ourselves achieving comparable results to the original authors (Tab. 6.4). We followed the training procedure described in the original paper where possible. However, the original authors trained the model using 16 GPUs allowing an effective batch size of 16. Due to our limited computational resources, we could only train on a single GPU using a batch size of 1. The large batch size enabled the original authors to compute better batch statistics for batch normalisation. When using a batch size of 1, the variance in the batch statistics is too high to perform batch normalisation. As a result, we "froze" our batch normalisation layers, and used the batch statistics (mean and variance) of the ImageNet-pretrained ResNet-101 model. This is common practice in training semantic segmentation [43] and object detection [129] networks where batch sizes are typically small.

As shown in Tab. 6.4, our reimplementation of PSPNet on VOC achieves comparable results to the original authors, even though it has been trained on 1449 fewer images (the VOC validation set). We compared our implementation to the authors on the held-out test set (evaluation is performed on an online server) as the performance on the validation set is not reported in the original paper.

CRF-RNN [329]. We used the publicly available model for Pascal VOC (trained on MS-COCO)[5].

[4] `https://github.com/hszhao/PSPNet`
MD5 checksum of Caffe model: 29bbdf0ce4d2a6546ed473656db1d6e2

[5] `https://github.com/torrvision/crfasrnn`
MD5 checksum of Caffe model: bc4926ad00ecc9a1c627db82377ecf56

Table 6.6: The number of parameters in each of the DNN models evaluated in this paper. As all the networks are stored as 32-bit/4-byte floating point numbers, we reported the number of parameters in megabytes (MB).

Model Name	Dataset	Number of parameters (MB)
E-Net	Cityscapes	1.5
ICNet	Cityscapes	30.1
PSPNet (ResNet-101)	Cityscapes	260.2
Dilated Frontend (VGG)	Cityscapes	512.4
FCN8s (VGG)	Cityscapes	512.5
Dilated Context (VGG)	Cityscapes	512.6
Segnet (VGG)	Pascal	112.4
Deeplab v2 (VGG)	Pascal	144.5
FCN8s (ResNet-101)	Pascal	162.9
Deeplab v2 (ResNet-101)	Pascal	168.4
PSPNet (ResNet-101)	Pascal	272.7
Dilated Frontend (VGG)	Pascal	512.4
FCN8s (VGG)	Pascal	513.0
CRF-RNN (VGG)	Pascal	513.0
Dilated Context (VGG)	Pascal	538.4

DilatedNet [319]. We used the public Pascal VOC and Cityscapes models[6].

ICNet [327]. We used the public Cityscapes model[7].

E-Net [227]. We used the public Cityscapes model[8].

SegNet [13]. We used the public Pascal VOC model[9].

6.A.3 Cityscapes dataset

Table 6.5 shows the performance of various publicly available models on the Cityscapes validation set consisting of 500 images. Cityscapes images are captured at a high resolution of 2048×1024, which is too large to fit into GPU memory for most networks. With the exception of E-Net [227] (which is trained on half-resolution images), the other networks we evaluated are trained on smaller crops of full-resolution images. Thereafter, at test time,

[6] `https://github.com/fyu/dilation`.
MD5 checksum for Pascal VOC: 7a44221dbc2611529bff32029ad1f6e2
MD5 checksum for Cityscapes: 0de4d78b5f9692f2aba5e7ed88f93ccb

[7] `https://github.com/hszhao/ICNet`
MD5 checksum of Caffe model: c7038630c4b6c869afaaadd811bdb539

[8] `https://github.com/TimoSaemann/ENet`
MD5 checksum of Caffe model: d9aabd630cf6bc29c48ea55a86124e14

[9] `https://github.com/alexgkendall/SegNet-Tutorial/blob/master/Example_Models/segnet_model_zoo.md`
MD5 checksum of Caffemodel: 6e01077e3cda996f95b2a82ea4641a4c

authors use different tiling strategies [319, 328] to process parts of an image at full resolution before combining the partial results. To make a fairer comparison between models, we process all images at half-resolution so that tiling is not required. In Tab. 6.5, we show the IoU at the resolution we tested on, 1024×512. And if the model was also trained on full resolution crops, we also include the IoU of the network on full resolution inputs.

6.B Qualitative results

Figure 6.11 visualises adversarial perturbations of varying ℓ_∞ norms, showing how the perturbations only become visible to the naked eye when the l_∞ of the perturbation, ϵ, is 8 (when viewed on screen).

Figure 6.12 shows the results of the four adversarial attacks considered in this paper when applied on the same image from the Pascal VOC dataset on the Deeplab v2 network.

Finally, Fig. 6.13 compares the outputs of different networks to the Iterative FGSM ll attack for varying values of ϵ on the Cityscapes dataset.

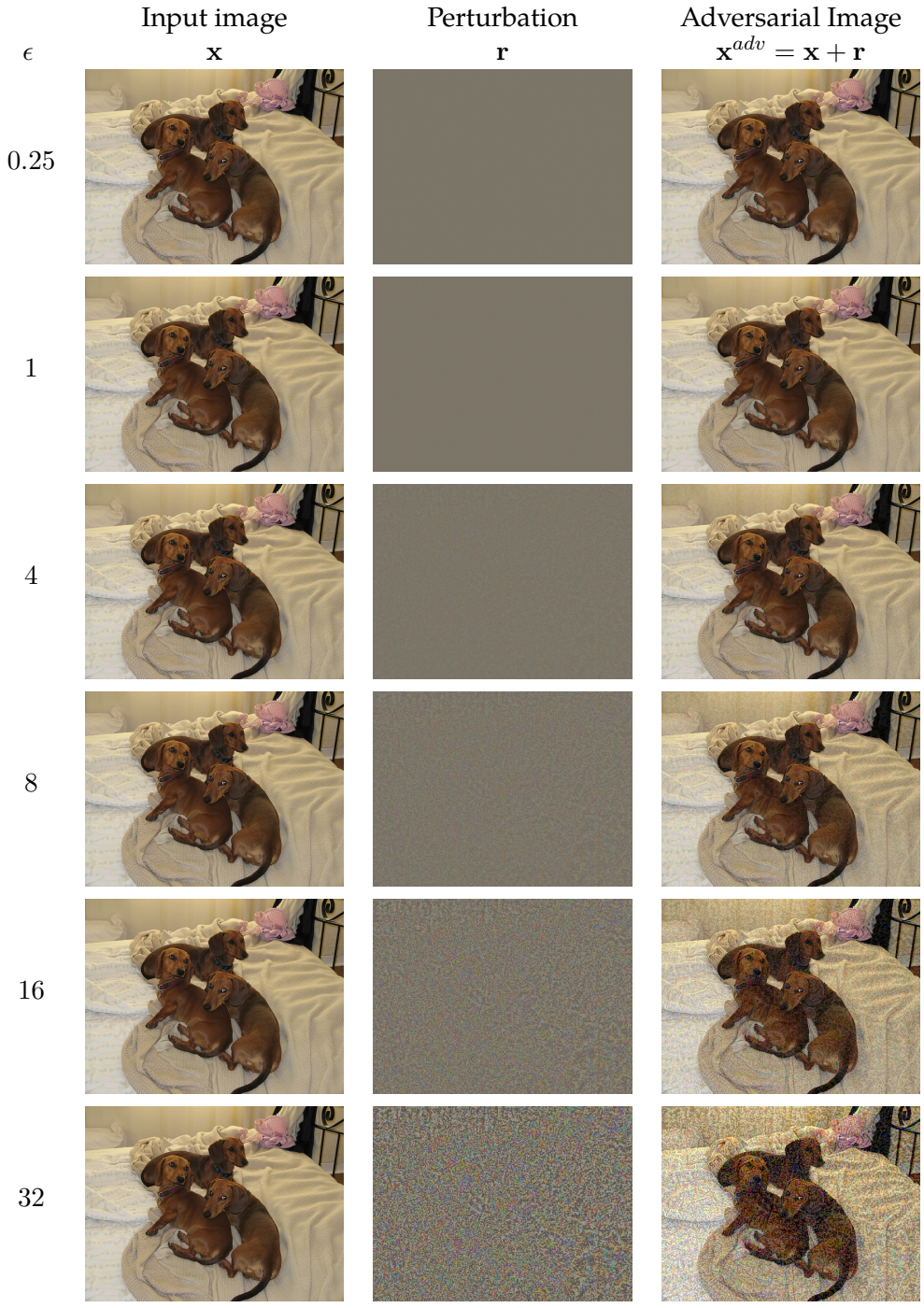


Figure 6.11: A visualisation of adversarial perturbations of varying ℓ_∞ norms. The perturbation, in the middle column, when added to the input, produces the adversarial example that fools neural networks. Note that the mean RGB value (of the Pascal VOC dataset) is already added to the perturbation, resulting in the grey background. This is required for visualisation as the perturbation can be negative, and RGB images are stored as positive integers $\in [0, 255]$. For $\epsilon = 0.25$, the adversarial image and input image are actually identical if rounded to integers (as RGB images are typically represented). Nevertheless, perturbations of this norm have fooled every neural network studied in this paper. Perturbations become noticeable when viewed on screen at around $\epsilon = 8$. In this figure, perturbations were created using FGSM on Deeplab v2.

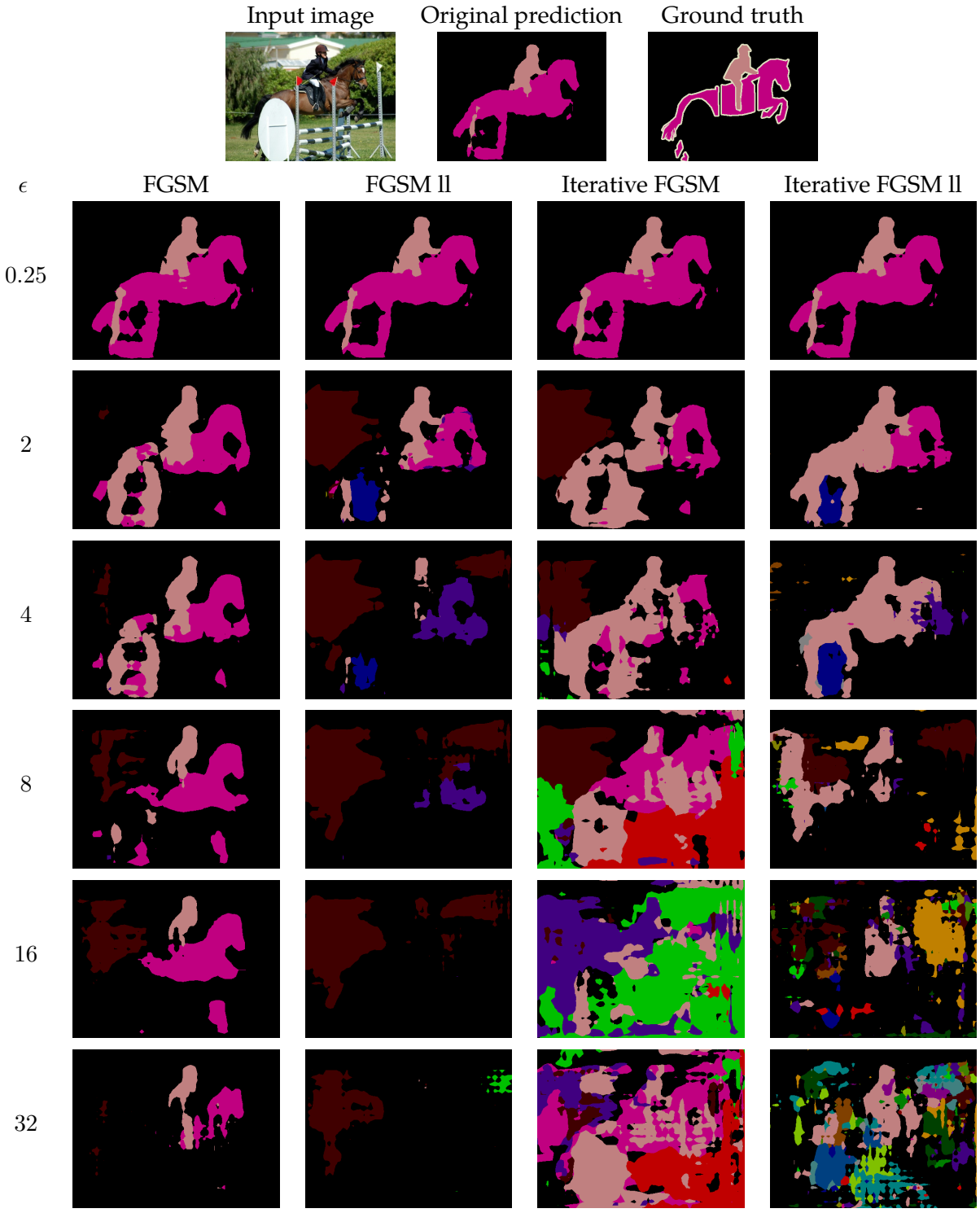


Figure 6.12: A comparison of different adversarial attacks on the Deeplab v2 Mulitscale ASPP network [43], on a common image from Pascal VOC. As expected, iterative attacks (last two columns) are more effective than single-step ones (first two columns). Higher l_∞ norms of the perturbation, ϵ, also degrade the network's prediction more.

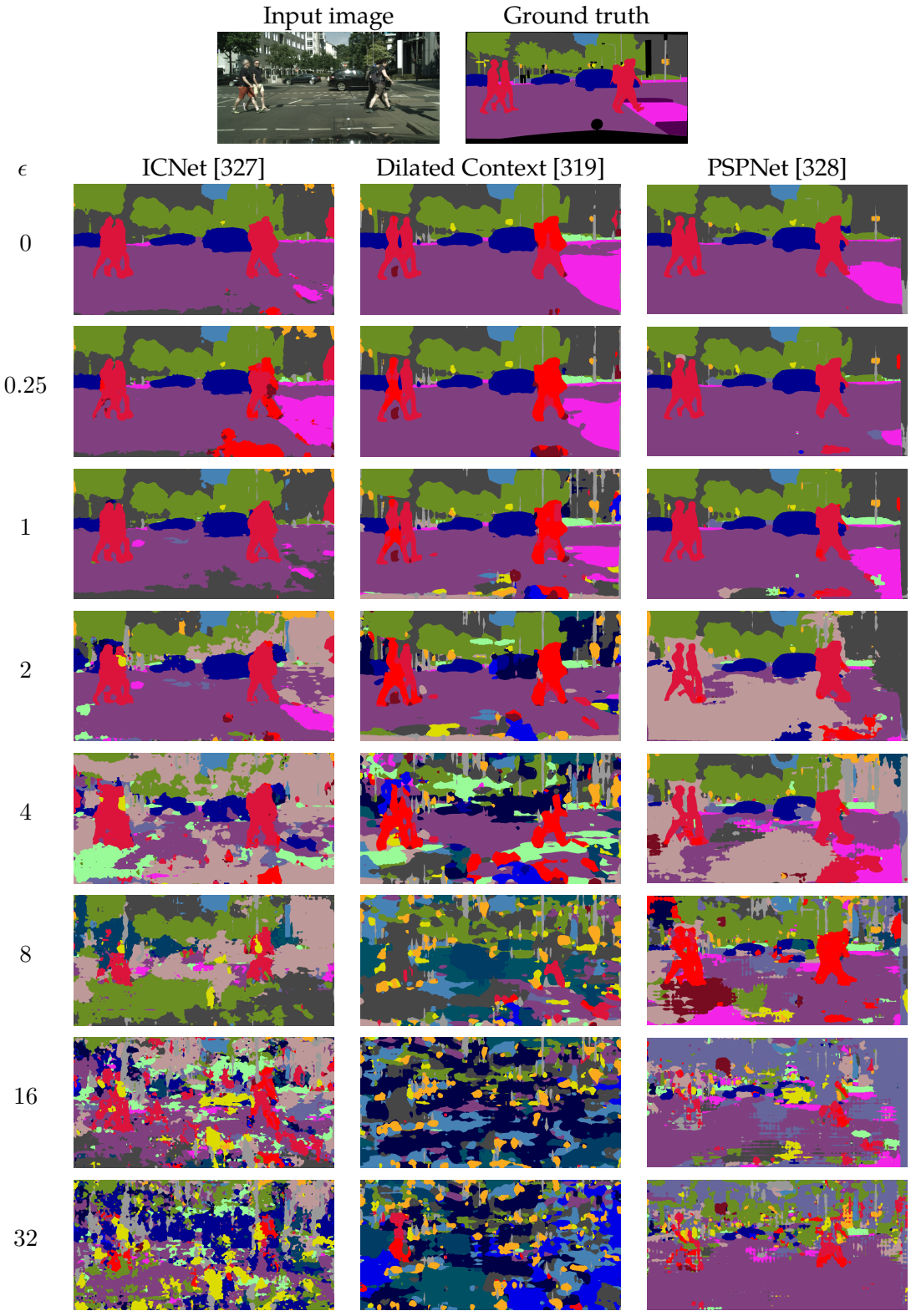


Figure 6.13: Comparison of ICNet, Dilated Context and PSPNet when attacked by Iterative FGSM ll, for different values of the l_∞ norm, ϵ. Note how each network is affected differently, with PSPNet the most robust. $\epsilon = 0$ is the original prediction of the network, since no perturbation is added here.

6.C Robustness of Different Architectures

The main paper presented results using the FGSM and Iterative FGSM ll attacks for both Pascal VOC and Cityscapes datasets. In this section, we present results for the targeted, single-step FGSM ll and untargeted Iterative FGSM attacks as well. Furthermore, we also include the Absolute IoU scores for each attack for different l_∞ perturbations.

6.C.1 Results of other attacks

Figures 6.14 and 6.15 show results of the FGSM ll and Iterative FGSM attacks on the VOC and Cityscapes datasets respectively. Our primary observations from the main paper are mostly consistent on these attacks as well:

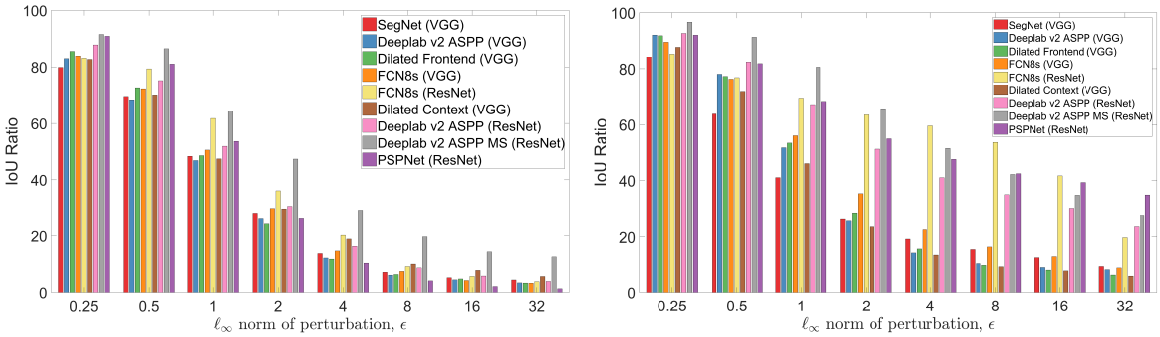
- ResNet based networks are more robust than models based on VGG.
- DilatedNet [319] without its "Context" module is typically more robust than the full, more accurate network.
- E-Net and ICNet show similar robustness to DilatedNet on the Cityscapes dataset. It is only for the FGSM ll attack for $\epsilon \geq 4$ that DilatedNet is robust than both of these lightweight networks.
- Single-step attacks (FGSM ll) are particularly effective on Cityscapes at high ϵ values.
- PSPNet, which achieves the highest IoU on clean inputs, is typically not the most robust network on Pascal VOC.

6.C.2 Result tables of Absolute IoU

In contrast to the main paper that showed the IoU Ratio for various attacks, Tables 6.7 through 6.10 show the absolute IoU for different models for each of the FGSM, FGSM ll, Iterative FGSM and Iterative FGSM ll attacks on the Pascal VOC dataset. Additionally, Tables 6.11 through 6.14 show the absolute IoU for different models on the Cityscapes dataset.

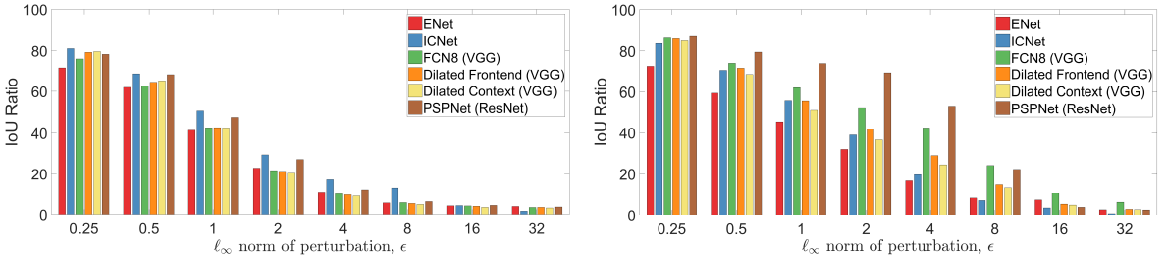
Note that PSPNet, which achieves the highest IoU on clean inputs, does not usually achieve the highest absolute IoU when attacked on the Pascal VOC dataset. When considering 4 adversarial attacks, and 8 ϵ values, PSPNet achieves the highest absolute IoU in only 2 out of 32 cases. Moreover, it never achieves the highest absolute IoU for imperceptible perturbations ($0 < \epsilon \leq 4$).

Additionally, the highest absolute IoU for any ϵ value is always from a ResNet-based model (Deeplab v2, FCN8s (ResNet) or PSPNet) on the Pascal VOC dataset. On Cityscapes, FCN8s (VGG) is sometimes the most robust network at high ϵ values. However, the performance of all the networks is severely degraded at this point.



(a) Untargeted Iterative FGSM attack (b) Targeted FGSM ll. attack

Figure 6.14: Adversarial robustness of state-of-the-art models on the Pascal VOC dataset. As with the FGSM and Iterative FGSM ll attacks in the main paper, models based on the ResNet backbone are more robust. Deeplab v2 is generally the most robust network, except on the Targeted FGSM attack for $\epsilon \geq 4$. The Iterative FGSM attack is also more effective at fooling the networks than the single-step Targeted FGSM attack, as shown by the lower IoU ratios.



(a) Untargeted Iterative FGSM attack (b) Targeted FGSM ll. attack

Figure 6.15: Adversarial robustness of state-of-the-art models on the Cityscapes dataset. As with the FGSM and Iterative FGSM ll attacks in the main paper, PSPNet is typically the most robust. Once again, DilatedNet without its "Context" module is slightly more robust than the full, more accurate network. The single-step FGSM ll attack is quite effective at high ϵ values, but as expected, the Iterative FGSM attack is still more effective overall.

Table 6.7: The absolute IoU on the *Pascal VOC* dataset for various models when attacked with *FGSM*. This is evaluated for eight different values of the ℓ_∞ norm of the perturbation, ϵ. $\epsilon = 0$ represents the IoU on clean inputs.

Network	ℓ_∞ norm of perturbation, ϵ								
	0	0.25	0.5	1	2	4	8	16	32
SegNet (VGG)	43.0	32.3	25.9	19.5	14.8	11.7	9.7	6.9	4.0
Deeplab v2 ASPP (VGG)	66.9	55.3	44.1	31.7	22.5	17.2	13.9	11.8	9.1
Dilated Frontend (VGG)	67.1	56.7	45.7	33.8	24.2	19.2	16.1	12.2	8.2
FCN8s (VGG)	68.7	55.7	45.4	36.1	28.8	23.9	19.9	16.1	10.3
FCN8s (ResNet)	68.8	55.9	49.9	44.2	39.5	35.9	32.0	24.8	12.8
Dilated Context (VGG)	70.4	55.8	44.9	34.4	26.0	20.6	17.2	13.9	9.0
Deeplab v2 ASPP (ResNet)	73.3	61.6	52.7	43.3	35.9	30.7	27.7	24.6	18.5
Deeplab v2 ASPP MS (ResNet)	73.9	**66.9**	**60.9**	**54.1**	**47.9**	**43.2**	**39.2**	**35.7**	**28.5**
PSPNet (ResNet)	**75.9**	66.8	59.0	48.9	39.8	33.8	29.2	26.7	21.2

Table 6.8: The absolute IoU on the *Pascal VOC* dataset for various models when attacked with *FGSM ll*. This is evaluated for eight different values of the ℓ_∞ norm of the perturbation, ϵ. $\epsilon = 0$ represents the IoU on clean inputs.

Network	ℓ_∞ norm of perturbation, ϵ								
	0	0.25	0.5	1	2	4	8	16	32
SegNet (VGG)	43.0	36.2	27.4	17.6	11.4	8.3	6.7	5.4	4.1
Deeplab v2 ASPP (VGG)	66.9	61.5	52.3	34.6	17.3	9.5	7.0	6.1	5.6
Dilated Frontend (VGG)	67.1	61.6	51.9	35.8	19.1	10.6	6.6	5.5	4.4
FCN8s (VGG)	68.7	61.5	52.5	38.6	24.4	15.5	11.4	8.8	6.2
FCN8s (ResNet)	68.8	58.7	52.9	47.7	43.6	**41.0**	**36.8**	28.6	13.6
Dilated Context (VGG)	70.4	61.7	50.5	32.5	16.5	9.4	6.6	5.6	4.3
Deeplab v2 ASPP (ResNet)	73.3	67.8	60.4	49.1	37.5	30.0	25.7	22.0	17.2
Deeplab v2 ASPP MS (ResNet)	73.9	**71.5**	**67.4**	**59.5**	**48.4**	38.0	31.1	25.8	20.4
PSPNet (ResNet)	**75.9**	69.8	62.1	51.8	41.8	36.2	32.1	**29.8**	**26.6**

Table 6.9: The absolute IoU on the *Pascal VOC* dataset for various models when attacked with *Iterative FGSM*. This is evaluated for eight different values of the ℓ_∞ norm of the perturbation, ϵ. $\epsilon = 0$ represents the IoU on clean inputs.

Network	ℓ_∞ norm of perturbation, ϵ								
	0	0.25	0.5	1	2	4	8	16	32
SegNet (VGG)	43.0	34.3	29.9	20.8	12.1	5.9	3.1	2.2	1.9
Deeplab v2 ASPP (VGG)	66.9	55.5	45.8	31.3	17.6	8.2	4.1	3.0	2.3
Dilated Frontend (VGG)	67.2	57.5	48.8	32.6	16.3	7.9	4.3	3.2	2.2
FCN8s (VGG)	68.7	57.7	49.7	34.7	20.5	10.1	5.1	2.9	2.3
FCN8s (ResNet)	68.8	57.2	54.6	42.5	24.8	13.9	6.3	3.9	2.6
Dilated Context (VGG)	70.4	58.2	49.4	33.4	20.9	13.3	7.1	5.5	4.0
Deeplab v2 ASPP (ResNet)	73.3	64.3	55.1	38.0	22.4	11.9	6.4	4.2	2.9
Deeplab v2 ASPP MS (ResNet)	73.9	67.6	**63.9**	**47.6**	**35.0**	**21.6**	**14.5**	**10.6**	**9.3**
PSPNet (ResNet)	**75.9**	**69.0**	61.5	40.7	20.1	7.9	3.1	1.6	1.0

Table 6.10: The absolute IoU on the *Pascal VOC* dataset for various models when attacked with *Iterative FGSM ll*. This is evaluated for eight different values of the ℓ_∞ norm of the perturbation, ϵ. $\epsilon = 0$ represents the IoU on clean inputs.

Network	ℓ_∞ norm of perturbation, ϵ								
	0	0.25	0.5	1	2	4	8	16	32
SegNet (VGG)	43.0	35.9	28.4	16.3	9.3	5.8	3.7	3.0	2.9
Deeplab v2 ASPP (VGG)	66.9	61.9	53.2	32.3	17.3	8.9	5.1	4.4	3.2
Dilated Frontend (VGG)	67.2	62.3	52.7	33.9	16.1	8.0	4.8	3.9	3.5
FCN8s (VGG)	68.7	61.3	51.7	32.6	17.5	11.7	7.9	**5.4**	4.4
FCN8s (ResNet)	68.8	58.3	53.8	41.3	29.3	16.7	**9.1**	5.2	3.2
Dilated Context (VGG)	70.4	57.8	48.9	35.1	19.6	10.6	5.6	3.7	3.4
Deeplab v2 ASPP (ResNet)	73.3	67.8	59.2	40.4	21.4	11.4	6.2	5.4	**4.5**
Deeplab v2 ASPP MS (ResNet)	73.9	**72.3**	**70.0**	**58.2**	**38.0**	**19.0**	8.0	5.0	4.2
PSPNet (ResNet)	**75.9**	69.4	60.7	40.7	21.8	12.0	5.0	2.1	0.7

Table 6.11: The absolute IoU on the *Cityscapes* dataset for various models when attacked with *FGSM*. This is evaluated for eight different values of the ℓ_∞ norm of the perturbation, ϵ. $\epsilon = 0$ represents the IoU on clean inputs.

Network	ℓ_∞ norm of perturbation, ϵ								
	0	0.25	0.5	1	2	4	8	16	32
ENet	53.4	39.6	35.6	31.0	24.0	13.2	5.8	4.1	1.4
ICNet	56.5	47.0	41.3	35.5	28.5	16.8	4.5	2.4	0.8
FCN8 (VGG)	62.1	46.0	38.0	31.9	27.8	23.9	**16.2**	**7.7**	**3.9**
Dilated Frontend (VGG)	59.0	46.3	38.1	31.1	25.7	20.7	13.3	5.0	1.7
Dilated Context (VGG)	62.3	48.4	39.0	31.6	26.0	20.8	13.3	4.8	1.8
PSPNet (ResNet)	**74.4**	**58.5**	**52.9**	**48.9**	**46.0**	**36.3**	16.0	2.8	1.9

Table 6.12: The absolute IoU on the *Cityscapes* dataset for various models when attacked with *FGSM ll*. This is evaluated for eight different values of the ℓ_∞ norm of the perturbation, ϵ. $\epsilon = 0$ represents the IoU on clean inputs.

Network	ℓ_∞ norm of perturbation, ϵ								
	0	0.25	0.5	1	2	4	8	16	32
ENet	53.4	38.5	31.7	24.2	17.0	8.9	4.3	3.8	1.4
ICNet	56.5	47.2	40.5	33.2	25.1	13.4	3.4	2.3	0.8
FCN8 (VGG)	62.1	53.8	46.0	38.4	32.5	26.3	14.9	**6.4**	**3.8**
Dilated Frontend (VGG)	59.0	50.9	42.0	32.8	24.6	16.8	8.7	3.1	1.7
Dilated Context (VGG)	62.3	53.2	42.5	31.8	22.8	15.1	8.2	3.0	1.7
PSPNet (ResNet)	**74.4**	**64.9**	**59.1**	**55.0**	**51.3**	**39.5**	**16.5**	2.8	1.9

Table 6.13: The absolute IoU on the *Cityscapes* dataset for various models when attacked with *Iterative FGSM*. This is evaluated for eight different values of the ℓ_∞ norm of the perturbation, ϵ. $\epsilon = 0$ represents the IoU on clean inputs.

Network	ℓ_∞ norm of perturbation, ϵ								
	0	0.25	0.5	1	2	4	8	16	32
ENet	53.4	38.2	33.1	22.1	12.2	5.7	3.1	2.3	2.1
ICNet	56.5	45.8	38.8	28.6	16.6	**9.7**	**7.3**	2.5	0.9
FCN8 (VGG)	62.1	47.2	38.9	26.2	13.4	6.4	3.7	2.6	2.2
Dilated Frontend (VGG)	59.0	46.8	38.1	24.9	12.5	5.8	3.2	2.4	2.1
Dilated Context (VGG)	62.3	49.6	40.6	26.2	12.7	5.8	3.1	2.2	2.0
PSPNet (ResNet)	**74.4**	**58.2**	**50.7**	**35.2**	**20.1**	8.9	4.8	**3.3**	**2.7**

Table 6.14: The absolute IoU on the *Cityscapes* dataset for various models when attacked with *Iterative FGSM ll*. This is evaluated for eight different values of the ℓ_∞ norm of the perturbation, ϵ. $\epsilon = 0$ represents the IoU on clean inputs.

Network	ℓ_∞ norm of perturbation, ϵ								
	0	0.25	0.5	1	2	4	8	16	32
ENet	53.4	36.6	30.1	21.6	11.4	5.5	3.1	2.1	1.8
ICNet	56.5	45.8	38.8	28.6	16.6	9.7	7.3	2.5	0.9
FCN8 (VGG)	62.1	52.4	43.5	28.5	14.6	8.6	5.4	4.1	3.6
Dilated Frontend (VGG)	59.0	50.2	40.4	26.2	11.3	7.6	4.5	3.7	3.1
Dilated Context (VGG)	62.3	52.3	41.4	25.4	11.4	6.1	3.3	2.2	2.0
PSPNet (ResNet)	**74.4**	**63.8**	**57.1**	**41.8**	**28.6**	**16.8**	**9.3**	**5.3**	**4.4**

6.D Multiscale Processing and Transferability of Adversarial Examples

This section details additional results with both Deeplab v2 and FCN8s.

6.D.1 Deeplab v2

Table 6.15 shows the performance, measured in IoU, on the VOC validation set when the input image is processed at different resolutions (50%, 75%, 100%). The fact that a different IoU is obtained for each input resolution, even though the weights of the network are the same, confirms that the network is not scale invariant. Note that the version of Deeplab which processes images at all the aforementioned resolutions, and max-pools the prediction at each pixel obtains the highest IoU. An alternative to max-pooling the predictions from each scale is to average-pool them. This method gives an insignificant improvement in accuracy, but does improve robustness as shown in Fig. 6.16.

Table 6.15: Performance of Deeplab v2 (ResNet) on the VOC validation set when processing images at different resolutions.

Model Name	IoU [%]
Deeplab v2 50% scale	67.8
Deeplab v2 75% scale	71.9
Deeplab v2 100% scale	73.3
Deeplab v2 100% scale (average pooling)	73.4
Deeplab v2 Multiscale (max pooling)	73.9

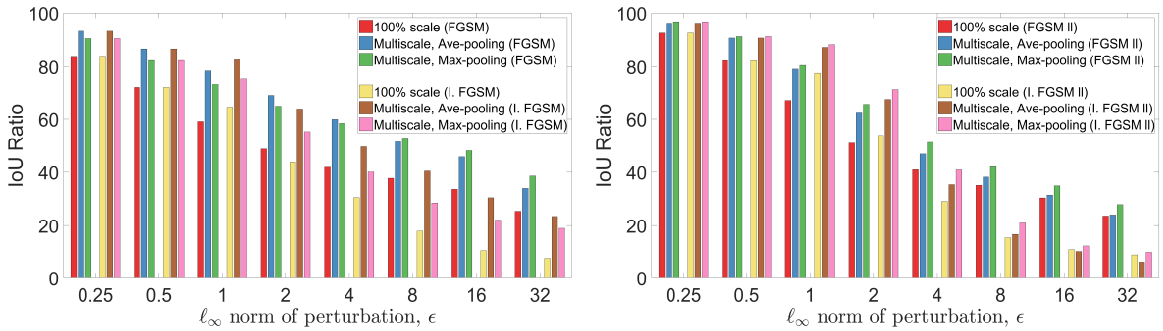
6.D.1.1 Average-pooling instead of max-pooling

As shown in Fig. 6.16, average-pooling the results from each scale is also more robust to all the adversarial attacks we tested compared to the single-scale version of Deeplab v2. In fact, multiscale processing (either max- or average-pooling) achieves a higher IoU Ratio at almost all ϵ values for each attack.

Table 6.17 also shows that black-box attacks generated from multiscale-averaging also transfer better to single scales of Deeplab v2, for all four adversarial attacks considered in this paper. This is similar to the case of max-pooling as shown in the main paper.

6.D.1.2 Transferability experiments using the FGSM ll and Iterative FGSM attacks

Table 6.18 shows the transferability of adversarial attacks to different scales of Deeplab v2 using the FGSM ll and Iterative FGSM attacks. The main paper presented results using the



(a) Untargeted attacks (b) Targeted attacks

Figure 6.16: Adversarial robustness of Deeplab ASPP (single-scale) and Deeplab Multiscale ASPP. We compare two types of multiscale ensembling – max-pooling and average-pooling the predictions from each of the three scales of Deeplab v2 (ResNet 101). Note that both average- and max-pooling are more robust than just a single-scale model, achieving higher IoU Ratios for almost every ϵ value for each attack on the Pascal VOC dataset.

Table 6.16: Performance of FCN8s when processing images at different resolutions. As with Deeplab v2, max-pooling the predictions from multiple scales achieves the best results.

Model Name	IoU [%]
FCN8s 50% scale	60.8
FCN8s 75% scale	67.8
FCN8s 100% scale	68.7
FCN8s Multiscale	69.9

FGSM and Iterative FGSM ll attacks. However, our findings remain consistent on these different attacks. The multiscale version of Deeplab v2 is the most robust to these attacks (as also seen in Fig. 6.14 and 6.16), and black-box attacks from it transfer the best to other scales of Deeplab v2.

6.D.1.3 Transferability experiments at multiple ϵ values

Figure 6.17 shows the results of black-box attacks for multiple ϵ values between different scales of Deeplab v2 for the FGSM attack. The results are largely consistent with those at $\epsilon = 8$ as reported in the main paper – the multiscale version of Deeplab v2 is the most robust to white-box attacks and black-box attacks generated from it transfer the best to other scales of Deeplab v2. Also note how the transferability from each scale to another varies greatly. For example, attacks generated from the 50% scale transfer very poorly to 100% and vice versa.

Table 6.17: Transferability of adversarial examples generated from different scales of Deeplab v2 (columns) and evaluated on different networks (rows). In this case, the outputs from each scale are *average-pooled* instead of max-pooled. The underlined diagonals for each attack show white-box attacks. Off-diagonals, show transfer (black-box) attacks. The most effective one in bold, is typically from the multiscale version of Deeplab v2. In the case of Iterative FGSM ll, black-box attacks from the multiscale networks are sometimes even more effective than white-box ones.

Network evaluated	FGSM ($\epsilon = 8$)				Iterative FGSM ll ($\epsilon = 8$)			
	50%	75%	100%	Multiscale	50%	75%	100%	Multiscale
Deeplab v2 0.5 (ResNet)	<u>37.3</u>	70.5	84.8	**48.8**	<u>18.0</u>	92.0	96.9	**12.1**
Deeplab v2 0.75 (ResNet)	85.5	<u>39.7</u>	62.2	**54.2**	99.5	<u>17.9</u>	89.9	**17.4**
Deeplab v2 1 (ResNet)	93.6	57.9	<u>37.7</u>	**51.7**	100.0	79.0	<u>15.5</u>	**9.6**
Deeplab v2 Multiscale (ResNet)	75.1	**54.2**	59.0	<u>51.6</u>	95.2	**84.9**	87.5	<u>16.7</u>
Network evaluated	FGSM ll ($\epsilon = 8$)				Iterative FGSM ($\epsilon = 8$)			
	50%	75%	100%	Multiscale	50%	75%	100%	Multiscale
Deeplab v2 50% (ResNet)	<u>36.4</u>	70.1	83.7	**36.6**	<u>21.3</u>	90.9	97.0	**37.3**
Deeplab v2 75% (ResNet)	89.9	<u>37.4</u>	61.6	**39.9**	99.1	<u>20.0</u>	88.6	**44.1**
Deeplab v2 100% (ResNet)	95.1	58.3	<u>35.1</u>	**36.9**	100.2	71.9	<u>18.6</u>	**33.5**
Deeplab v2 Multiscale (ResNet)	96.0	**91.4**	94.7	<u>38.2</u>	94.5	**76.2**	86.5	<u>37.7</u>

6.D.2 FCN8s

Table 6.16 shows the IoU of FCN8s (VGG) as the input resolution of the image is varied from the VOC dataset. As with Deeplab v2, a multiscale version which max-pools the predictions from each scale achieves the highest IoU.

The transferability experiments from Section 6 of the paper are repeated on FCN8 in Tables 6.19 and 6.20. Note that FCN8s has not been trained in a multiscale manner as Deeplab v2, and it is rather done as a post-processing step. Nevertheless, the results show a similar trend as Deeplab v2: The multiscale network is more robust to white-box attacks and black-box attacks generated from it transfer better. This suggests that training the network in a multiscale manner does not confer robustness to adversarial examples. Rather it is the fact that CNNs are not scale invariant, and that adversarial examples generated at one scale are not as malignant at another. Finally Fig. 6.18 shows the transferability experiments at multiple ϵ values, as was done for Deeplab v2 in the previous subsection.

Table 6.18: Transferability of adversarial examples generated from different scales of Deeplab v2 (columns) and evaluated on different networks (rows). As with the main paper, *max-pooling* is performed from the output of each scale. However, in contrast to the main paper, the FGSM ll and Iterative FGSM attacks are reported. The underlined diagonals for each attack show white-box attacks. Off-diagonals, show transfer (black-box) attacks. The most effective one in bold, is typically from the multiscale version of Deeplab v2.

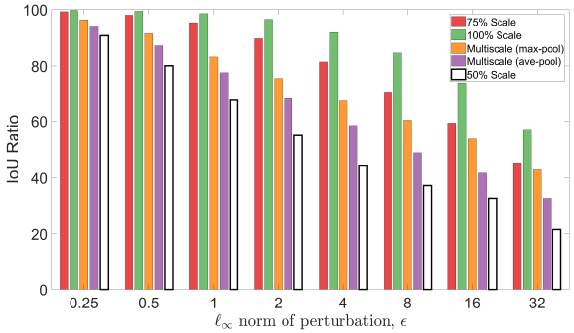
Network evaluated	FGSM ll ($\epsilon = 8$)				Iterative FGSM ($\epsilon = 8$)			
	50%	75%	100%	Multiscale	50%	75%	100%	Multiscale
Deeplab v2 0.5 (ResNet)	36.4	70.1	83.7	**46.0**	21.3	90.9	97.0	**39.2**
Deeplab v2 0.75 (ResNet)	89.9	37.4	61.6	**43.3**	99.1	20.0	88.6	**34.0**
Deeplab v2 1 (ResNet)	95.1	58.3	35.1	**33.9**	100.2	71.9	18.6	**22.0**
Deeplab v2 Multiscale (ResNet)	90.7	**60.8**	68.9	42.1	96.5	**81.9**	87.5	29.2
Deeplab v2 (VGG)	95.1	69.9	63.8	**61.9**	98.5	86.9	86.3	**81.2**
FCN8 (VGG)	94.5	67.7	64.7	**62.4**	98.7	86.9	86.0	**82.0**

Table 6.19: Transferability of adversarial examples generated from different scales of FCN8s (VGG) (columns) and evaluated on different networks (rows) on the Pascal VOC dataset. For the multiscale network, the outputs from each scale are max-pooled. The underlined diagonals for each attack show white-box attacks. Off-diagonals, show transfer (black-box) attacks. The most effective one in bold, is typically from the multiscale version of FCN8s.

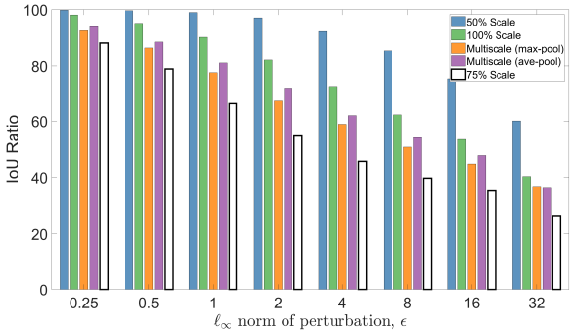
Network evaluated	FGSM ($\epsilon = 8$)				Iterative FGSM ll ($\epsilon = 8$)			
	50%	75%	100%	Multiscale	50%	75%	100%	Multiscale
FCN8 50%	32.1	**53.3**	81.0	53.7	20.5	87.3	96.9	**21.9**
FCN8 75%	78.4	30.9	45.5	**40.5**	96.3	17.6	77.8	**20.5**
FCN8 100%	94.0	41.7	28.9	**28.7**	98.2	58.6	15.3	**17.5**
FCN8 Multiscale	79.1	**42.8**	53.3	47.8	97.5	**79.3**	85.2	20.0

Table 6.20: Transferability of adversarial examples generated from different scales of FCN8s (VGG) (columns) and evaluated on different networks (rows) on the Pascal VOC dataset. For the multiscale network, the outputs from each scale are max-pooled. The underlined diagonals for each attack show white-box attacks. Off-diagonals, show transfer (black-box) attacks. The most effective one in bold, is typically from the multiscale version of FCN8s.

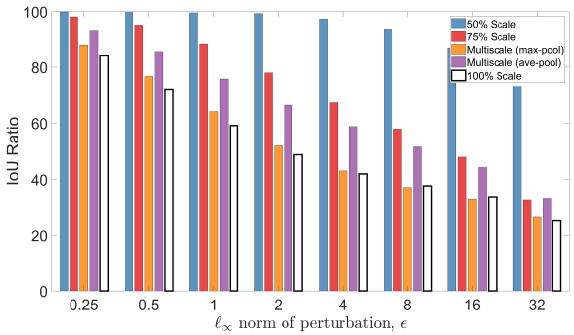
Network evaluated	FGSM ll ($\epsilon = 8$)				Iterative FGSM ($\epsilon = 8$)			
	50%	75%	100%	Multiscale	50%	75%	100%	Multiscale
FCN8 50%	18.5	51.4	79.2	**24.0**	23.6	85.7	97.1	**38.1**
FCN8 75%	80.9	18.5	37.0	**23.4**	97.3	15.9	74.7	**28.1**
FCN8 100%	93.0	33.8	16.6	**17.1**	99.1	54.9	14.7	**18.1**
FCN8 Multiscale	87.5	**40.0**	60.3	21.1	96.4	**74.5**	82.3	25.1



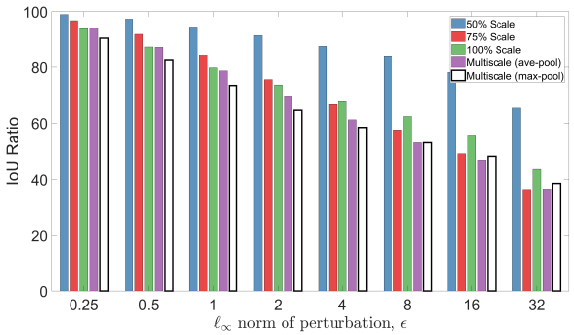
(a) Transfer attacks on Deeplab v2 50% scale



(b) Transfer attacks on Deeplab v2 75% scale

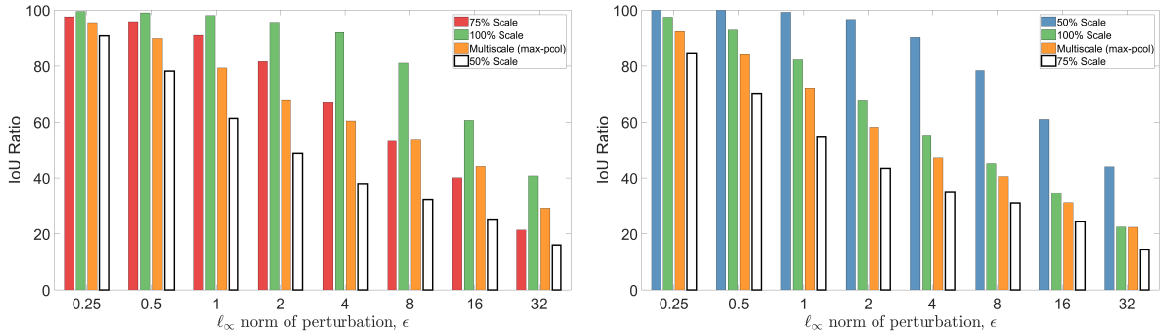


(c) Transfer attacks on Deeplab v2 100% scale



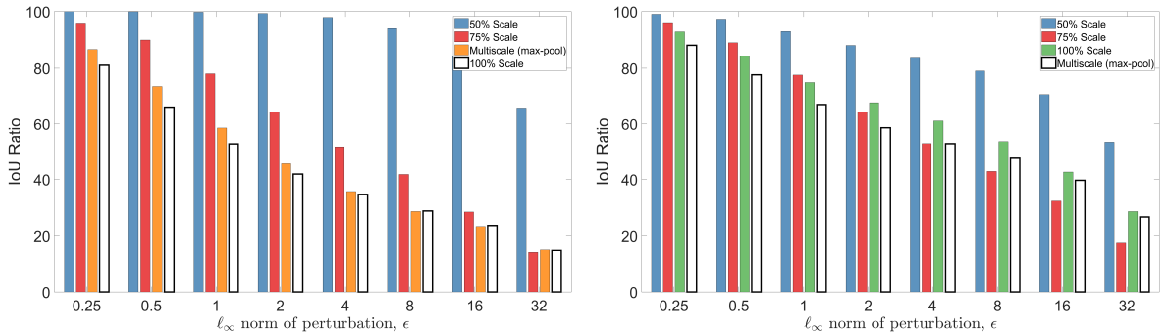
(d) Transfer attacks on Deeplab v2 Multiscale (max-pooling)

Figure 6.17: Black-box attacks on each scale of Deeplab v2, from each other scale, using adversarial perturbations generated by FGSM for differing values of ϵ on the Pascal VOC dataset. In each figure, the last bar shows the "white-box" attack on the network, where the attack is generated from the network that is being evaluated. This is typically the most powerful attack, as expected. Note that attacks generated from the multiscale version of Deeplab v2 (using either max- or average-pooling) produce the most effective black-box attacks across multiple ϵ values. The trend from the main paper, which only tabulated the IoU Ratio for $\epsilon = 8$, can thus be seen across all other ϵ values considered in this paper.



(a) Transfer attacks on FCN8s 50% scale

(b) Transfer attacks on FCN8s 75% scale



(c) Transfer attacks on FCN8s 100% scale

(d) Transfer attacks on FCN8s Multiscale (max-pooling)

Figure 6.18: Black-box attacks on each scale of FCN8, from each other scale, using adversarial perturbations generated by FGSM for differing values of ϵ on the Pascal VOC dataset. In each figure, the last bar shows the "white-box" attack on the network, where the attack is generated from the network that is being evaluated. The results from this experiment are very similar to Deeplab v2 – attacks generated from the multiscale network transfer the best to other scales. However, unlike Deeplab v2, the FCN8s network in this case was not trained with multiscale ensembling. This was simply done at test-time. This suggests that the increased robustness of multiscale networks to adversarial attacks, and their transferability to other networks, is not a result of the training procedure, but rather the fact that these networks are not scale invariant.

6.E Effect of CRFs on Adversarial Robustness

6.E.1 Adversarial Robustness and Smoothing

The pairwise term of DenseCRF [154] (which is interpreted as a neural network in CRF-RNN [329]) takes the form of a weighted sum of a Bilateral and Gaussian filter,

$$\psi_p(x_i, x_j) = \mu(x_i, x_j)\left[w_1 \exp\left(\frac{|p_i - p_j|^2}{\theta_\alpha} + \frac{|I_i - I_j|^2}{\theta_\beta}\right) + w_2 \exp\left(\frac{|p_i - p_j|^2}{\theta_\gamma}\right)\right]. \quad (6.8)$$

Increasing θ_α, θ_β, θ_γ, w_1 and w_2 all correspond to favouring smoother predictions. The compatibility function, $\mu(x_i, x_j)$, is given by the Potts model, and is equal to 1 if $x_i \neq x_j$ and 0 otherwise [154].

Figure 6.19 shows the effect of varying θ_α, Fig. 6.20 the effect of varying θ_β and Fig. 6.21 the effect of varying both θ_γ and w_2. Note that in all cases, each of the other hyperparameters remains unchanged at the values from the public CRF-RNN model.

In all of these cases, we can see that increasing the smoothness does not correspond to increasing adversarial robustness to the FGSM attack. Rather, as detailed in the next subsection, there is a correlation between the confidence of the prediction and robustness to the FGSM attack.

6.E.2 Results about the confidence on VOC

We empirically measured the confidence of the predictions of CRF-RNN. This was done by recording the probability (from the softmax activation function) of the predicted (highest-scoring) label, and also by calculating the entropy of the marginal distribution over labels at each pixel in the image. A lower entropy indicates a more certain or confident prediction. This was then averaged over the Pascal VOC validation set.

Figures 6.22 and 6.23 show the mean confidence and entropy respectively as a function of the IoU Ratio. This is done for the FGSM attack for all the ϵ values considered in the paper. There is a clear correlation between the IoU Ratio and the confidence of the prediction. Moreover, the results of CRF-RNN are always more confident than FCN8s. Note that multiple variants of CRF-RNN, using different θ_α, θ_β and θ_γ hyperparameter values were considered, as in Figures 6.19 through 6.21.

6.E.3 Experiments on Deeplab v2 with CRF

In contrast to CRF-RNN [329], a common approach is to apply CRFs as a post-processing step, as done in Deeplab [43]. We perform adversarial attacks on this by appending the CRF-RNN layer of [329] onto the Deeplab v2 network. This allows us to compute the gradient of the loss with respect to the input image (required for all the attacks) by backpropagating

through the CRF-RNN layer. The parameters of the CRF-RNN layer appended to Deeplab v2 were manually set to the parameters used by the original authors[10] (who obtained them via cross-validation). Note that appending the CRF-RNN layer to Deeplab v2 and using the same parameters as the authors produces output that is identical to the post-processing code used by the original authors. The difference is that this allows us to compute gradients as well.

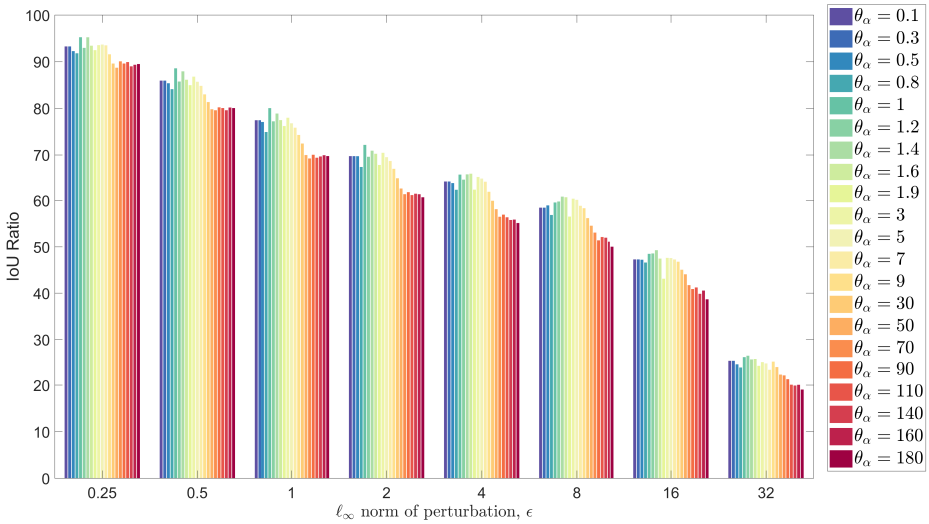


Figure 6.19: The IoU Ratio of CRF-RNN for various values of the θ_α (filter bandwidth) hyperparameter when attacked with FGSM on the Pascal VOC dataset. Increasing this hyperparameter visually smoothes the result further, but we can see that this does not increase adversarial robustness. In fact, lower filter bandwidths of approximately $\theta_\alpha = 1$ provide more robustness.

[10] http://liangchiehchen.com/projects/DeepLabv2_resnet.html

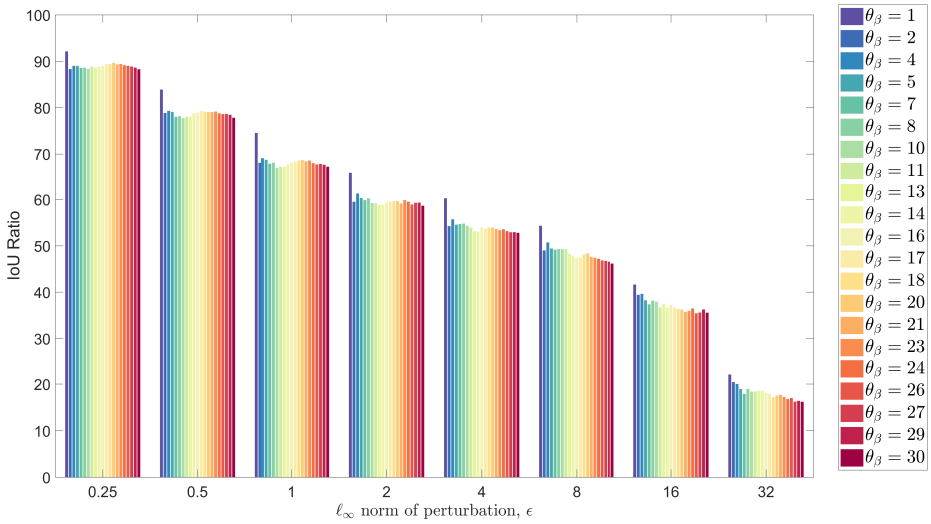


Figure 6.20: The IoU Ratio of CRF-RNN for various values of the θ_β (filter bandwidth) hyperparameter when attacked with FGSM on the Pascal VOC dataset. Again, we can see that larger filter bandwidths, which encourage more spatial smoothness, do not increase adversarial robustness.

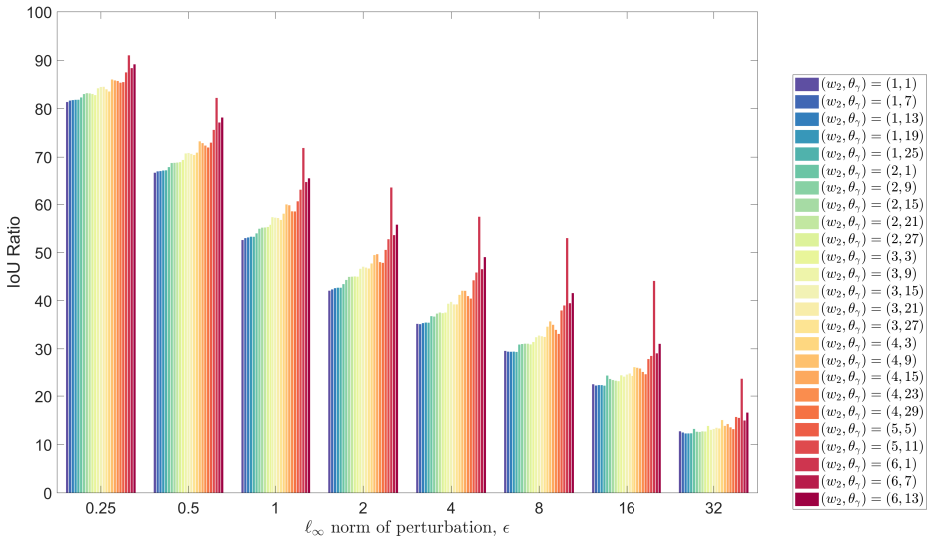


Figure 6.21: The IoU Ratio of CRF-RNN for various values of the w_2 and θ_γ parameters when attacked with FGSM on the Pascal VOC dataset. Increasing the weight of the Gaussian term (w_2) tends to increase robustness. However, we still see that lower filter bandwidths (θ_γ) tend to provide more robustness.

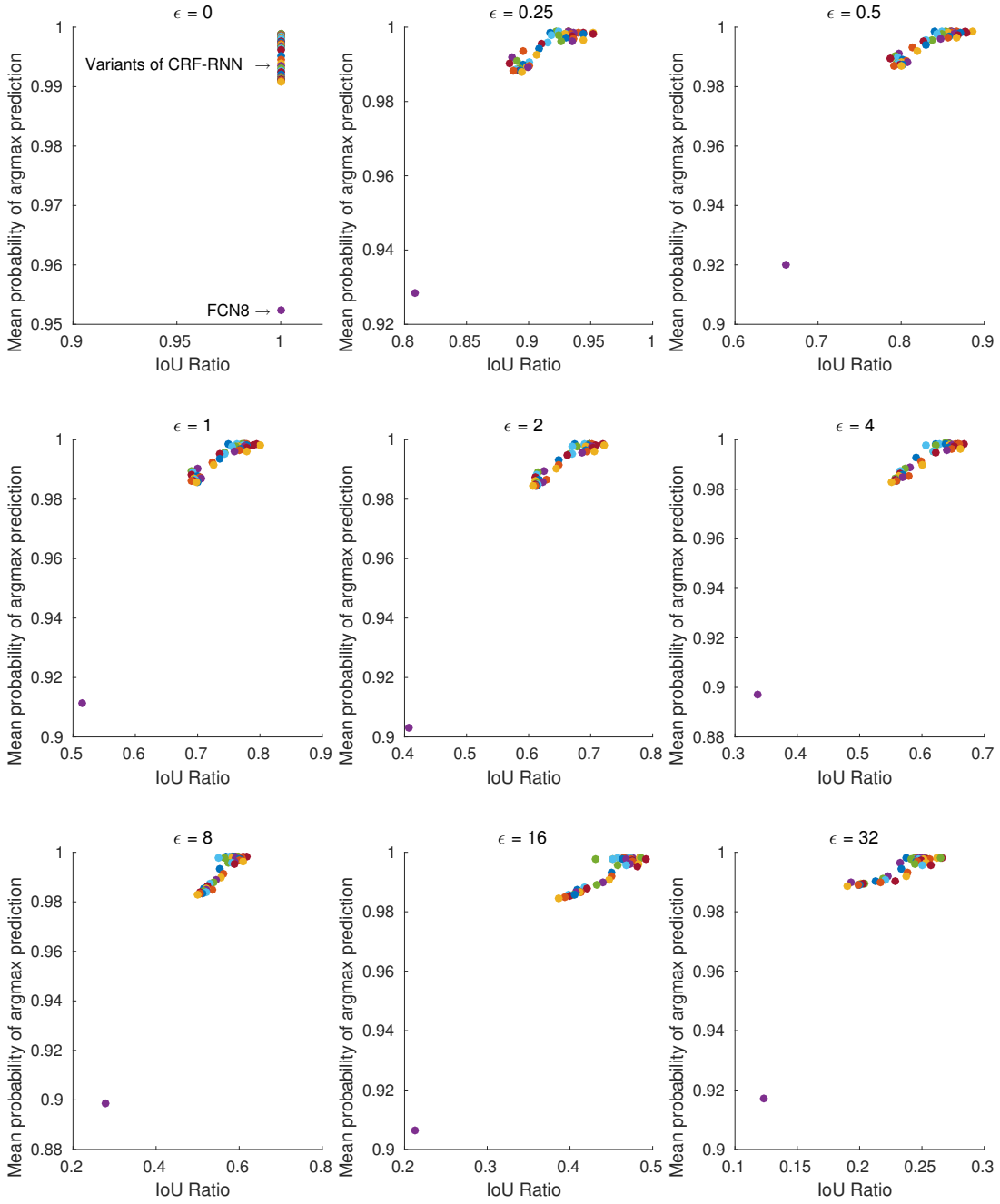


Figure 6.22: The mean probability of the highest-scoring class for each pixel, averaged over the Pascal VOC validation set. This is performed for the FGSM attack for multiple ϵ values. $\epsilon = 0$ corresponds to clean inputs (no adversarial attack). Note how FCN8s (the purple dot) consistently has the lowest mean probability. This probability is significantly lower than other variants of CRF-RNN (with varying θ_α, θ_β, θ_γ), shown by the other coloured dots. Moreover, note the correlation between the confidence in the prediction, and adversarial robustness to the FGSM attack. Additionally, the probability of the predicted class remains high (above 90%) for all models throughout all adversarial attacks.

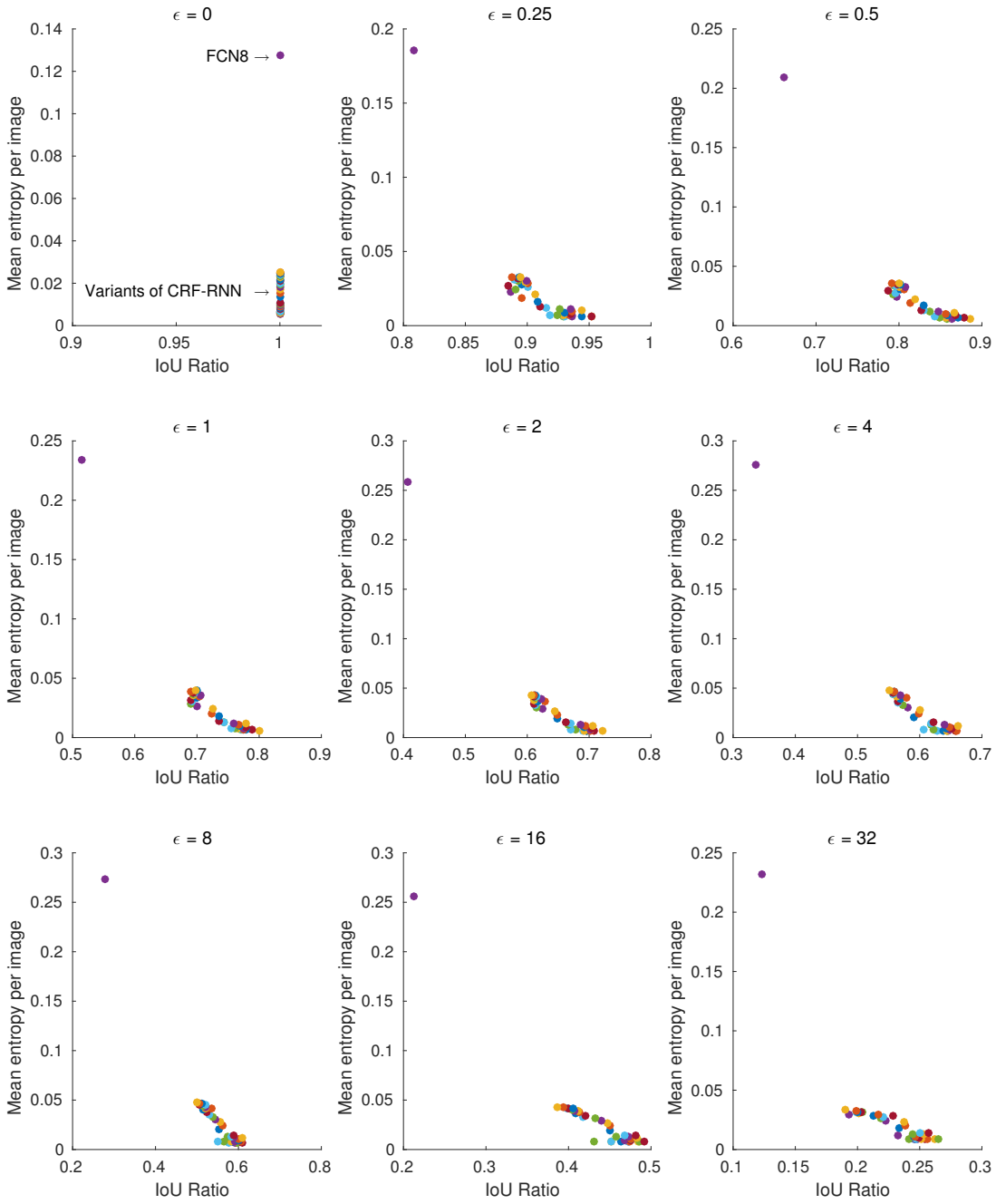


Figure 6.23: The mean entropy of the marginal distribution over all labels at each pixel, averaged over all images in the Pascal VOC validation set. A lower entropy corresponds to a more confident prediction. This is performed for the FGSM attack for multiple ϵ values. $\epsilon = 0$ corresponds to clean inputs (no adversarial attack). Note how FCN8s (the purple dot) consistently has the highest mean entropy (least confidence). This entropy is significantly higher than other variants of CRF-RNN (with varying θ_α, θ_β, θ_γ), shown by the other coloured dots. Moreover, note the correlation between the confidence in the prediction, and adversarial robustness to the FGSM attack.

Chapter 7

Conclusions

Chapter 3 of this integrated thesis first presented a method for the task of semantic segmentation by integrating mean-field inference of a Conditional Random Field (CRF) with higher order potentials directly into a deep neural network. This end-to-end trained network achieved state-of-the-art results at the time of publication. Chapter 4 then extended this network to perform the task of instance segmentation. In contrast to previous approaches, this method jointly predicted all object instances in the image and thus does not predict overlapping instances. Moreover it can naturally deal with "stuff" classes as well. The fact that pixel-accurate training data for segmentation models is time-consuming and expensive to obtain was addressed in Chapter 5 which presented a method of training the instance segmentation model from the previous chapter with weaker annotations in the form of bounding boxes and image-level tags. Finally, motivated by the fact that segmentation systems are now becoming accurate enough to use in real-world applications, Chapter 6 studied the adversarial robustness of different segmentation architectures. The findings from this chapter improve our understanding of adversarial examples and will help future efforts to train models that are both accurate and robust to adversarial attacks.

Section 7.1 now summarises contributions of each of the four papers presented in this thesis, and discusses subsequent advances in the field since publication, potential extensions of the work and the impact the work had on the field. Thereafter, Sec. 7.2 discusses future directions and open questions that were not considered in this thesis. Finally, concluding remarks are presented in Sec. 7.3.

7.1 Discussion of contributions

Chapter 3: Higher Order Conditional Random Fields in Deep Neural Networks

Chapter 3 proposed a CRF with higher order potentials for the task of semantic segmentation. Mean-field inference of this CRF was formulated as a recurrent neural network layer, enabling

joint, end-to-end training of both the parameters of the CRF and the underlying CNN. At the time of writing, it was the leading method on the public Pascal VOC benchmark, and is currently still the second-best for methods using the VGG [267] backbone architecture. Furthermore, this algorithm also obtained the highest performance on the Pascal Context dataset.

The methods that initially outperformed the Higher Order CRF-CNN architecture used a combination of the more powerful ResNet [117] backbone and CRFs [43, 40]. On a separate track, Luc *et al.* [194] also proposed an alternate method of encouraging higher order structural constraints using a generative adversarial network framework (the "generator" was the segmentation network, and the "discriminator" judged whether the predicted segmentation was "real" or "fake").

However, more recently, state-of-the-art methods in semantic segmentation have not been explicitly encouraging structure in the network's output, but have rather designed architectures for pixel-level prediction tasks [328, 48, 320]. Furthermore, neural networks with attention modules [320, 92, 300], originating from natural language processing [293, 14], have been popular as well. In contrast to convolution operations which propagate information in a fixed grid in each layer, attention-based methods are able to propagate information throughout the image and thus model long-range interactions between pixels. These messages, however, are learned automatically by a neural network from data and conditioned on each input image. Densely connected pairwise potentials of a CRF (Eq. 2.11) are also able to model long-range interactions, but the pairwise potential is based only on positional and image-intensity features, and not learned features.

It is also worth noting that current state-of-the-art segmentation architectures are dependent on using large batch sizes during training, as it enables better estimates of batch statistics for batch normalisation [328, 45, 129]. For example, PSPNet [328] was trained with 16 GPUs, with the batch statistics after each layer being synchronously aggregated across the different GPUs. When training PSPNet with only a single GPU, performance is about 5% lower. Therefore, understanding the effect of batch normalisation on training and developing methods for training accurate models with lower hardware resources is an important future avenue for research.

Although CRFs are currently not being used commonly in fully-supervised semantic segmentation, the more general idea of interpreting inference algorithms as differentiable operations which can be incorporated into neural networks has proven popular in multiple domains. Examples include unrolled and differentiable versions of a primal-dual solver for depth super-resolution [245], bundle adjustment [279], non-linear least squares [56],

the "Deep matching" algorithm (which is not a neural network contrary to what the name suggests) for optical flow estimation [280] and correlation filters for object tracking [291].

Chapter 4: Pixelwise Instance Segmentation with a Dynamically Instantiated Network

Chapter 4 extended the semantic segmentation network from the previous chapter to perform the task of instance segmentation. The formulation, which associated each pixel with an object detection, had the advantage that it considered all object instances in the image jointly, and did not allow one pixel to belong to multiple instances as in other related algorithms [116, 65, 183]. Furthermore, the algorithm presented in Chapter 4 has no problem in segmenting "stuff" [91] classes in contrast to detection-based approaches [116, 65, 183, 142].

At the time of publication, this method was the leading method on the public Cityscapes benchmark [57], and also the Pascal VOC [81] and SBD datasets [110]. The subsequent methods which have surpassed it on the Cityscapes benchmark [183, 116] have mostly been detection-based approaches which output more instances than are actually present in the image. Furthermore, the predicted instances are allowed to overlap each other. Public benchmarks which use the AP^r ranking metric are biased to such approaches and do not require one pixel to be assigned to only one instance.

The fact that the instance segmentation literature broadly consists of two approaches – firstly, methods which assign an instance identifier to each pixel in the image [9, 8, 15, 182, 143], and secondly, detection-based methods which produce a ranked list of instance masks which can overlap each other [116, 65, 183, 111, 112] – has since also been noted by Kirilov *et al.* [142]. To differentiate these approaches, Kirilov *et al.* have introduced the task of "Panoptic Segmentation" [142] which does not allow overlaps, and requires one to segment both "thing" and "'stuff" classes.

The task of "Panoptic Segmentation" is indeed what was being accomplished in Chapter 4. However, at the time of writing, this task was not considered as an independent scene understanding problem. Furthermore, the "Panoptic Quality" (PQ) metric proposed by [142] to evaluate the task of "Panoptic Segmentation" is the product of two terms – the Segmentation Quality (SQ) and Detection Quality (DQ). Note that the "Segmentation Quality" is actually the same as the "Matching IoU" used as an additional evaluation metric in Chapter 4 and [317] before that.

One of the shortcomings of the algorithm presented in Chapter 4 was that it used a separately trained object detector to provide inputs to the system. This problem has been addressed in many recent works which train a multi-task network with a common-set of

layers, and separate "heads" for object detection and semantic segmentation [311, 141, 171, 236]. The current state-of-the-art approach for panoptic segmentation, UPSNet [311] follows this multi-task training idea. Note that its "Panoptic head" is similar to the "Box term" described in Chapter 4 and its Mask-RCNN head performs a similar function to the "Shape term" used in Chapter 4.

Chapter 5: Weakly and Semi-supervised Panoptic Segmentation

This work extended the segmentation model from the previous chapter by showing how it could be trained with weaker supervision using a method based on self-training and the Expectation Maximisation (EM) algorithm. Instead of using full pixel-wise annotations, which are expensive to collect, weaker annotations in the form of image-level tags and bounding boxes were used. On the Cityscapes dataset, this corresponded to a reduction in the annotation time (and hence cost) by a factor of 35. Even with this weaker supervision, the weakly-supervised model could obtain up to 95% of fully-supervised performance with the same data.

This paper remains, at the time of writing, the only approach to perform "panoptic segmentation" without pixel-level annotations in the training data. Although this paper uses the same model as Chapter 4, it refers to the task being performed as "panoptic segmentation" as it was published subsequent to [142].

One of the weaknesses of the segmentation model is that it needs additional object detections as its input. Consequently, the model cannot trivially be extended to perform instance- or panoptic-segmentation from only image-level tags (the number of instances of a particular class would also be required). It is however possible that the same self-training procedure, outlined in this chapter, can be used for this scenario of only training with image-level tags, although a different segmentation model would be needed.

Chapter 6: On the Robustness of Semantic Segmentation Models to Adversarial Attacks

Chapter 6 presented a rigorous evaluation of the robustness of semantic segmentation models to adversarial attacks. This chapter was motivated by the fact that during the course of this thesis, the performance of semantic segmentation systems have rapidly improved to the point that they are becoming suitable for use in real-world applications, where security of machine learning systems are critical.

The chapter presented numerous observations that will facilitate future efforts in understanding and defending against adversarial examples. In the shorter term, it showed how to effectively benchmark adversarial robustness and suggested which models should

currently be preferred in safety-critical applications due to their inherent robustness. Furthermore, mean-field inference was shown to produce overconfident predictions and naturally perform "gradient masking", conferring robustness only to common untargeted adversarial attacks. There were, however, no robustness benefits from the smoothness priors enforced by the DenseCRF model.

At the time of writing the original conference paper, adversarial examples were attracting attention in the computer vision community, and numerous defences to adversarial examples had been proposed [176, 237, 270, 107, 308]. However, these have all since been subverted by [11] and [286]. Chapter 6 also showed how concurrently proposed defences based on input transformations [107, 308] were ineffectual if knowledge of these input transformations was used to generate the attack. Input transformations could also be subverted if they were stochastic and/or non-differentiable. Adversarial training [196] is the only method which has increased the robustness of neural networks significantly in white-box settings. However, adversarial training is not effective against adversarial attack algorithms that were not used in training, and also comes at the cost of decreased accuracy on clean inputs.

Although several theoretical and empirical works [256, 96, 285, 272, 86, 84] have since attempted to improve our understanding of adversarial examples and neural networks, there is still currently no solution that both preserves predictive accuracy and is robust to adversarial examples. Understanding and countering adversarial examples thus remain an important and open research question.

7.2 Future directions and open questions

Incrementally learning new classes from few examples

The algorithms described in this thesis have considered datasets such as Cityscapes [57] and Pascal VOC [81] which have 19 and 20 labelled classes respectively. However, an ideal scene understanding system should be able to extend its capabilities after it has been trained, by being able to learn to classify new object classes without degrading its performance on the original classes it was trained on. Furthermore, it should be able to learn these new classes from few training examples, assuming that these new classes are visually related to the previous classes the system was trained on. Finally, training examples from the original dataset should not be required when learning new classes.

Simply fine-tuning a network on examples of the new classes is not sufficient, as it results in the performance on the original classes degrading severely. This outcome is known as "catastrophic forgetting" [198, 197]. Although the problem of learning tasks sequentially with neural networks without forgetting previous tasks has recently been studied, and

known as continual- or incremental-learning [144, 240, 173], the field is not as mature as other topics in computer vision.

Currently, methods for incremental learning still require some training examples from the original dataset to prevent performance degradation, are not as accurate as standard supervised learning on a single dataset and have not been demonstrated on more complex scene understanding tasks such as segmentation and detection.

As a result, many practical applications – such as a robot which must quickly learn environment-specific object models in addition to its default ones – are currently still not possible despite the high performance of recent methods on benchmark datasets.

Model introspection

The neural networks in this thesis have been evaluated in "closed-world" settings where each of the test examples has a ground truth label which the network was trained on. However, such an evaluation protocol is not suitable for the "open-world" setting encountered in practical applications where a scene understanding system may encounter an input that does not belong to any of the categories that it was trained on. In such a situation, the algorithms presented in this thesis will still incorrectly classify the input example into one of the predefined object classes that it was trained on, and often with high confidence.

In the same situation, a scene understanding system should be able to identify that it is likely to fail, and predict an "I don't know" label, rather than one of the predefined labels that it was trained with. Note that this situation can also arise when there is a significant "domain gap" between the input example and the training data of the model. For example, Zendel *et al.* [323] showed how semantic segmentation models trained on common road scene datasets [57, 214] fail catastrophically in hazardous conditions (which were not in the training set).

Training a classifier with an additional "I don't know" class is not sufficient [27], as training data for this additional class will be required (and by definition, the unknown examples in the test set cannot be in the training set).

Robustness in open-world or out-of-domain scenarios is closely related to quantifying the uncertainty in a network's prediction [195, 93, 137], and is crucial for safety-critical applications such as autonomous vehicles where fatalities can occur as a result of errors made by the perception system [289].

Robust computer vision models

Chapter 6 investigated the vulnerability of common segmentation models to adversarial attacks, and follows the adversarial example literature which considers perturbations with

a constrained ℓ_p norm (typically l_∞ or ℓ_2).

There are, however, many other transformations of an input example that neural networks should be robust to. For example, Engstrom *et al.* [77] have shown that carefully chosen rotations and translations are sufficient to fool neural network-based image classification models (including adversarially trained [196] ones). Similarly, Hendrycks *et al.* [122] have shown how a wide range of image corruptions also severely degrades the performance of image classifiers.

Developing robust computer vision models for practical applications thus requires considering more complex attack models than the ones currently used in the adversarial attack literature.

Overcoming the "Curse of Dataset Annotation"

The deep neural networks used in this thesis require large amounts of training data to learn powerful feature representations automatically. The dependence of neural networks on large datasets has been termed by Xie *et al.* [310] as the "Curse of dataset annotation" due to the cost incurred in creating labelled datasets, particularly for tasks such as semantic segmentation.

Chapter 5 addressed this problem by using weaker annotations – in the form of bounding boxes and image-level tags – for learning segmentation models. However, even these annotations require significant human effort and cost to obtain.

A promising alternative is to create synthetic datasets which allow one to produce effectively unlimited amounts of training data for free. This approach was demonstrated by Richter *et al.* [244] and Dosovitskiy *et al.* [75] who utilised computer games and a purpose-built simulator based on a computer game engine respectively.

However, models trained on only synthetic data to not generalise well to the real-world due to the differences in the distribution of training and test data [244]. As a result, effective domain adaptation methods [59] to adapt a model trained on simulated data to the distribution of real-world data that it will ultimately be tested on remains an important research direction. Furthermore, how to optimally generate synthetic data, given a parametric simulator, for the task of interest is an interesting and open research question.

3D Scene Understanding

The algorithms in this thesis have considered 2D scene understanding tasks where each pixel in the image is assigned a label. However, images are the result of projections of 3D objects onto cameras. Therefore, a richer understanding of a scene could be obtained by predicting 3D models of objects and the camera parameters that result in the observed 2D

image being formed. This idea, of directly modelling and inverting the complex image formation process, is also referred to as "vision- as inverse-graphics" [247, 158, 78, 307]. A major benefit of performing 3D scene understanding is that traditional scene representations, such as pixel-level segmentations, bounding boxes and depth maps can easily be produced as a by-product by rendering the 3D model. Furthermore, it is also possible to model parts of an object which are occluded in the 2D image.

"Vision as inverse-graphics" approaches have only recently been demonstrated on real-world data [159, 26], and have fitted purpose-designed 3D models [190, 249] or CAD models [41] of specific object classes (such as humans) to images. Scaling up these approaches to the datasets considered in this thesis would also require efficient methods of constructing 3D object models, or using high-dimensional volumetric representations of objects. Furthermore, as it is not possible to obtain metric ground truth 3D information outside of lab-controlled environments, such 3D scene understanding models have to be trained with weaker supervision. For example, 2D segmentation masks or keypoints can be used as weak supervision since the rendering of the 3D model onto 2D should match the segmentation mask.

Unified models for scene understanding

Scene understanding consists of many different problems (Sec. 1.1) which are typically solved independently of one another, even though these problems are highly related. For example, in Chapter 4 and 5, a separate object detector was used to solve the task of instance/panoptic segmentation.

An obvious goal is thus to perform multiple scene understanding tasks with a single, unified model. In addition to reducing the computational cost of performing multiple tasks individually, such a unified model should also be able to exploit the synergies between different tasks so that it performs better than a specialised, task-specific model.

This goal has been pursued by Kokkinos [150] by training a multi-task neural network with a common set of layers followed by individual "heads" for each task. However, performance on each task deteriorated as more tasks were added, meaning that separate, task-specific models were more accurate than a single, multi-task one.

7.3 Concluding remarks

The widespread adoption of deep neural networks, and availability of large datasets and computing power has facilitated great advances in computer vision in the last few years. For example, the state-of-the-art approach for semantic segmentation on the Pascal VOC

benchmark before the use of deep learning obtained an IoU of 47.8% [37]. The method presented in Chapter 3 achieved 77.9% which was the best at the time. The current leading approach now achieves 88.5% [48].

However, as discussed in the previous section (Sec. 7.2), there are still many open questions to be solved, which may require new problem formulations, new datasets and new evaluation metrics as more challenging problems are considered. Further advances in these areas will bring us closer to computers that understand our physical world and help enrich it.

Bibliography

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, et al. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012 (cited on pages 36, 40, 43).

[2] A. Adams, J. Baek, and M. A. Davis. Fast high-dimensional filtering using the permutohedral lattice. In *Computer Graphics Forum*. Wiley Online Library, 2010 (cited on pages 23, 27).

[3] E. H. Adelson. On seeing stuff: the perception of materials by humans and machines. In *Human vision and electronic imaging VI*. International Society for Optics and Photonics, 2001 (cited on page 86).

[4] M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele. 2d human pose estimation: new benchmark and state of the art analysis. In *Computer Vision and Pattern Recognition*, 2014 (cited on page 3).

[5] P. Arbelaez, J. Pont-Tuset, J. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping. In *Computer Vision and Pattern Recognition*, 2014 (cited on pages 17, 18, 57, 89, 90, 99).

[6] A. Arnab, S. Zheng, S. Jayasumana, B. Romera-Paredes, M. Larsson, et al. Conditional random fields meet deep neural networks for semantic segmentation: combining probabilistic graphical models with deep learning for structured prediction. *IEEE Signal Processing Magazine*, 2018 (cited on pages 93, 133).

[7] A. Arnab, S. Jayasumana, S. Zheng, and P. H. S. Torr. Higher order conditional random fields in deep neural networks. In *European Conference on Computer Vision*, 2016 (cited on pages 58, 59, 66, 96).

[8] A. Arnab and P. H. S. Torr. Bottom-up instance segmentation using deep higher-order crfs. In *British Machine Vision Conference*, 2016 (cited on pages 58, 64, 69, 75, 76, 86, 95, 96, 115, 165).

[9] A. Arnab and P. H. S. Torr. Pixelwise instance segmentation with a dynamically instantiated network. In *Computer Vision and Pattern Recognition*, 2017 (cited on pages 86, 87, 92, 93, 95, 96, 98, 165).

[10] A. Athalye and N. Carlini. On the robustness of the cvpr 2018 white-box adversarial example defenses. In *arXiv preprint arXiv:1804.03286*, 2018 (cited on pages 118, 133).

[11] A. Athalye, N. Carlini, and D. Wagner. Obfuscated gradients give a false sense of security: circumventing defenses to adversarial examples. In *International Conference on Machine Learning*, 2018 (cited on pages 118, 127, 130, 133, 167).

[12] A. Athalye and I. Sutskever. Synthesizing robust adversarial examples. In *arXiv preprint arXiv:1707.07397v1*, 2017 (cited on page 133).

[13] V. Badrinarayanan, A. Handa, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling. In *arXiv preprint arXiv:1511.00561*, 2015 (cited on pages 119, 139, 141).

[14] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*, 2015 (cited on page 164).

[15] M. Bai and R. Urtasun. Deep watershed transform for instance segmentation. In *Computer Vision and Pattern Recognition*, 2017 (cited on pages 70, 71, 87, 95, 98, 165).

[16] A. Bansal, X. Chen, B. Russell, A. Gupta, and D. Ramanan. Pixelnet: representation of the pixels, by the pixels, and for the pixels. In *arXiv preprint arXiv:1702.06506*, 2017 (cited on page 89).

[17] P. Baqué, T. Bagautdinov, F. Fleuret, and P. Fua. Principled parallel mean-field inference for discrete random fields. In *Computer Vision and Pattern Recognition*, 2016 (cited on page 39).

[18] D. Barber. *Bayesian reasoning and machine learning*. 2012 (cited on page 20).

[19] H. G. Barrow and J. Tenenbaum. Interpreting line drawings as three-dimensional surfaces, 1981 (cited on page 115).

[20] A. Bearman, O. Russakovsky, V. Ferrari, and L. Fei-Fei. What's the point: semantic segmentation with point supervision. In *European Conference on Computer Vision*, 2016 (cited on pages 86, 88, 89).

[21] Y. Bengio, N. Léonard, and A. Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. In *arXiv preprint arXiv:1308.3432*, 2013 (cited on page 130).

[22] I. Biederman, R. J. Mezzanotte, and J. C. Rabinowitz. Scene perception: detecting and judging objects undergoing relational violations. *Cognitive psychology*, 1982 (cited on page 7).

[23] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, et al. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, 2013 (cited on pages 114, 116, 117).

[24] B. Biggio, B. Nelson, and P. Laskov. Poisoning attacks against support vector machines. In *International Conference on Machine Learning*, 2012 (cited on page 123).

[25] N. Bjorck, C. P. Gomes, B. Selman, and K. Q. Weinberger. Understanding batch normalization. In *Conference on Neural Information Processing Systems*, 2018 (cited on page 19).

[26] F. Bogo, A. Kanazawa, C. Lassner, P. Gehler, J. Romero, et al. Keep it SMPL: automatic estimation of 3d human pose and shape from a single image. In *European Conference on Computer Vision*, 2016 (cited on page 170).

[27] T. Boult, S Cruz, A Dhamija, M Gunther, J Henrydoss, et al. Learning and the unknown: surveying steps toward open world recognition. In *AAAI Conference on Artificial Intelligence*, 2019 (cited on page 168).

[28] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11, 2001 (cited on page 20).

[29] T. B. Brown, D. Mané, A. Roy, M. Abadi, and J. Gilmer. Adversarial patch. In *arXiv preprint arXiv:1712.09665*, 2017 (cited on page 133).

[30] J. Buckman, A. Roy, C. Raffel, and I. Goodfellow. Thermometer encoding: one hot way to resist adversarial examples. In *International Conference on Learning Representations*, 2018 (cited on page 133).

[31] R. Bunel, I. Turkaslan, P. H. Torr, P. Kohli, and M. P. Kumar. Piecewise linear neural network verification: a comparative study. In *arXiv preprint arXiv:1711.00455*, 2017 (cited on page 118).

[32] J. Canny. A computation approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1986 (cited on page 17).

[33] P. Carbonetto and N. D. Freitas. Conditional mean field. In *Conference on Neural Information Processing Systems*, 2007 (cited on page 134).

[34] N. Carlini and D. Wagner. Adversarial examples are not easily detected: bypassing ten detection methods. In *arXiv preprint arXiv:1705.07263v1*, 2017 (cited on pages 115, 118, 133).

[35] N. Carlini and D. Wagner. Defensive distillation is not robust to adversarial examples. In *arXiv preprint arXiv:1607.04311v1*, 2016 (cited on pages 115, 118).

[36] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy*, 2017 (cited on pages 115–117).

[37] J. Carreira, R. Caseiro, J. Batista, and C. Sminchisescu. Free-form region description with second-order pooling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2014 (cited on pages 48, 171).

[38] J. Carreira, R. Caseiro, J. Batista, and C. Sminchisescu. Semantic segmentation with second-order pooling. In *European Conference on Computer Vision*. 2012 (cited on page 32).

[39] K. Chalupka, P. Perona, and F. Eberhardt. Visual causal feature learning. In *UAI*, 2015 (cited on page 115).

[40] S. Chandra and I. Kokkinos. Fast, exact and multi-scale inference for semantic image segmentation with deep gaussian crfs. In *European Conference on Computer Vision*, 2016 (cited on pages 115, 164).

[41] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, et al. Shapenet: an information-rich 3d model repository. In *arXiv preprint arXiv:1512.03012*, 2015 (cited on page 170).

[42] A. Chaudhry, P. K. Dokania, and P. H. Torr. Discovering class-specific pixels for weakly-supervised semantic segmentation. In *British Machine Vision Conference*, 2017 (cited on page 87).

[43] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018 (cited on pages 55, 94, 98, 115, 119–121, 125, 136, 139, 140, 144, 157, 164).

[44] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *International Conference on Learning Representations*, 2015 (cited on pages 16, 24, 30–33, 41, 48, 52, 58, 92, 115, 125).

[45] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. Rethinking atrous convolution for semantic image segmentation. In *arXiv preprint arXiv:1706.05587*, 2017 (cited on pages 94, 164).

[46] L.-C. Chen, A. Schwing, A. Yuille, and R. Urtasun. Learning deep structured models. In *International Conference on Machine Learning*, 2015 (cited on page 58).

[47] L.-C. Chen, Y. Yang, J. Wang, W. Xu, and A. L. Yuille. Attention to scale: scale-aware semantic image segmentation. In *Computer Vision and Pattern Recognition*, 2016 (cited on pages 41, 48).

[48] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *European Conference on Computer Vision*, 2018 (cited on pages 164, 171).

[49] Y.-H. Chen, W.-Y. Chen, Y.-T. Chen, B.-C. Tsai, Y.-C. Frank Wang, et al. No more discrimination: cross city adaptation of road scene segmenters. In *International Conference on Computer Vision*, 2017 (cited on page 6).

[50] Y.-T. Chen, X. Liu, and M.-H. Yang. Multi-instance object segmentation with occlusion handling. In *Computer Vision and Pattern Recognition*, 2015 (cited on pages 57, 60–62, 69, 75, 76, 96).

[51] M.-M. Cheng, N. J. Mitra, X. Huang, P. H. Torr, and S.-M. Hu. Global contrast based salient region detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015 (cited on pages 87, 91).

[52] A. Choromanska, M. Henaff, M. Mathieu, G. B. Arous, and Y. LeCun. The loss surfaces of multilayer networks. In *International Conference on Artificial Intelligence and Statistics*, 2015 (cited on page 19).

[53] M. M. Chun and Y. Jiang. Contextual cueing: implicit learning and memory of visual context guides spatial attention. *Cognitive psychology*, 1998 (cited on page 7).

[54] M. Cisse, Y. Adi, N. Neverova, and J. Keshet. Houdini: fooling deep structured prediction models. In *Conference on Neural Information Processing Systems*, 2017 (cited on pages 116–118).

[55] M. Cisse, P. Bojanowski, E. Grave, Y. Dauphin, and N. Usunier. Parseval networks: improving robustness to adversarial examples. In *International Conference on Machine Learning*, 2017 (cited on pages 116, 136).

[56] R. Clark, M. Bloesch, J. Czarnowski, S. Leutenegger, and A. J. Davison. Ls-net: learning to solve nonlinear least squares for monocular stereo. In *European Conference on Computer Vision*, 2018 (cited on page 164).

[57] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, et al. The cityscapes dataset for semantic urban scene understanding. In *Computer Vision and Pattern Recognition*, 2016 (cited on pages 4–6, 20, 56, 67, 70, 71, 85, 93, 110, 119, 120, 165, 167, 168).

[58] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 1995 (cited on page 13).

[59] G. Csurka. A comprehensive survey on domain adaptation for visual applications. In *Domain Adaptation in Computer Vision Applications*. 2017 (cited on page 169).

[60] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, 2004 (cited on page 13).

[61] E. D. Cubuk, B. Zoph, S. S. Schoenholz, and Q. V. Le. Intriguing properties of adversarial examples. In *arXiv preprint arXiv:1711.02846*, 2017 (cited on page 124).

[62] J. Dai, K. He, Y. Li, S. Ren, and J. Sun. Instance-sensitive fully convolutional networks. In *European Conference on Computer Vision*, 2016 (cited on pages 56, 70).

[63] J. Dai, K. He, and J. Sun. Boxsup: exploiting bounding boxes to supervise convolutional networks for semantic segmentation. In *International Conference on Computer Vision*, 2015 (cited on pages 34, 41–43, 48, 86, 88, 95, 125).

[64] J. Dai, K. He, and J. Sun. Convolutional feature masking for joint object and stuff segmentation. *Computer Vision and Pattern Recognition*, 2015 (cited on pages 32, 41, 48, 57, 70).

[65] J. Dai, K. He, and J. Sun. Instance-aware semantic segmentation via multi-task network cascades. In *Computer Vision and Pattern Recognition*, 2016 (cited on pages 56–58, 62, 64, 66, 70–72, 80, 81, 86, 87, 165).

[66] J. Dai, Y. Li, K. He, and J. Sun. R-fcn: object detection via region-based fully convolutional networks. In *Conference on Neural Information Processing Systems*, 2016 (cited on pages 55, 67).

[67] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition*, 2005 (cited on pages 13, 17).

[68] N. Dalvi, P. Domingos, S. Sanghai, and D. Verma. Adversarial classification. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2004 (cited on pages 114, 116).

[69] B. De Brabandere, D. Neven, and L. Van Gool. Semantic instance segmentation with a discriminative loss function. In *Computer Vision and Pattern Recognition Workshops*, 2017 (cited on page 87).

[70] J. De Fauw, J. R. Ledsam, B. Romera-Paredes, S. Nikolov, N. Tomasev, et al. Clinically applicable deep learning for diagnosis and referral in retinal disease. *Nature medicine*, 2018 (cited on page 2).

[71] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, et al. ImageNet: A Large-Scale Hierarchical Image Database. In *Computer Vision and Pattern Recognition*, 2009 (cited on pages 5, 14).

[72] Z. Deng, M. Zhai, L. Chen, Y. Liu, S. Muralidharan, et al. Deep structured models for group activity recognition. In *British Machine Vision Conference*, 2015 (cited on page 32).

[73] J. Domke. Learning graphical model parameters with approximate marginal inference. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2013 (cited on page 32).

[74] J. Dong, Q. Chen, S. Yan, and A. Yuille. Towards unified object detection and semantic segmentation. In *European Conference on Computer Vision*, 2014 (cited on page 48).

[75] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. Carla: an open urban driving simulator. In *Conference on Robot Learning (CoRL)*, 2017 (cited on page 169).

[76] G. K. Dziugaite, Z. Ghahramani, and D. M. Roy. A study of the effect of jpg compression on adversarial images. In *arXiv preprint arXiv:1608.00853v1*, 2016 (cited on pages 120, 123, 129).

[77] L. Engstrom, B. Tran, D. Tsipras, L. Schmidt, and A. Madry. A rotation and a translation suffice: fooling cnns with simple transformations. In *arXiv preprint arXiv:1712.02779*, 2017 (cited on page 169).

[78] S. A. Eslami, N. Heess, T. Weber, Y. Tassa, D. Szepesvari, et al. Attend, infer, repeat: fast scene understanding with generative models. In *Conference on Neural Information Processing Systems*, 2016 (cited on page 170).

[79] A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, et al. Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, 2017 (cited on pages 2, 114).

[80] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, et al. The pascal visual object classes challenge: a retrospective. *International Journal of Computer Vision*, 2015 (cited on page 97).

[81] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 2010 (cited on pages 4, 6, 30, 40, 57, 64, 67, 86, 93, 94, 97, 102, 109, 119, 120, 165, 167).

[82] I. Evtimov, K. Eykholt, E. Fernandes, T. Kohno, B. Li, et al. Robust physical-world attacks on machine learning models. In *arXiv preprint arXiv:1707.08945v3*, 2017 (cited on pages 114, 133).

[83] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2013 (cited on page 32).

[84] A. Fawzi, H. Fawzi, and O. Fawzi. Adversarial vulnerability for any classifier. In *Conference on Neural Information Processing Systems*, 2018 (cited on page 167).

[85] A. Fawzi and P. Frossard. Manitest: are classifiers really invariant? In *British Machine Vision Conference*, 2015 (cited on page 125).

[86] A. Fawzi, S.-M. Moosavi-Dezfooli, P. Frossard, and S. Soatto. Empirical study of the topology and geometry of deep networks. In *Computer Vision and Pattern Recognition*, 2018 (cited on page 167).

[87] R. Feinman, R. R. Curtin, S. Shintre, and A. B. Gardner. Detecting adversarial samples from artifacts. In *arXiv preprint arXiv:1703.00410v2*, 2017 (cited on page 118).

[88] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010 (cited on pages 17, 57, 63).

[89] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 2004 (cited on pages 17, 36, 37, 40).

[90] V. Fischer, M. C. Kumar, J. H. Metzen, and T. Brox. Adversarial examples for semantic image segmentation. In *International Conference on Learning Representations Workshops*, 2017 (cited on page 117).

[91] D. A. Forsyth, J. Malik, M. M. Fleck, H. Greenspan, T. Leung, et al. *Finding pictures of objects in large collections of images*. 1996 (cited on pages 6, 86, 123, 165).

[92] J. Fu, J. Liu, H. Tian, Y. Li, Y. Bao, et al. Dual attention network for scene segmentation. In *Computer Vision and Pattern Recognition*, 2019 (cited on page 164).

[93] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: representing model uncertainty in deep learning. In *International Conference on Machine Learning*, 2016 (cited on page 168).

[94] J. Gao, B. Wang, and Y. Qi. Deepmask: masking dnn models for robustness against adversarial samples. In *International Conference on Learning Representations Workshops*, 2017 (cited on page 136).

[95] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: the kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013 (cited on page 3).

[96] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, et al. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations*, 2018 (cited on page 167).

[97] G. Ghiasi and C. C. Fowlkes. Laplacian pyramid reconstruction and refinement for semantic segmentation. In *European Conference on Computer Vision*, 2016 (cited on page 16).

[98] R. Girshick. Fast r-cnn. In *International Conference on Computer Vision*, 2015 (cited on pages 18, 34).

[99] R. Girshick. Joint coco and mapillary recognition challenge workshop, 2018. Available: `http://presentations.cocodataset.org/ECCV18/COCO18-Detect-Overview.pdf`. Accessed 17 April 2019 (cited on page 5).

[100] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition*, 2014 (cited on pages 14, 18, 19, 57).

[101] A. Giusti, D. C. Ciresan, J. Masci, L. M. Gambardella, and J. Schmidhuber. Fast image scanning with deep max-pooling convolutional neural networks. In *International Conference on Image Processing*, 2013 (cited on page 14).

[102] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International Conference on Artificial Intelligence and Statistics*, 2010 (cited on page 14).

[103] J. M. Gonfaus, X. Boix, J. Van de Weijer, A. D. Bagdanov, J. Serrat, et al. Harmony potentials for joint classification and segmentation. In *Computer Vision and Pattern Recognition*, 2010 (cited on page 31).

[104] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015 (cited on pages 114, 116, 118, 119, 123, 136).

[105] K. Grosse, P. Manoharan, N. Papernot, M. Backes, and P. McDaniel. On the (statistical) detection of adversarial examples. In *arXiv preprint arXiv:1702.06280v1*, 2017 (cited on page 118).

[106] S. Gu and L. Rigazio. Towards deep neural network architectures robust to adversarial examples. In *International Conference on Learning Representations Workshops*, 2015 (cited on page 115).

[107] C. Guo, M. Rana, M. Cissé, and L. van der Maaten. Countering adversarial images using input transformations. In *International Conference on Learning Representations*, 2018 (cited on pages 115, 118, 129, 133, 167).

[108] S. Gupta, R. Girshick, P. Arbelaez, and J. Malik. Learning rich features from RGB-D images for object detection and segmentation. In *European Conference on Computer Vision*. 2014 (cited on page 56).

[109] J. Hammersley and P. Clifford. Markov random fields on finite graphs and lattices. Unpublished manuscript, URL `http://www.statslab.cam.ac.uk/~grg/books/hammfest/hamm-cliff.pdf`, 1971 (cited on page 21).

[110] B. Hariharan, P. Arbeláez, L. Bourdev, S. Maji, and J. Malik. Semantic contours from inverse detectors. In *International Conference on Computer Vision*, 2011 (cited on pages 40, 67, 94, 119, 138, 165).

[111] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Hypercolumns for object segmentation and fine-grained localization. *Computer Vision and Pattern Recognition*, 2015 (cited on pages 56–58, 60–62, 64, 66, 70, 71, 165).

[112] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Simultaneous detection and segmentation. In *European Conference on Computer Vision*. 2014 (cited on pages 30, 32, 56–58, 61, 62, 64, 66, 67, 69, 70, 75, 76, 87, 94, 96, 165).

[113] R. Hartley, F. Kahl, and P. Torr. *Solutions of Markov Random Fields*. Cambridge University Press, 2019 (cited on pages 20, 25–27, 38).

[114] Z. Hayder, X. He, and M. Salzmann. Shape-aware instance segmentation. In *arXiv preprint arXiv:1612.03129*, 2016 (cited on page 71).

[115] K. He. Tutorial on deep residual networks, ICML, 2016. Available: `http://kaiminghe.com/icml16tutorial/icml2016_tutorial_deep_residual_networks_kaiminghe.pdf`. Accessed 5 May 2019 (cited on page 18).

[116] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *International Conference on Computer Vision*, 2017 (cited on pages 86–88, 165).

[117] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition*, 2016 (cited on pages 14, 19, 55, 85, 94, 109, 115, 119, 121, 164).

[118] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In *Computer Vision and Pattern Recognition*, 2015 (cited on page 14).

[119] W. He, J. Wei, X. Chen, N. Carlini, and D. Song. Adversarial example defenses: ensembles of weak defenses are not strong. In *arXiv preprint arXiv:1706.04701v1*, 2017 (cited on pages 115, 118).

[120] X. He and S. Gould. An Exemplar-based CRF for Multi-instance Object Segmentation. In *Computer Vision and Pattern Recognition*, 2014 (cited on page 62).

[121] X. He and R. S. Zemel. Learning hybrid models for image annotation with partially labeled data. In *Conference on Neural Information Processing Systems*, 2009 (cited on page 88).

[122] D. Hendrycks and T. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2019 (cited on page 169).

[123] J. F. Henriques and A. Vedaldi. Warped convolutions: efficient invariance to spatial transformations. In *International Conference on Machine Learning*, 2017 (cited on page 125).

[124] S. L. Hicks, I. Wilson, L. Muhammed, J. Worsfold, S. M. Downes, et al. A depth-based head-mounted visual display to aid navigation in partially sighted individuals. *PLOS ONE*, 2013 (cited on page 3).

[125] G. Hinton, L. Deng, D. Yu, G. Dahl, A.-r. Mohamed, et al. Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal Processing Magazine*, 2012 (cited on page 19).

[126] M. Holschneider, R. Kronland-Martinet, J. Morlet, and P. Tchamitchian. A real-time algorithm for signal analysis with the help of the wavelet transform. In *Wavelets*. 1990 (cited on page 16).

[127] S. Hong, D. Yeo, S. Kwak, H. Lee, and B. Han. Weakly supervised semantic segmentation using web-crawled videos. In *Computer Vision and Pattern Recognition*, pages 7322–7330, 2017 (cited on page 88).

[128] R. Hu, P. Dollár, K. He, T. Darrell, and R. Girshick. Learning to segment every thing. In *arXiv preprint arXiv:1711.10370*, 2017 (cited on page 88).

[129] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, et al. Speed/accuracy trade-offs for modern convolutional object detectors. In *Computer Vision and Pattern Recognition*, 2017 (cited on pages 19, 94, 140, 164).

[130] S. Ioffe and C. Szegedy. Batch normalization: accelerating deep network training by reducing internal covariate shift. In *arXiv preprint arXiv:1502.03167*, 2015 (cited on page 14).

[131] C. Ionescu, O. Vantzos, and C. Sminchisescu. Matrix backpropagation for deep networks with structured layers. In *International Conference on Computer Vision*, 2015 (cited on page 32).

[132] J. Janai, F. Güney, A. Behl, and A. Geiger. Computer vision for autonomous vehicles: problems, datasets and state-of-the-art. In *arXiv preprint arXiv:1704.05519v1*, 2017 (cited on page 114).

[133] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, et al. Caffe: convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014 (cited on page 138).

[134] J. H. Kappes, B. Andres, F. A. Hamprecht, C. Schnörr, S. Nowozin, et al. A comparative study of modern inference techniques for structured discrete energy minimization problems. *International Journal of Computer Vision*, 2015 (cited on page 20).

[135] G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer. Reluplex: an efficient smt solver for verifying deep neural networks. In *International Conference on Computer Aided Verification*, 2017 (cited on page 118).

[136] K. Kawaguchi. Deep learning without poor local minima. In *Conference on Neural Information Processing Systems*, 2016 (cited on page 19).

[137] A. Kendall and Y. Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Conference on Neural Information Processing Systems*, 2017 (cited on page 168).

[138] A. Kerckhoffs. La cryptographie militaire. *Journal des sciences militaires*, 1883 (cited on pages 129, 133).

[139] A. Khoreva, R. Benenson, J. Hosang, M. Hein, and B. Schiele. Simple does it: weakly supervised instance and semantic segmentation. In *Computer Vision and Pattern Recognition*, 2017 (cited on pages 88, 89, 92, 95, 96).

[140] D. P. Kingma and J. Ba. Adam: a method for stochastic optimization. In *International Conference on Learning Representations*, 2015 (cited on page 14).

[141] A. Kirillov, R. Girshick, K. He, and P. Dollár. Panoptic feature pyramid networks. In *Computer Vision and Pattern Recognition*, 2019 (cited on page 166).

[142] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár. Panoptic segmentation. In *arXiv preprint arXiv:1801.00868*, 2018 (cited on pages 86, 87, 95, 165, 166).

[143] A. Kirillov, E. Levinkov, B. Andres, B. Savchynskyy, and C. Rother. Instancecut: from edges to instances with multicut. In *Computer Vision and Pattern Recognition*, 2017 (cited on pages 71, 87, 98, 165).

[144] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 2017 (cited on page 168).

[145] K. M. Kitani, B. D. Ziebart, J. A. Bagnell, and M. Hebert. Activity forecasting. In *European Conference on Computer Vision*, 2012 (cited on page 3).

[146] P. W. Koh and P. Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning*, 2017 (cited on page 123).

[147] P. Kohli, M. P. Kumar, and P. H. Torr. P3 & beyond: solving energies with higher order cliques. In *Computer Vision and Pattern Recognition*, 2007 (cited on pages 22, 37).

[148] P. Kohli, L. Ladicky, and P. H. Torr. Robust higher order potentials for enforcing label consistency. *International Journal of Computer Vision*, 2009 (cited on pages 31, 33, 42).

[149] I. Kokkinos. Pushing the boundaries of boundary detection using deep learning. In *International Conference on Learning Representations*, 2016 (cited on pages 41, 48).

[150] I. Kokkinos. Ubernet: training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In *Computer Vision and Pattern Recognition*, 2017 (cited on pages 71, 170).

[151] A. Kolesnikov and C. H. Lampert. Seed, expand and constrain: three principles for weakly-supervised image segmentation. In *European Conference on Computer Vision*, 2016 (cited on pages 86, 88–90, 92).

[152] D. Koller and N. Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009 (cited on pages 20, 21, 27, 30, 38).

[153] V. Kolmogorov and R. Zabih. What energy functions can be minimizedvia graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2004 (cited on pages 20, 23).

[154] P. Krähenbühl and V. Koltun. Efficient inference in fully connected CRFs with Gaussian edge potentials. In *Conference on Neural Information Processing Systems*, 2011 (cited on pages 23, 24, 27, 31–35, 39, 59, 64, 92, 93, 133, 157).

[155] P. Krähenbühl and V. Koltun. Parameter learning and convergent inference for dense random fields. In *International Conference on Machine Learning*, 2013 (cited on pages 32, 33, 65).

[156] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, et al. Visual genome: connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 2017 (cited on page 3).

[157] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Conference on Neural Information Processing Systems*. 2012 (cited on pages 13, 14, 19, 29, 115).

[158] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. Tenenbaum. Deep convolutional inverse graphics network. In *Conference on Neural Information Processing Systems*, 2015 (cited on page 170).

[159] A. Kundu, Y. Li, and J. M. Rehg. 3d-rcnn: instance-level 3d object reconstruction via render-and-compare. In *Computer Vision and Pattern Recognition*, 2018 (cited on page 170).

[160] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial examples in the physical world. In *International Conference on Learning Representations Workshops*, 2017 (cited on pages 114, 131, 133).

[161] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial machine learning at scale. In *International Conference on Learning Representations*, 2017 (cited on pages 114–124, 126, 136).

[162] L. Ladicky, C. Russell, P. Kohli, and P. H. Torr. Graph cut based inference with co-occurrence statistics. In *European Conference on Computer Vision*. 2010 (cited on pages 22, 31).

[163] L. Ladicky, C. Russell, P. Kohli, and P. H. Torr. Associative hierarchical crfs for object class image segmentation. In *International Conference on Computer Vision*, 2009 (cited on pages 24, 30–32).

[164] L. Ladický, P. Sturgess, K. Alahari, C. Russell, and P. H. S. Torr. What, where and how many? combining object detectors and crfs. In *European Conference on Computer Vision*, 2010 (cited on pages 22, 30–32, 35, 60).

[165] J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning*, 2001 (cited on page 21).

[166] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, et al. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1989 (cited on page 14).

[167] K. Lenc and A. Vedaldi. R-cnn minus r. In *British Machine Vision Conference*, 2015 (cited on page 19).

[168] E. Levinkov, J. Uhrig, S. Tang, M. Omran, E. Insafutdinov, et al. Joint graph decomposition & node labeling: problem, algorithms, applications. In *Computer Vision and Pattern Recognition*, pages 6012–6020, 2017 (cited on page 71).

[169] K. Li, B. Hariharan, and J. Malik. Iterative Instance Segmentation. In *Computer Vision and Pattern Recognition*, 2016 (cited on pages 56, 57, 61, 66, 70, 77).

[170] Q. Li, A. Arnab, and P. H. Torr. Holistic, instance-level human parsing. In *British Machine Vision Conference*, 2017 (cited on page 89).

[171] Y. Li, X. Chen, Z. Zhu, L. Xie, G. Huang, et al. Attention-guided unified network for panoptic segmentation. In *Computer Vision and Pattern Recognition*, 2019 (cited on page 166).

[172] Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei. Fully convolutional instance-aware semantic segmentation. In *Computer Vision and Pattern Recognition*, 2017 (cited on pages 72, 82, 86, 87).

[173] Z. Li and D. Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018 (cited on page 168).

[174] X. Liang, Y. Wei, X. Shen, Z. Jie, J. Feng, et al. Reversible recursive instance-level object segmentation. In *Computer Vision and Pattern Recognition*, 2016 (cited on pages 56, 58).

[175] X. Liang, Y. Wei, X. Shen, J. Yang, L. Lin, et al. Proposal-free network for instance-level object segmentation. *arXiv preprint arXiv:1509.02636*, 2015 (cited on pages 58, 66, 69, 73, 75, 76, 95, 96).

[176] F. Liao, M. Liang, Y. Dong, T. Pang, J. Zhu, et al. Defense against adversarial attacks using high-level representation guided denoiser. In *Computer Vision and Pattern Recognition*, 2018 (cited on pages 116, 118, 133, 167).

[177] D. Lin, J. Dai, J. Jia, K. He, and J. Sun. Scribblesup: scribble-supervised convolutional networks for semantic segmentation. In *Computer Vision and Pattern Recognition*, 2016 (cited on pages 86–88).

[178] G. Lin, C. Shen, and I. Reid. Efficient piecewise training of deep structured models for semantic segmentation. In *Computer Vision and Pattern Recognition*, 2016 (cited on pages 30, 31, 48, 58, 125).

[179] G. Lin, C. Shen, I. D. Reid, and A. van den Hengel. Deeply learning the messages in message passing inference. In *Conference on Neural Information Processing Systems*, 2015 (cited on page 31).

[180] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, et al. Microsoft coco: common objects in context. In *European Conference on Computer Vision*. 2014 (cited on pages 4, 40, 41, 67, 82, 86, 94, 97, 102, 109, 119, 138).

[181] M.-Y. Liu, O. Tuzel, A. Veeraraghavan, and R. Chellappa. Fast directional chamfer matching. In *Computer Vision and Pattern Recognition*, 2010 (cited on page 63).

[182] S. Liu, J. Jia, S. Fidler, and R. Urtasun. Sgn: sequential grouping networks for instance segmentation. In *International Conference on Computer Vision*, 2017 (cited on pages 87, 96, 98, 99, 165).

[183] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia. Path aggregation network for instance segmentation. In *arXiv preprint arXiv:1803.01534*, 2018 (cited on pages 86, 87, 165).

[184] S. Liu, X. Qi, J. Shi, H. Zhang, and J. Jia. Multi-scale patch aggregation (mpa) for simultaneous detection and segmentation. In *Computer Vision and Pattern Recognition*, 2016 (cited on pages 56, 58, 60, 64, 66, 69–71, 73, 75, 76, 86, 87, 96).

[185] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, et al. Ssd: single shot multibox detector. In *European Conference on Computer Vision*, 2016 (cited on page 19).

[186] W. Liu, A. Rabinovich, and A. C. Berg. Parsenet: looking wider to see better. *arXiv preprint arXiv:1506.04579*, 2015 (cited on page 41).

[187] Y. Liu, X. Chen, C. Liu, and D. Song. Delving into transferable adversarial examples and black-box attacks. In *International Conference on Learning Representations*, 2017 (cited on pages 116–118, 123).

[188] Z. Liu, X. Li, P. Luo, C. C. Loy, and X. Tang. Semantic image segmentation via deep parsing network. In *International Conference on Computer Vision*, 2015 (cited on pages 30, 31, 41, 48).

[189] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Computer Vision and Pattern Recognition*, 2015 (cited on pages 14, 15, 24, 30–32, 40, 41, 43, 48, 52, 59, 66, 115, 119, 121, 134, 138, 139).

[190] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black. Smpl: a skinned multi-person linear model. *ACM Transactions on Graphics (TOG)*, 2015 (cited on page 170).

[191] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 2004 (cited on page 13).

[192] J. Lu, H. Sibai, E. Fabry, and D. Forsyth. No need to worry about adversarial examples in object detection in autonomous vehicles. In *Computer Vision and Pattern Recognition Workshops*, 2017 (cited on pages 132, 133).

[193] J. Lu, H. Sibai, E. Fabry, and D. Forsyth. Standard detectors aren't (currently) fooled by physical adversarial stop signs. In *arXiv preprint arXiv:1710.03337v1*, 2017 (cited on page 133).

[194] P. Luc, C. Couprie, S. Chintala, and J. Verbeek. Semantic segmentation using adversarial networks. In *Conference on Neural Information Processing Systems Workshop*, 2016 (cited on page 164).

[195] D. J. MacKay. A practical bayesian framework for backpropagation networks. *Neural computation*, 1992 (cited on page 168).

[196] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018 (cited on pages 114, 115, 117–124, 136, 167, 169).

[197] J. L. McClelland, B. L. McNaughton, and R. C. O'reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, 1995 (cited on page 167).

[198] M. McCloskey and N. J. Cohen. Catastrophic interference in connectionist networks: the sequential learning problem. In *Psychology of learning and motivation*. 1989 (cited on page 167).

[199] B. H. Menze, A. Jakab, S. Bauer, J. Kalpathy-Cramer, K. Farahani, et al. The multimodal brain tumor image segmentation benchmark (BRATS). *IEEE Transactions on Medical Imaging*, 2015 (cited on page 2).

[200] J. H. Metzen, T. Genewein, V. Fischer, and B. Bischoff. On detecting adversarial perturbations. In *International Conference on Learning Representations*, 2017 (cited on page 118).

[201] J. H. Metzen, M. C. Kumar, T. Brox, and V. Fischer. Universal adversarial perturbations against semantic image segmentation. In *International Conference on Computer Vision*, 2017 (cited on page 118).

[202] K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *International Journal of Computer Vision*, 2004 (cited on page 13).

[203] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, et al. A comparison of affine region detectors. *International Journal of Computer Vision*, 2005 (cited on page 13).

[204] I. Misra, A. Shrivastava, A. Gupta, and M. Hebert. Cross-stitch Networks for Multi-task Learning. In *Computer Vision and Pattern Recognition*, 2016 (cited on page 71).

[205] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard. Universal adversarial perturbations. In *Computer Vision and Pattern Recognition*, 2017 (cited on page 118).

[206] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Computer Vision and Pattern Recognition*, 2016 (cited on page 117).

[207] G. Morris. Smart glasses for blind 'in shops by 2016', 2014. Available: `https://news.sky.com/story/smart-glasses-for-blind-in-shops-by-2016-10393683`. Accessed 16 April 2019 (cited on page 3).

[208] M. Mostajabi, P. Yadollahpour, and G. Shakhnarovich. Feedforward semantic segmentation with zoom-out features. In *Computer Vision and Pattern Recognition*, 2015 (cited on page 48).

[209] R. Mottaghi, X. Chen, X. Liu, N.-G. Cho, S.-W. Lee, et al. The role of context for object detection and semantic segmentation in the wild. In *Computer Vision and Pattern Recognition*, 2014 (cited on page 41).

[210] K. P. Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012 (cited on page 134).

[211] A. Myronenko. 3d mri brain tumor segmentation using autoencoder regularization. In *International MICCAI Brain Lesion Workshop*, 2018 (cited on page 3).

[212] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *International Conference on Machine Learning*, 2010 (cited on page 14).

[213] N. Narodytska and S. P. Kasiviswanathan. Simple black-box adversarial perturbations for deep networks. In *Computer Vision and Pattern Recognition Workshops*, 2017 (cited on page 117).

[214] G. Neuhold, T. Ollmann, S. Rota Bulò, and P. Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *International Conference on Computer Vision*, 2017 (cited on page 168).

[215] H. Noh, S. Hong, and B. Han. Learning deconvolution network for semantic segmentation. In *International Conference on Computer Vision*, 2015 (cited on pages 16, 48).

[216] S. J. Oh, R. Benenson, A. Khoreva, Z. Akata, M. Fritz, et al. Exploiting saliency for object segmentation from image level labels. In *Computer Vision and Pattern Recognition*, 2017 (cited on pages 87, 91).

[217] A. Oliva and A. Torralba. The role of context in object recognition. *Trends in cognitive sciences*, 2007 (cited on page 7).

[218] W. H. Organization. New estimates of visual impairment and blindness, 2010. Available: `https://www.who.int/blindness/publications/globaldata/en/`. Accessed 16 April 2019 (cited on page 3).

[219] D. P. Papadopoulos, A. D. Clarke, F. Keller, and V. Ferrari. Training object class detectors from eye tracking data. In *European Conference on Computer Vision*, 2014 (cited on pages 85, 110).

[220] D. P. Papadopoulos, J. R. Uijlings, F. Keller, and V. Ferrari. Extreme clicking for efficient object annotation. In *International Conference on Computer Vision*, 2017 (cited on pages 85, 110).

[221] G. Papandreou, L. Chen, K. Murphy, and A. L. Yuille. Weakly- and semi-supervised learning of a DCNN for semantic image segmentation. In *International Conference on Computer Vision*, 2015 (cited on pages 41, 48, 86, 88, 95).

[222] N. Papernot, P. McDaniel, and I. Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. In *arXiv preprint arXiv:1605.07277v1*, 2016 (cited on page 114).

[223] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, et al. Practical black-box attacks against machine learning. In *2017 ACM Asia Conference on Computer and Communications Security*, 2017 (cited on pages 115, 118, 136).

[224] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, et al. The limitations of deep learning in adversarial settings. In *IEEE European Symposium on Security and Privacy*, 2016 (cited on page 117).

[225] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *IEEE Symposium on Security and Privacy*, 2016 (cited on pages 114, 115, 134, 136).

[226] R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, 2013 (cited on page 66).

[227] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello. Enet: a deep neural network architecture for real-time semantic segmentation. In *arXiv preprint arXiv:1606.02147v1*, 2016 (cited on pages 119, 122, 140, 141).

[228] D. Pathak, P. Krahenbuhl, and T. Darrell. Constrained convolutional neural networks for weakly supervised segmentation. In *International Conference on Computer Vision*, 2015 (cited on pages 88, 89).

[229] B. Pepik, R. Benenson, T. Ritschel, and B. Schiele. What is holding back convnets for detection? In *German Conference on Pattern Recognition*, 2015 (cited on page 125).

[230] F. Perronnin and C. Dance. Fisher kernels on visual vocabularies for image categorization. In *Computer Vision and Pattern Recognition*, 2007 (cited on page 13).

[231] P. O. Pinheiro and R. Collobert. From image-level to pixel-level labeling with convolutional networks. In *Computer Vision and Pattern Recognition*, 2015 (cited on page 88).

[232] P. O. Pinheiro, R. Collobert, and P. Dollar. Learning to segment object candidates. In *Conference on Neural Information Processing Systems*, 2015 (cited on page 56).

[233] P. O. Pinheiro, T.-Y. Lin, R. Collobert, and P. Dollar. Learning to refine object segments. In *European Conference on Computer Vision*, 2016 (cited on page 56).

[234] T. Pohlen, A. Hermans, M. Mathias, and B. Leibe. Full-resolution residual networks for semantic segmentation in street scenes. In *Computer Vision and Pattern Recognition*, 2017 (cited on page 89).

[235] B. T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 1964 (cited on page 14).

[236] L. Porzi, S. Rota Bulo, A. Colovic, and P. Kontschieder. Seamless scene segmentation. In *Computer Vision and Pattern Recognition*, 2019 (cited on page 166).

[237] A. Prakash, N. Moran, S. Garber, A. DiLillo, and J. Storer. Deflecting adversarial attacks with pixel deflection. In *Computer Vision and Pattern Recognition*, 2018 (cited on pages 118, 167).

[238] S. J. Prince. *Computer vision: models, learning, and inference*. Cambridge University Press, 2012 (cited on page 21).

[239] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, and S. Belongie. Objects in context. In *International Conference on Computer Vision*, 2007 (cited on page 31).

[240] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert. Icarl: incremental classifier and representation learning. In *Computer Vision and Pattern Recognition*, 2017 (cited on page 168).

[241] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: unified, real-time object detection. In *Computer Vision and Pattern Recognition*, 2016 (cited on page 19).

[242] M. Ren and R. S. Zemel. End-to-end instance segmentation with recurrent attention. In *Computer Vision and Pattern Recognition*, 2017 (cited on pages 71, 98).

[243] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Conference on Neural Information Processing Systems*, 2015 (cited on pages 18, 34, 40, 47, 58, 85, 94, 108).

[244] S. R. Richter, V. Vineet, S. Roth, and V. Koltun. Playing for data: ground truth from computer games. In *European Conference on Computer Vision*, 2016 (cited on page 169).

[245] G. Riegler, M. Rüther, and H. Bischof. Atgv-net: accurate depth super-resolution. In *European Conference on Computer Vision*, 2016 (cited on page 164).

[246] H. Robbins and S. Monro. A stochastic approximation method. *The annals of mathematical statistics*, 1951 (cited on page 14).

[247] L. G. Roberts. *Machine perception of three-dimensional solids*. PhD thesis, Massachusetts Institute of Technology, 1963 (cited on page 170).

[248] B. Romera-Paredes and P. H. Torr. Recurrent instance segmentation. In *European Conference on Computer Vision*, 2016 (cited on pages 58, 64, 65).

[249] J. Romero, D. Tzionas, and M. J. Black. Embodied hands: modeling and capturing hands and bodies together. *ACM Transactions on Graphics (TOG)*, 2017 (cited on page 170).

[250] O. Ronneberger, P. Fischer, and T. Brox. U-net: convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention*, 2015 (cited on page 16).

[251] S. Ross, D. Munoz, M. Hebert, and J. A. Bagnell. Learning message-passing inference machines for structured prediction. In *Computer Vision and Pattern Recognition*, 2011 (cited on page 32).

[252] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics (TOG)*, 2004 (cited on pages 35, 40, 89, 90, 99).

[253] D. E. Rumelhart, G. E. Hinton, R. J. Williams, et al. Learning representations by back-propagating errors. *Nature*, 1986 (cited on page 14).

[254] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 2015 (cited on pages 5, 14, 85, 91).

[255] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry. How does batch normalization help optimization? In *Conference on Neural Information Processing Systems*, 2018 (cited on page 19).

[256] L. Schmidt, S. Santurkar, D. Tsipras, K. Talwar, and A. Madry. Adversarially robust generalization requires more data. In *Conference on Neural Information Processing Systems*, 2018 (cited on page 167).

[257] H Scudder. Probability of error of some adaptive pattern-recognition machines. *IEEE Transactions on Information Theory*, 1965 (cited on page 88).

[258] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, et al. Grad-cam: visual explanations from deep networks via gradient-based localization. In *International Conference on Computer Vision*, 2017 (cited on page 91).

[259] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Proceedings of the 23rd ACM SIGSAC Conference on Computer and Communications Security*, 2016 (cited on pages 114, 131).

[260] E. Shelhamer, J. Long, and T. Darrell. Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017 (cited on page 138).

[261] E. Shelhamer, K. Rakelly, J. Hoffman, and T. Darrell. Clockwork convnets for video semantic segmentation. In *European Conference on Computer Vision Workshops*, 2016 (cited on pages 139, 140).

[262] X. Shen, A. Hertzmann, J. Jia, S. Paris, B. Price, et al. Automatic portrait segmentation for image stylization. In *Computer Graphics Forum*, 2016 (cited on page 3).

[263] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000 (cited on page 17).

[264] J. Shi, Q. Yan, L. Xu, and J. Jia. Hierarchical image saliency detection on extended cssd. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2016 (cited on pages 87, 91).

[265] J. Shotton, A. Blake, and R. Cipolla. Contour-based learning for object detection. In *International Conference on Computer Vision*, 2005 (cited on page 63).

[266] J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost for image understanding: multi-class object recognition and segmentation by jointly modeling texture, layout, and context. *International Journal of Computer Vision*, 2009 (cited on pages 23, 24, 31, 32).

[267] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015 (cited on pages 14, 30, 40, 59, 85, 96, 115, 119, 121, 164).

[268] K. Sirinukunwattana, J. P. Pluim, H. Chen, X. Qi, P.-A. Heng, et al. Gland segmentation in colon histology images: the glas challenge contest. *Medical image analysis*, 2017 (cited on page 2).

[269] J. Sivic and A. Zisserman. Video google: a text retrieval approach to object matching in videos. In *International Conference on Computer Vision*, 2003 (cited on page 13).

[270] Y. Song, T. Kim, S. Nowozin, S. Ermon, and N. Kushman. Pixeldefend: leveraging generative models to understand and defend against adversarial examples. In *International Conference on Learning Representations*, 2018 (cited on pages 118, 167).

[271] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 2014 (cited on page 14).

[272] D. Su, H. Zhang, H. Chen, J. Yi, P.-Y. Chen, et al. Is robustness the cost of accuracy?–a comprehensive study on the robustness of 18 deep image classification models. In *European Conference on Computer Vision*, 2018 (cited on page 167).

[273] J. Su, D. V. Vargas, and S. Kouichi. One pixel attack for fooling deep neural networks. In *arXiv preprint arXiv:1710.08864*, 2017 (cited on page 117).

[274] C. Sun, A. Shrivastava, S. Singh, and A. Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *International Conference on Computer Vision*, 2017 (cited on page 85).

[275] M. Sun, B.-s. Kim, P. Kohli, and S. Savarese. Relating things and stuff via objectproperty interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2014 (cited on pages 32, 60).

[276] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Conference on Neural Information Processing Systems*, 2014 (cited on page 19).

[277] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Computer Vision and Pattern Recognition*, 2016 (cited on page 124).

[278] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, et al. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014 (cited on pages 19, 114, 116, 118, 123).

[279] C. Tang and P. Tan. Ba-net: dense bundle adjustment network. In *International Conference on Learning Representations*, 2019 (cited on page 164).

[280] J. Thewlis, S. Zheng, P. H. Torr, and A. Vedaldi. Fully-trainable deep matching. In *British Machine Vision Conference*, 2016 (cited on page 165).

[281] P. Tokmakov, K. Alahari, and C. Schmid. Weakly-supervised semantic segmentation using motion cues. In *European Conference on Computer Vision*, pages 388–404. Springer, 2016 (cited on page 88).

[282] J. J. Tompson, A. Jain, Y. LeCun, and C. Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *Conference on Neural Information Processing Systems*, 2014 (cited on page 32).

[283] A. Torralba and A. A. Efros. Unbiased look at dataset bias. In *Computer Vision and Pattern Recognition*, 2011 (cited on page 6).

[284] F. Tramèr, A. Kurakin, N. Papernot, D. Boneh, and P. McDaniel. Ensemble adversarial training: attacks and defenses. In *arXiv preprint arXiv:1705.07204v2*, 2017 (cited on pages 118, 123).

[285] D. Tsipras, S. Santurkar, L. Engstrom, A. Turner, and A. Madry. Robustness may be at odds with accuracy. In *International Conference on Learning Representations*, 2019 (cited on page 167).

[286] J. Uesato, B. O'Donoghue, A. v. d. Oord, and P. Kohli. Adversarial risk and the dangers of evaluating against weak attacks. In *International Conference on Machine Learning*, 2018 (cited on pages 118, 127, 133, 167).

[287] J. Uhrig, M. Cordts, U. Franke, and T. Brox. Pixel-level encoding and depth layering for instance-level semantic labeling. In *German Conference on Pattern Recognition*, 2016 (cited on pages 70, 71, 98).

[288] J. R. Uijlings, K. E. van de Sande, T. Gevers, and A. W. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 2013 (cited on pages 17, 18).

[289] N. H. T. A. U.S. Department of Transportation. Tesla crash preliminary evaluation report, 2017. Available: `https://static.nhtsa.gov/odi/inv/2016/INCLA-PE16007-7876.PDF`. Accessed 26 May 2019 (cited on page 168).

[290] R. Vaillant, C. Monrocq, and Y. Le Cun. Original approach for the localisation of objects in images. *IEE Proceedings-Vision, Image and Signal Processing*, 1994 (cited on page 17).

[291] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P. H. Torr. End-to-end representation learning for correlation filter based tracking. In *Computer Vision and Pattern Recognition*, 2017 (cited on page 165).

[292] J. J. Van Rheede, I. R. Wilson, L. Di Bon-Conyers, S. Croxford, R. E. MacLaren, et al. Smart specs: electronic vision enhancement in real-life scenarios. *Investigative Ophthalmology & Visual Science*, 2016 (cited on page 3).

[293] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, et al. Attention is all you need. In *Conference on Neural Information Processing Systems*, 2017 (cited on page 164).

[294] A. Vedaldi and K. Lenc. Matconvnet: convolutional neural networks for matlab. In *Proceedings of the 23rd ACM international conference on Multimedia*, 2015 (cited on page 13).

[295] J. J. Verbeek and B. Triggs. Scene segmentation with crfs learned from partially labeled images. In *Conference on Neural Information Processing Systems*, 2008 (cited on page 88).

[296] V. Vineet, J. Warrell, and P. H. Torr. Filter-based mean-field inference for random fields with higher-order terms and product label-spaces. *International Journal of Computer Vision*, 2014 (cited on pages 30, 32, 33, 39).

[297] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition*, 2001 (cited on page 17).

[298] M. J. Wainwright, M. I. Jordan, et al. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 2008 (cited on page 20).

[299] S. Wang, R. Urtasun, M. Bai, G. Mattyus, H. Chu, et al. Torontocity: seeing the world with a million eyes. In *International Conference on Computer Vision*, 2017 (cited on page 3).

[300] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. In *Computer Vision and Pattern Recognition*, 2018 (cited on page 164).

[301] Y. Wei, J. Feng, X. Liang, M.-M. Cheng, Y. Zhao, et al. Object region mining with adversarial erasing: a simple classification to semantic segmentation approach. In *Computer Vision and Pattern Recognition*, 2017 (cited on pages 86–88, 91).

[302] Y. Wei, X. Liang, Y. Chen, X. Shen, M.-M. Cheng, et al. Stc: a simple to complex framework for weakly-supervised semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017 (cited on pages 87, 91).

[303] W. Wein, S. Brunke, A. Khamene, M. R. Callstrom, and N. Navab. Automatic ct-ultrasound registration for diagnostic imaging and image-guided intervention. *Medical Image Analysis*, 2008 (cited on page 2).

[304] D. Weiss and B. Taskar. Scalpel: segmentation cascades with localized priors and efficient learning. In *Computer Vision and Pattern Recognition*, 2013 (cited on pages 62, 63).

[305] J. Winn and J. Shotton. The layout consistent random field for recognizing and segmenting partially occluded objects. In *Computer Vision and Pattern Recognition*, 2006 (cited on page 57).

[306] C. Wojek and B. Schiele. A dynamic conditional random field model for joint labeling of object and scene classes. In *European Conference on Computer Vision*, 2008 (cited on pages 31, 60).

[307] J. Wu, T. Xue, J. J. Lim, Y. Tian, J. B. Tenenbaum, et al. Single image 3d interpreter network. In *European Conference on Computer Vision*, 2016 (cited on page 170).

[308] C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. Yuille. Mitigating adversarial effects through randomization. In *International Conference on Learning Representations*, 2018 (cited on pages 115, 118, 126, 130, 133, 167).

[309] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie, et al. Adversarial examples for semantic segmentation and object detection. In *International Conference on Computer Vision*, 2017 (cited on pages 117–119).

[310] J. Xie, M. Kiefel, M.-T. Sun, and A. Geiger. Semantic instance annotation of street scenes by 3d to 2d label transfer. In *Computer Vision and Pattern Recognition*, 2016 (cited on pages 5, 169).

[311] Y. Xiong, R. Liao, H. Zhao, R. Hu, M. Bai, et al. Upsnet: a unified panoptic segmentation network. In *Computer Vision and Pattern Recognition*, 2019 (cited on page 166).

[312] N. Xu, B. Price, S. Cohen, J. Yang, and T. Huang. Deep interactive object selection. In *Computer Vision and Pattern Recognition*, 2016 (cited on page 56).

[313] W. Xu, D. Evans, and Y. Qi. Feature squeezing: detecting adversarial examples in deep neural networks. In *arXiv preprint arXiv:1704.01155v1*, 2017 (cited on page 118).

[314] X. Xu, X. Chen, C. Liu, A. Rohrbach, T. Darell, et al. Can you fool ai with adversarial examples on a visual turing test? In *arXiv preprint arXiv:1709.08693v1*, 2017 (cited on page 117).

[315] C. Yang, L. Zhang, H. Lu, X. Ruan, and M.-H. Yang. Saliency detection via graph-based manifold ranking. In *Computer Vision and Pattern Recognition*, 2013 (cited on pages 87, 91).

[316] G. Yang, J. Pennington, V. Rao, J. Sohl-Dickstein, and S. S. Schoenholz. A mean field theory of batch normalization. In *International Conference on Learning Representations*, 2019 (cited on page 19).

[317] Y. Yang, S. Hallman, D. Ramanan, and C. C. Fowlkes. Layered object models for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012 (cited on pages 32, 57, 67, 95, 165).

[318] J. Yao, S. Fidler, and R. Urtasun. Describing the Scene as a Whole: Joint Object Detection, Scene Classification and Semantic Segmentation. In *Computer Vision and Pattern Recognition*, 2012 (cited on pages 22, 31, 32, 35, 60).

[319] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. In *International Conference on Learning Representations*, 2016 (cited on pages 16, 41, 43, 48, 114, 115, 119, 122, 139–142, 145, 146).

[320] Y. Yuan and J. Wang. Ocnet: object context network for scene parsing. In *arXiv preprint arXiv:1809.00916*, 2018 (cited on page 164).

[321] S. Zagoruyko, A. Lerer, T.-Y. Lin, P. O. Pinheiro, S. Gross, et al. A multipath network for object detection. In *British Machine Vision Conference*, 2016 (cited on page 56).

[322] M. D. Zeiler. Adadelta: an adaptive learning rate method. In *arXiv preprint arXiv:1212.5701*, 2012 (cited on page 14).

[323] O. Zendel, K. Honauer, M. Murschitz, D. Steininger, and G. Fernandez Dominguez. Wilddash-creating hazard-aware benchmarks. In *European Conference on Computer Vision*, 2018 (cited on pages 6, 168).

[324] J. Zhang, Z. Lin, J. Brandt, X. Shen, and S. Sclaroff. Top-down neural attention by excitation backprop. In *European Conference on Computer Vision*, 2016 (cited on page 91).

[325] Z. Zhang, S. Fidler, and R. Urtasun. Instance-level segmentation for autonomous driving with deep densely connected mrfs. In *Computer Vision and Pattern Recognition*, 2016 (cited on pages 58, 60, 64).

[326] Z. Zhang, A. G. Schwing, S. Fidler, and R. Urtasun. Monocular object instance segmentation and depth ordering with cnns. In *International Conference on Computer Vision*, 2015 (cited on pages 58, 60, 64, 86).

[327] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia. Icnet for real-time semantic segmentation on high-resolution images. In *arXiv preprint arXiv:1704.08545v1*, 2017 (cited on pages 114, 122, 140, 141, 145).

[328] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *Computer Vision and Pattern Recognition*, 2017 (cited on pages 70, 94, 98, 99, 108, 114, 115, 119, 121, 124, 125, 139, 140, 142, 145, 164).

[329] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, et al. Conditional random fields as recurrent neural networks. In *International Conference on Computer Vision*, 2015 (cited on pages 10, 30–34, 39–42, 44, 45, 47–50, 52, 58, 59, 64, 66, 93, 114, 115, 119, 120, 134–136, 139, 140, 157).

[330] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *Computer Vision and Pattern Recognition*, 2016 (cited on pages 90, 91).

[331] B. Zhou, H. Zhao, X. Puig, T. Xiao, S. Fidler, et al. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision*, 2019 (cited on page 7).