Thesis for the Degree of Doctor of Philosophy

Geometric Supervision and Deep Structured Models for Image Segmentation

Måns Larsson

CHALMERS

Department of Electrical Engineering
Chalmers University of Technology
Göteborg, Sweden 2020

Geometric Supervision and Deep Structured Models for Image Segmentation

Måns Larsson

ISBN 978-91-7905-294-2

Doktorsavhandlingar vid Chalmers tekniska högskola
Ny serie nr 4761
ISSN 0346-718X
Computer Vision and Medical Image Analysis group
Department of Electrical Engineering
Chalmers University of Technology
SE–412 96 Göteborg, Sweden

Cover:
3D model colored according to (from left): image values, semantic classes and semantic clusters. Related to work done in Paper I and II.

Typeset by the author using LaTeX.

Chalmers Digitaltryck
Göteborg, Sweden 2020

Geometric Supervision and Deep Structured Models for Image Segmentation

Måns Larsson
Department of Electrical Engineering
Chalmers University of Technology

Abstract

The task of semantic segmentation aims at understanding an image at a pixel level. Due to its applicability in many areas, such as autonomous vehicles, robotics and medical surgery assistance, semantic segmentation has become an essential task in image analysis. During the last few years a lot of progress have been made for image segmentation algorithms, mainly due to the introduction of deep learning methods, in particular the use of Convolutional Neural Networks (CNNs). CNNs are powerful for modeling complex connections between input and output data but have two drawbacks when it comes to semantic segmentation. Firstly, CNNs lack the ability to directly model dependent output structures, for instance, explicitly enforcing properties such as label smoothness and coherence. This drawback motivates the use of Conditional Random Fields (CRFs), applied as a post-processing step in semantic segmentation. Secondly, training CNNs requires large amounts of annotated data. For segmentation this amounts to dense, pixel-level, annotations that are very time-consuming to acquire.

This thesis summarizes the content of five papers addressing the two aforementioned drawbacks of CNNs. The first two papers present methods on how geometric 3D models can be used to improve segmentation models. The 3D models can be created with little human labour and can be used as a supervisory signal to improve the robustness of semantic segmentation and long-term visual localization methods.

The last three papers focuses on models combining CNNs and CRFs for semantic segmentation. The models consist of a CNN capable of learning complex image features coupled with a CRF capable of learning dependencies between output variables. Emphasis has been on creating models that are possible to train end-to-end, giving the CNN and the CRF a chance to learn how to interact and exploit complementary information to achieve better performance.

Keywords: Semantic segmentation, supervised learning, convolutional neural networks, conditional random fields, deep structured models, self-supervised learning, semi-supervised learning.

Acknowledgements

I would like to start off by thanking my supervisor Fredrik Kahl for introducing me to the field of Computer Vision as well as guiding me through my PhD while constantly having to convince me that what I'm doing is actually worth publishing. Your input, encouragement and ideas have been invaluable during this time. Another big thanks to Torsten Sattler, who has been my co-supervisor during the last two years of my PhD. Your never-ending ideas for experiments and inhuman pre-deadline work capacity have been crucial during this time.

I am also grateful for my current and former colleagues in the Computer Vision Group and the department of Electrical Engineering at Chalmers. Thank you Carl Toft, Erik Stenborg, Lars Hammarstrand, Olof Enqvist, Carl Olsson, Christopher Zach, Lucas Brynte, José Pedro Lopes Iglesias, Huu Le, Rasmus Kjær Høier, Georg Bökman, Eskil Jörgensen, Kunal Chelani, Mikaela Åhlén, Yuhang Zhang and Jesús Briales García for you good company, great coffee break discussions and the occasional after work. A special thanks to Jennifer Alvén for starting her PhD a few months before me and hence constantly having to guide me through mine, your help has been much appreciated. In addition I would like to extend my thanks to my collaborators at the Torr Vision Group in Oxford, especially Anurag Arnab and Shuai Zheng who have been involved in the development of the second part of this thesis and introduced me to the wonderful yet frightening world of large-scale deep learning experiments.

Lastly, I would like to thank my friends and family. My family, for continuing to support me in what I'm doing, even though they stopped understanding what it is a long time ago. My friends, for brightening my spare time and constantly reminding me that there are more important and enjoyable things than work. My wonderful girlfriend, Maria, for all her support and constant encouragement in everything I do. Thank you all!

Included Publications

Paper I M. Larsson, E. Stenborg, L. Hammarstrand, M. Pollefeys, T. Sattler, F. Kahl "A Cross-Season Correspondence Dataset for Robust Semantic Segmentation". *Conference on Computer Vision and Pattern Recognition (CVPR).* 2019.

Paper II M. Larsson, E. Stenborg, C. Toft, L. Hammarstrand, T. Sattler, F. Kahl "Fine-Grained Segmentation Networks: Self-Supervised Segmentation for Improved Long-Term Visual Localization". *International Conference on Computer Vision (ICCV).* 2019.

Paper III M. Larsson, A. Arnab, F. Kahl, S. Zheng, and P. Torr. "Revisiting Deep Structured Models in Semantic Segmentation with Gradient-Based Inference". *SIAM Journal on Imaging Sciences (SIIMS).* 2018. Extended version of paper (a)[1].

Paper IV M. Larsson, J. Alvén and F. Kahl. "Max-Margin Learning of Deep Structured Models for Semantic Segmentation". *Scandinavian Conference on Image Analysis (SCIA).* 2017.

Paper V M. Larsson, Y. Zhang and F. Kahl "Robust Abdominal Organ Segmentation Using Regional Convolutional Neural Networks". *Applied Soft Computing.* 2018. Extended version of paper (b)[1].

[1]Subsidiary publications can be found on the next page.

Subsidiary publications

(a) M. Larsson, A. Arnab, F. Kahl, S, Zheng, and P. Torr. "A Projected Gradient Descent Method for CRF Inference allowing End To End Training of Arbitrary Pairwise Potentials". *International Conference on Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR).* 2017.

(b) M. Larsson, Y. Zhang and F. Kahl "Robust Abdominal Organ Segmentation Using Regional Convolutional Neural Networks". *Scandinavian Conference on Image Analysis (SCIA).* 2017.

(c) A. Arnab, S. Zheng, S. Jayasumana, B. Romera-Paredes, M. Larsson, A. Kirillov, B. Savchynskyy, C. Rother, F. Kahl, P. Torr. Conditional Random Fields Meet Deep Neural Networks for Semantic Segmentation: Combining Probabilistic Graphical Models with Deep Learning for Structured Prediction". *Signal Processing Magazines Special Issue on: Deep Learning for Visual Understanding.* 2018.

Contents

Part I

Introductory Chapters

Chapter 1

Introduction

Understanding the content of an image is something that humans excel at. If I were to ask you to describe the objects present in an image you would in almost all cases manage that task effortlessly. However, if I ask you to state a set of rules to decide if an image contains a cat or a dog, you might have difficulties. Humans are so good at parsing and understanding visual scenes that we do not reflect on how we do it. Designing methods that do this automatically has however been proven to be a challenging problem and the field of Computer Vision is still very active.

Given an image, information can be extracted on different levels. This is illustrated in Figure 1.1 where a few examples of image analysis tasks of different detail are shown. The focus of this thesis is semantic segmentation, which aims at understanding an image on a pixel level. This means that we want to assign a label to each pixel, describing the object it is depicting. For example, going back to Figure 1.1, we have assigned the label "person" to the pixels colored pink and the label "dining table" to the pixels colored yellow.

Semantic segmentation has numerous of applications. In robotics, agents are usually required to extract useful information and understand their environment to perform tasks such as navigation and manipulation of objects. This is something that can be achieved with a camera and a semantic segmentation algorithm. Also, autonomous vehicles require a precise understanding of their surrounding to be able to make safe decisions in traffic. Semantic segmentation algorithms are also useful for numerous applications in medical research and clinical care, such as computer aided diagnosis and surgery assistance. Since many of the images handled in medical applications are three dimensional manual segmentation is time consuming. Having an automatic method will in these cases save medical personnel a lot of time and be very helpful for time-critical tasks such as surgery planning.

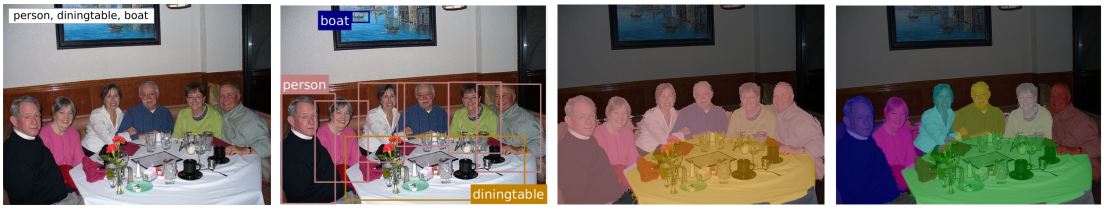

Figure 1.1: *Example of scene understanding tasks with increasing detail from left to right. From left: image captioning, object detection, semantic segmentation and instance segmentation. This thesis focuses on semantic segmentation. Image modified from [1].*

Traditionally, semantic segmentation algorithms have been approached by extracting some type of hand-crafted image features from the image. These features could be something as simple as a color gradient or a more complex function of the pixel values. A model relating these features to semantic classes is then created, or learnt from annotated examples, *i.e.* a set of images paired with their "true" semantic segmentations. During recent years most methods have moved from hand-crafted features to using Convolutional Neural Networks (CNNs), capable of learning complex features directly from image data.

The introduction of CNNs for semantic segmentation meant a large improvement in performance and we are now able to create models that are fairly good at understanding the content of an image (given that it is similar to the images it has been trained on). A drawback with a CNN is however that they cannot explicitly take the dependencies between output variables, *i.e.* how the label of one pixel depends on the label of the output pixels, into account. This can however be done using Conditional Random Fields (CRFs), which have been used extensively for semantic segmentation. Because of this, many methods combine a CNN and a CRF creating a Deep Structured Model (DSM) capable of learning complex image features while still taking output dependencies into account.

The parameters of these DSMs are usually learnt from data. This learning can be easily achieved by using traditional deep learning methods to train the CNN. Then, using the output of the CNN to form the CRF, learning the weight of the CRF. This approach, commonly referred to as piece-wise training, is suboptimal since the parameters of the CNN is learnt while ignoring output dependencies. A better approach is to train the CNN and CRF jointly, or end-to-end. This gives the CNN and the CRF a chance to learn how to interact to achieve better results. A sketch of piece-wise and end-to-end training of a DSM is shown in Figure 1.2.

DSMs and CNNs usually contains many learnable parameters, or weights, which require a lot of annotated data to train properly. For semantic segmenta-

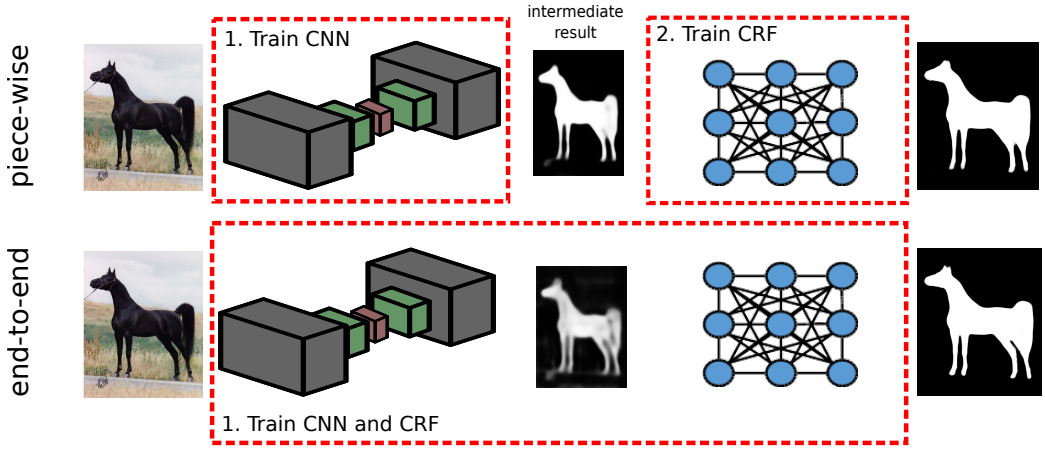

Figure 1.2: *Comparison of piece-wise and end-to-end training of a deep structured model (DSM). For the piece-wise training (above) the CNN is trained first and as a second step the parameters of the CRF are trained keeping the weights of the CNN fixed. During end-to-end training (below) the weights of the CNN and CRF are jointly trained, giving them a chance to learn how to interact to achieve better results.*

tion, annotating data is a tedious task meaning that large datasets are costly and time-consuming to acquire. Methods that alleviate the need for densely annotated data can usually be divided into one of the following categories: unsupervised, weakly-supervised or semi-supervised learning. As their respective names suggest unsupervised learning aims at training models without any manually annotated data, weakly-supervised learning with weaker annotations that are easier to obtain and semi-supervised learning where one part of the dataset is annotated and the other contains no labels.

1.1 Thesis Scope

The topic of this thesis is semantic segmentation in Computer Vision. It contains five papers that can be divided into two main parts. The first one being development of methods to utilize geometric supervision to improve segmentation methods and the second one being development of DSMs for semantic segmentation.

The first part consists of two papers, both of them revolving around the utilization of 3D models to improve segmentation models. These works contain tools from unsupervised, weakly-supervised and semi-supervised learning that are used

conjunction with geometric 3D models created from images. The geometric supervision is used to improve upon semantic segmentation methods or train CNNs that output fine-grained segmentation containing information useful for visual localization methods.

This second part consists of three papers, all of them presenting solutions to semantic segmentation problems. The applications have varied widely and different types of data have been considered, from 3D CT images to RGB images of horses to indoor scene understanding. The main focus has been on developing robust and accurate models to solve these problems. These models consist of a CNN capable of learning complex image features coupled with a CRF capable of learning dependencies between output variable, in our case pixel or voxel labels. Emphasis have been put on creating this type of models that also are possible to train end-to-end as well as the methods needed to enable this type of training.

1.2 Thesis Outline

The first part of this thesis consists of this introductory chapter, followed by Chapter 2 that provides background knowledge to the papers included in the thesis as well as placing them in an academic context. Chapter 3 summarizes the work and contributions of the thesis as well as each paper separately. A brief discussion of future work is given in Chapter 4. Finally, the included papers are appended in Part II.

Chapter 2

Background

The background chapter will start off with a brief introduction to the problem of semantic segmentation. Afterwards some background on Convolutional Neural Networks (CNNs) as well as Conditional Random Fields (CRFs) will be given. Following that, end-to-end training of Deep Structured Models (DSMs), *i.e.* a combination of a CNN and a CRF, will be discussed. Lastly, a section on training semantic segmentation algorithms without full supervision is included. The sections in this chapter are by no means exhaustive but aim at giving the reader enough background knowledge to understand the papers included as well as place them in an academic context.

2.1 Semantic Segmentation

Semantic segmentation, or scene labeling, is the process of assigning each pixel of an image to the semantic class that it is depicting. The semantic class should depend on the surrounding information, or context, of the pixel. That means that we want to understand what the image is containing on a pixel level. What classes we are interested in dividing the image pixel in depends on the task and what information about our surrounding we are interested in. Given a set of images from a camera mounted on the front of a car we might want to classify each pixel as being one of *e.g.* "driveable surface", "sidewalk" or "pedestrian" whereas given a medical CT image of the abdomen we might want to classify pixels into different organs, or perhaps "tumour" and "not tumour". An example of visualizations of semantic segmentations is shown in Figure 2.1.

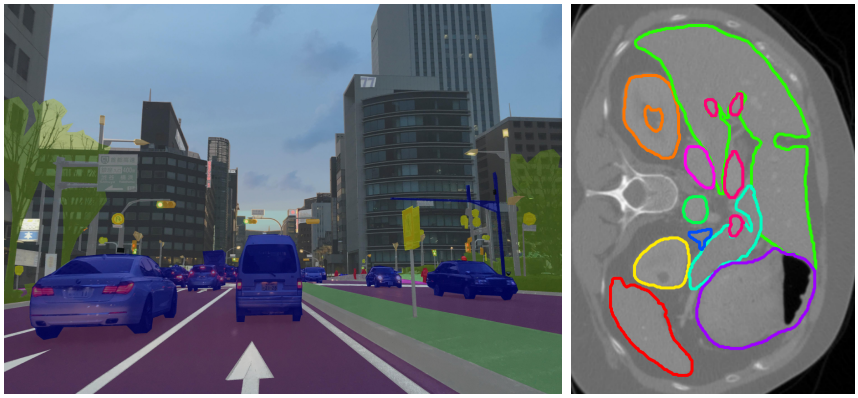

Figure 2.1: *Two examples of semantic segmentations. To the left is an image from the Mapillary Vistas dataset [2], a street-level image dataset with 66 semantic classes. The semantic class of each pixel is visualized by overlaying the original pixel with the class color. To the right is a slice of a CT image from the MICCAI 2015 challenge "Multi-Atlas Labeling Beyond the Cranial Vault" [3] for organ segmentation in the abdomen. Here the voxels of a class are visualized by delineating them with the class color. Note that only one slice of the original 3D CT volume is shown.*

2.1.1 Evaluation

Given an image paired with a semantic segmentation it is quite easy for a human to visually evaluate the segmentation as good or bad. It is however important to quantify how good a segmentation is, both to be able to quickly evaluate a method applied to a big set of images and also to be able to compare between different methods. A straightforward metric to use is the pixel accuracy which is defined as the ratio between correctly classified pixels and total number of pixels. However, for some datasets, the per-pixel accuracy can be quite misleading. Given, for example, an image with a lot of pixels labeled as "background". A segmentation method simply assigning the "background" label to all pixels will get a high pixel accuracy even though it obviously performs poorly.

An alternative metric is the commonly used Intersection over Union (IoU) or "Jaccard" index. Given the set of pixels A segmented as a class and the set of pixels B belonging to the same class according to the annotation the IoU is

$$\text{IoU} = \frac{|A \cap B|}{|A \cup B]}. \tag{2.1}$$

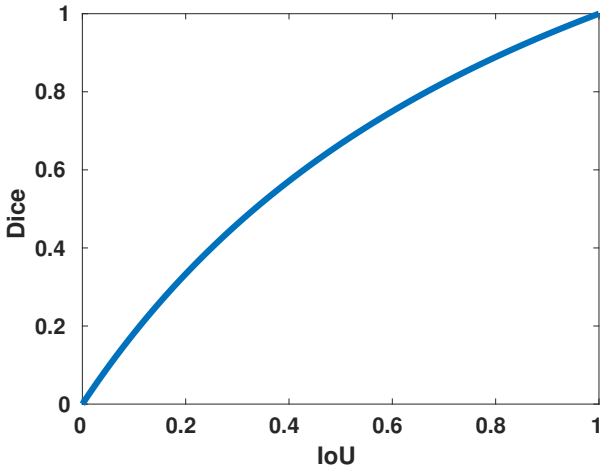

Figure 2.2: *The Dice coefficient plotted as a function of the intersection over union. The Dice coefficient and intersection over union are two commonly used measure to quantify segmentation results. Note that the Dice coefficient is higher than the intersection over union for any segmentation result.*

In terms of true/false positives/negatives we get

$$\text{IoU} = \frac{\#tp}{\#tp + \#fn + \#fp}, \tag{2.2}$$

where $\#tp$ denotes number of true positives, $\#fn$ denotes number of false negatives and so on. The IoU is a value between zero and one where a value of one means a perfect overlap of the segmentation and the ground truth while a value of zero means no overlap at all. For multi-label problems the mean IoU (mIoU) over all classes is usually measured as an overall performance indicator of a segmentation method.

For medical image segmentation tasks the Sørensen-Dice coefficient, or simply Dice coefficient, is a common metric. Keeping the notation above, the Dice coefficient is defined as

$$\text{Dice} = \frac{2|A \cap B|}{|A| + |B|}, \tag{2.3}$$

which in terms of true/false positives/negatives can be written as

$$\text{Dice} = \frac{2\#tp}{2\#tp + \#fn + \#fp}. \tag{2.4}$$

Similarly to the IoU, the Dice score can vary between zero and one. The relation between the Dice score and the IoU is Dice = 2 IoU/(1+IoU) which is visualized in

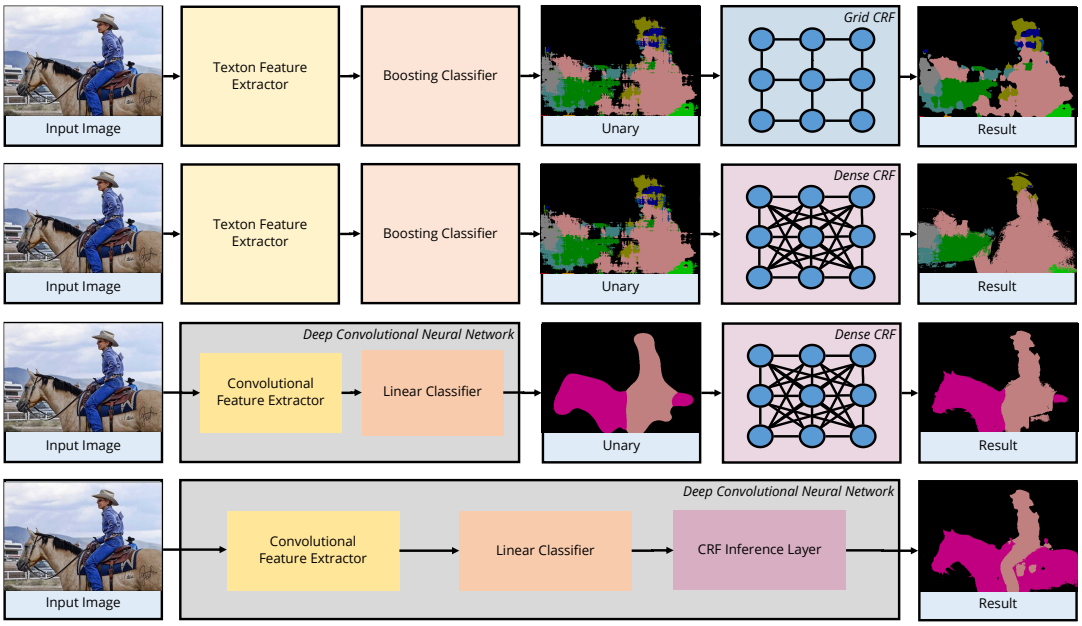

Figure 2.3: *Evolution of Semantic Segmentation systems. Initially, most approaches relied on hand-crafted image features and a fairly simple CRF model, this is represented in the first row showing the "Textonboost" work [4]. The second row uses a more sophisticated CRF model, DenseCRF, presented in [5]. Later on, most works have replaced the hand-crafted features with features learned from data with a Convolutional Neural Network. An early example of this is [6]. In [7], it was shown that the CRF inference could incorporated as a part of the CNN. This allowed simultaneous learning of the CNN and CRF weights, further improving results. This image is taken from [1] and result for this figure were obtained using the publicly available code of [6–10].*

Figure 2.2. As can be seen from the figure, the Dice coefficient always corresponds to a lower intersection over union.

2.1.2 Development of Approaches

Semantic segmentation methods date back to the 1970s [11,12]. Many of the early approaches tried to divide the image into semantic areas and then relate these areas to each other using a fixed rule-based system. It was in most cases hard to get these kind of rule-based or grammar-based methods to generalize well and performance was quite poor.

From the early 2000s up until now the popularity and performance of semantic segmentation methods have increased tremendously. The early methods utilized

powerful tools such as image descriptor and machine learning [13]. The majority of these methods are data driven and require manually annotated images to be able to train the models. The models can, once trained, be applied to unseen images and segment them into the semantic classes that were present in the manually annotated images. Many state-of-the-art methods [4, 5] used a CRF to be able to model interactions between the input images and output labels but also interactions between output labels. Given a CRF model most early approaches used the following pipeline

1. Extract features from the image, *e.g.* the RGB color of the pixel and its surrounding pixel or some more advanced features such as Textons [14] or SIFT [4].

2. Use extracted features and the annotated image to train an appearance model, *i.e.* a local classifier.

3. Use the output of the appearance model to form the unary term, *i.e.* the part of the CRF that models interactions between input and output.

4. Define, or learn from data, how the CRF should model interactions between output labels. Most commonly the type of interactions were pairwise, *i.e.* between the classes of two pixels.

5. Perform inference on the CRF model to segment an image.

This is of course a rough pipeline which many methods will not fit into. In addition, a lot of extensions and variants exists for each step of the pipeline. Regarding the first point of extracting features, most work has moved from carefully designed features to learning features from annotated data, usually with a CNN. This will be discussed thoroughly in Chapter 2.2. Also, several works have done data driven approaches to learn the pairwise interactions described by the CRF. In addition, several different types of CRF models have been proposed. A notable example is the DenseCrf presented in [5] where every pair of pixels is connected by a pairwise term in the CRF. Finally, during the last few years methods that learn the parameters of the CRF as well as the weights of the feature extracting CNN jointly have appeared. Two of them are the papers included in this thesis but more examples exist [7, 15, 16]. Figure 2.3 provides a summary of the development of semantic segmentation systems.

2.2 Learning Features

As mentioned in Section 2.1.2, most methods for semantic segmentation currently use image features learnt from annotated data. The dominating approach is to apply a CNN to learn these features and inspecting most popular semantic segmentation benchmarks, top entries on the leaderboard all use a CNN. In this section an introduction to Artificial Neural Networks (ANNs) and CNNs is given.

The idea behind ANNs and CNNs is not new. Already in the 1960s the biologically inspired Perceptron was introduced [17] which resembles the commonly used ANNs of today. Also the idea of introducing spatial invariance in ANNs were presented already in 1980, when K. Fukushima *et al.* introduced the "Neocognitron" [18]. During the 1980s and 1990s there was some progression in the field of neural networks but it was not until 2012 that these types of methods got their breakthrough. In 2012 Krizhevsky *et al.* [19] presented "Alexnet", a CNN for classifying images of the ImageNet [20] dataset that achieved considerably better than the previous state-of-the-art. Since then, approaches using CNNs have become dominant in most detection and classification problems [21]. For the task of semantic segmentation a defining paper was J Longs *et al.* "Fully convolutional networks for semantic segmentation" [10] which introduced a method of transforming CNNs previously used for classification to efficiently segment an image. These types of "Fully Convolutional" networks has since then been standard for semantic segmentation.

2.2.1 Multilayer Neural Networks

The most common ANNs have a "feed-forward" neural network architecture. In feed-forward neural networks computations are done layer-wise, and the values of the data at one layer of the network depend only on computations in previous layers. In one layer of the network, the input to the layers is multiplied by a weight vector $\boldsymbol{W_i}$ and a bias vector is added $\boldsymbol{b_i}$ according to

$$\boldsymbol{g}_i = \boldsymbol{W}_i \boldsymbol{h}_{i-1} + \boldsymbol{b}_i, \tag{2.5}$$

where $\boldsymbol{h}_{i-1}$ is the output of the previous layer (or the input data if i is the first layer). The vectors $\boldsymbol{h}_i$ are commonly referred to as hidden units (except for the inputs $\boldsymbol{h}_0$ and the output $\boldsymbol{h}_L$), or neurons, and their size depends on the size of the weight matrices $\boldsymbol{W}_i$. A weight matrix with less rows than columns will decrease the size of the hidden units vector. After this computation the output $\boldsymbol{g}_i$ is passed through a non-linear activation function

$$\boldsymbol{h}_i = \sigma(\boldsymbol{g}_i). \tag{2.6}$$

Computation layers are stacked on top of each other and the output of the last layer L is the output of the neural network $\boldsymbol{y} = \boldsymbol{h}_L$. In modern literature, these types of neural network layers are often referred to as fully connected layers.

2.2.2 Activation Functions

Activation functions are a crucial part of the neural network. If activation functions were to be omitted the computations of the entire network would consist of only linear functions and could be replaced with an equivalent single matrix multiplication. In contrast, with non-linear activation functions, it has been shown that a feed-forward neural network is a universal function approximator [22]. This means that, in theory, they can learn any function.

In the early days of neural networks, smooth non-linear activation functions were commonly used. Two examples of these are the sigmoid, defined as

$$\sigma(x) = \frac{1}{1 + e^{-x}}, \tag{2.7}$$

and the hyperbolicus tangent function $\sigma(x) = \tanh x$. These functions are quite similar but differs in range, the output of a sigmoid lie within $]0, 1[$ while the output of $\tanh(x)$ lie within $]-1, 1[$. The rectified linear unit (ReLU) [23], which is defined as $\sigma(x) = \max(0, x)$, is a commonly used activation function. Including ReLU activation functions in a neural network generally makes the training faster as well as allowing training of networks with more layers [21].

The activation function of the final layer is usually chosen differently to the intermediate activation functions. The choice usually depends on what task we are training the network to solve. For example, if we want to use the network to solve a regression problem we might not use any final activation function at all, allowing for unbounded output values of the network. If we instead are interested in image classification we could use a softmax activation function defined as

$$\sigma(x)_j = \frac{e^{x_j}}{\sum_{k=1}^{C} e^{x_k}} \quad \text{for} \;\; j = 1, ..., C. \tag{2.8}$$

The softmax function outputs a set of values all between zero and one and which sum to one. The value of $\sigma(x)_j$ can hence be used as an estimation of the probability of the current input belonging to class j.

2.2.3 Convolutional Neural Networks

Convolutional Neural Networks [24] are designed to process data that has an inherent grid-like structure, such as 2D RGB images, 3D videos or medical images such as CT scans. Since many of these data types have a lot of input parameters, an RGB image of standard size can for example be represented with $512 \times 512 \times 3 = 786432$ values, the weight matrix $\boldsymbol{W}$ of a fully connected layer would become very large. This would make the computations very demanding while also giving the neural network an extremely large amount of weights, something that might cause overfitting.

CNNs circumvent this problem by using a biologically inspired spatial weight sharing scheme [25]. Instead of learning full weight matrices for each layer a CNN learns a bank of filters for each convolutional layer. The intermediate values between layers are referred to as feature maps and keep their spatial grid-like structure throughout the network. Learning filters, instead of full weight matrices, means that the same weight values will be applied at every spatial position for each layer, greatly reducing the number of weights needed to be learnt. This would be the equivalent of restricting the weight matrix of equation (2.5) to be a Toeplitz matrix. In addition, the convolutional layers are spatially invariant, meaning that input patterns found in different parts of an image will be processed similarly regardless of spatial position.

Another common component of a CNN are pooling layers. A pooling layer applies a rectangular window to each feature map forwarding for example the maximum number present in the window (for max-pooling layers). Pooling layers introduces an invariance to small shifts in input data while also reducing the spatial size of the feature maps, controlling the capacity of the neural network [21]. Adding pooling layers also enlarges the receptive field of higher level features. The receptive field of a feature is the part of the input image that might influence the value of the feature, a larger receptive field enables learning of more complex features. A common approach to building an CNN for image classification is to stack a couple of convolutional and pooling layers, adding activation functions (typically ReLU) after the convolutional layers. This enables the CNN to learn more complex and high-level features for each stacked layer. Ideally the first few layers learn to extract low level image features such as edges, lines and blobs while later layers extracts complex features such as faces, legs or wheels. Finally one or several fully connected layer can be applied to transform the features from spatially structured maps to for example a vector of estimated class probabilities.

Convolutional Layers

As mentioned, the convolutional layers of a CNN each learn a bank of filters. Given an input feature map $\boldsymbol{X}$ of size $W^{in} \times H^{in} \times F^{in}$, where W is the width, H the height and F the number of feature maps. A trained bank of filters is applied according to

$$\boldsymbol{Y}_j = \boldsymbol{B}_j + \sum_i \boldsymbol{W}_{ij} * \boldsymbol{X}_i, \tag{2.9}$$

where $\boldsymbol{X}_i$ denotes the i-th feature map of $\boldsymbol{X}$ and $\boldsymbol{W}_{ij}$ denotes the values of the learnt filters. The output feature map $\boldsymbol{Y}$ has a size of $W^{out} \times H^{out} \times F^{out}$, padding can be used to keep the same width and height as the input feature map. The filter bank has a size of $K_1 \times K_2 \times F^{in} \times F^{out}$, where $K_1 \times K_2$ is the size of each filter. For each output, optionally, a bias weight is learnt. $\boldsymbol{B}_j$ denotes this bias resized to the width and height of the output feature map. The size of the filters differs from application to application but a size of 3×3 is most commonly used [26–28].

A variant of the convolutional layer designed to provide a greater increase of the receptive field (the region of the input that affects a particular unit of the network) between subsequent layers is the *à-trous* or dilated convolutions [29]. These convolutions uses a set of upsampled filters where only weights at every l-th index is non-zero. Here, l is usually referred to as the dilation factor, note that dilated convolution with $l = 1$ is just standard convolution.

Pooling Layers

Pooling layers perform down-sampling of the image features [30]. Several types of pooling layers exist, the most common ones are max pooling and average pooling. These layers applies a fixed size window to the input feature map in strides. It then outputs the max (or average) of the values in this window. Choosing a stride equal to the window size results in non-overlapping regions that forwards information to the next feature map. Choosing a window size of 2×2 and a stride of 2 would result in a down-sampling of the spatial size of the feature map by a factor of 2.

2.2.4 Learning

Once the architecture of our CNN is set we can view it as a function approximator $f(\boldsymbol{x}, \boldsymbol{\theta})$, where $\boldsymbol{x}$ is the input and $\boldsymbol{\theta}$ the learnable weights of all layers. This section will give a brief introduction to the most important parts needed for the learning process. Note that this is only applicable for supervised learning, where an annotated dataset is available.

Loss Function

A loss function is a way to quantify how well the CNN is performing, measuring the compatibility of the CNNs output, or prediction, to the ground truth label. The loss is generally defined for one sample of the dataset, and during learning the weights of the CNN, $\boldsymbol{\theta}$ are adjusted to minimize the mean of the losses

$$L(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\theta}) = \frac{1}{N} \sum_i l(y_i, f(\boldsymbol{x}_i, \boldsymbol{\theta})). \tag{2.10}$$

Here $\boldsymbol{X}, \boldsymbol{Y}$ are the set of input and labels of a given dataset with N samples and $\boldsymbol{x}_i, y_i$ denotes the data/label pair of one sample.

A commonly used loss function for classification tasks is the cross-entropy loss. For a CNN with a softmax activation function as a last layer, outputting an estimate of the probabilities of the input $\boldsymbol{x}_i$ belonging to each class, the cross-entropy loss may be defined as

$$l(y_i, f(\boldsymbol{x}_i, \boldsymbol{\theta})) = -\log(f_{y_i}(\boldsymbol{x}_i, \boldsymbol{\theta})). \tag{2.11}$$

Here, $f_{y_i}(\boldsymbol{x}_i, \boldsymbol{\theta})$ is the estimate of the probability from the CNN that the input belongs to the ground truth class y_i. The name cross-entropy loss comes from the fact that this loss minimizes the cross entropy between the distribution of the ground truth labels and the label distribution generated by the CNN, given that the samples are independent and identically distibuted random variables.

Loss Minimization

As previously mentioned, the learning is achieved by minimizing a defined loss function over the given dataset. Since there generally is not any closed-form solution to the learning problem, $\boldsymbol{\theta}^* = \arg\max(L(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\theta}))$, local optimization methods are often used. Commonly, a variant of gradient descent is used which updates the parameters of the CNN according to

$$\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i - \eta \nabla_{\boldsymbol{\theta}} L(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\theta}_i), \tag{2.12}$$

where η is the step size or learning rate. For large datasets this is however inefficient and a stochastic approximation of the gradient can be used instead. This is called mini-batch gradient descent and is defined as

$$\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i - \eta \sum_{i \in \mathcal{B}} \nabla_{\boldsymbol{\theta}} l(y_i, f(\boldsymbol{x}_i, \boldsymbol{\theta}_i)), \tag{2.13}$$

where $\mathcal{B}$ is the batch which in turn is a subset of the complete dataset. Using a mini-batch of size one is referred to as stochastic gradient descent. There are

several variants of this update rule, designed to reduce noise of the estimated gradient and accelerate convergence. Some examples are gradient descent with momentum [31], with Nesterov momentum [32], AdaGrad [33] and Adam [34]. All of these are first-order methods which means that they only require the calculation of the gradient with regards to the weights.

The gradient of the loss function with respect to all the weights of the network can be efficiently computed using the back-propagation algorithm [31] – a practical application of the chain rule for derivatives. Given the loss derivative $\frac{\partial L}{\partial y}$ with respect to the output of a simple layer described by $y = f(x, \theta)$, where x is the input, y the output and θ the weights. The loss derivative with respect to the input can calculated by simply applying the chain rule $\frac{\partial L}{\partial x} = \frac{\partial L}{\partial y}\frac{\partial f}{\partial x}$, similarly for the weights we get $\frac{\partial L}{\partial \theta} = \frac{\partial L}{\partial y}\frac{\partial f}{\partial \theta}$. Using this back-propagation we can start from the final layer of the network and calculate the loss with respect to the input of each layer, propagating the loss gradient all through the network until we have calculated it with respect to every weight of the network.

Since the learning problem is non-convex and almost all methods are based on local optimization it is a possibility for the learning to get stuck in a poor local minimum. In practice, this is generally not a big problem, even for different initial conditions many networks reach a solution of very similar quality [21]. Recent work points towards the existence of a lot of saddle points in the loss surface that the learning algorithm might get stuck in [35, 36]. However, all of them have very similar and low values of the loss function and hence give a good enough solution.

Regularization

Overfitting is a term used for when training a model makes it perform well on the training data but poor on unseen input data. Since a typical CNN contains a very large number of free parameters it is prone to overfitting. A large enough CNN could "memorize" the training data instead of learning good rules that generalize to unseen data. Because of this, several regularization methods meant to prevent the CNN from overfitting have been developed.

Ideally, we would like to just add training data until the CNN is incapable of overfitting. Annotating new data is however a timely process which we generally want to avoid. An alternative to adding new training data is to perform data augmentation on the already available data. This means changing the samples of the data slightly in an randomized way during training. Common augmentation operations used for image tasks are, random cropping of the image, random rotation or simply adding noise to the pixel values.

Another fairly simple regularization technique is weight decay. This means adding an extra term to the loss function that penalize large values of the parameters of the CNN. For the case of L^2 weight decay the term $\lambda\frac{1}{2}\sum_i \theta_i^2$, where λ is the weight decay strength, is added to the loss function in (2.10). For L^1 regularization we instead add the term $\lambda\sum_i |\theta_i|$, favoring sparse solutions.

Two additional, very popular, regularization techniques are Dropout [37] and Batch Normalization [38]. These are added as separate layers to the CNN and have different functionalities during learning and during inference. Dropout works by only keeping the values of each neuron non-zero with given probability p, the others are set to zero. During inference, all neuron values are kept but scaled with a factor p. Batch Normalization shifts the values of the features to have a specific mean and variance for each mini-batch during training. In addition to avoiding overfitting to some extent, this also allows the use of a higher learning rate during training [38].

CNNs for Semantic Segmentation

The task of annotating data is considerably harder and more time-consuming for semantic segmentation where each pixel need to be annotated, compared to image classification where only one label per image is needed. This restricts the size of available datasets for semantic segmentation and there are no available datasets of the same size as for example Imagenet [39], which contains millions of annotated images. Due to this, many popular CNNs for semantic segmentation have a classification counterpart that has been trained on the million images of Imagenet. The architecture of the classification CNN is then changed to enable dense output maps, transforming it to a segmentation CNN. The weights of the first few layers are however kept, with the motivation that these layers have learnt to extract meaningful image features during the extensive classification training. This approach have shown to be preferable to training from scratch for many segmentation tasks, even for images fairly different from the Imagenet data [40–42].

Repurposing a classification network for semantic segmentation is not entirely straight-forward. As mentioned in Section 2.2, a defining paper for this was "Fully convolutional networks for semantic segmentation" by J Long *et al.* [10] where they presented segmentation version of several classification network that were state-of-the-art on Imagenet at that time. The fully connected layers of these networks were transformed to convolutional layers with filter size 1×1, which changes the previous classification scores to spatial feature maps. These feature maps, together with feature maps earlier in the CNN were upsampled using deconvolution layers [43] and merged providing dense output predictions for images of arbitrary size. These

CNNs can then be trained for segmentation end-to-end using a pixel-wise version of the cross-entropy loss presented in Section 2.2.4.

Despite the success of fully convolutional networks of [10] this architecture has several drawbacks. Pooling layers are great for image classification, enabling the CNN to learn complex high-level image features. It is however not ideal since performing pooling operations implies loss of spatial information on where the image features came from in the image. Some works have tried to get rid of the pooling layers entirely [44] and other types of layers have been introduced in an effort to keep spatial information while still achieving large receptive fields. An example of this is the dilated convolutions mentioned in Section 2.2.3.

Many recent works considering CNNs for semantic segmentation try to design networks that are able to learn high-level image features while not losing spatial information. Some examples include encoder-decoder networks such as Segnet [45] and U-Net [46] as well as PSP-Net [47] and DeepLabv3+ [48] that processes features on different resolution in separate paths.

2.3 Learning Structure

As mentioned in Section 2.2.3, CNNs are good at modeling complex relations between input data and output data. They cannot however explicitly take dependencies between output variables into account. In addition, they are often trained with a pixel-wise loss function, disregarding the fact that the output data is actually structured.

A way of taking output structure into account while also allowing for explicit modeling of dependencies between output variables is using Probabilistic Graphical Models (PGMs). A PGM models a probability distribution over a set of random variables whose structure is defined via a graph. In this thesis we will focus on Conditional Random Fields (CRFs) that are commonly used for semantic segmentation.

2.3.1 Conditional Random Fields

Conditional Random Fields (CRFs) models the conditional probability, $P(\mathcal{Y}|\boldsymbol{x})$ of a given output set $\mathcal{Y} = \{Y_1, ..., Y_N\}$ and input $\boldsymbol{x}$. Working with images, $\boldsymbol{x}$ denotes the image values and we generally associate one input and one output variable with each pixel. For semantic segmentation each output Y_i is assigned a value from a finite set of possible states $\mathcal{L} = \{l_1, l_2, ..., l_L\}$, where each state represent a class label. The dependencies between output variables are described by an undirected graph whose vertices are the random variables $\{Y_1, ..., Y_N\}$. The

conditional probability for the CRF can be written as

$$P(\mathcal{Y} = \boldsymbol{y}|\boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x})} \exp(-E(\boldsymbol{y}, \boldsymbol{x}; \boldsymbol{w})), \tag{2.14}$$

where $E(\boldsymbol{y}, \boldsymbol{x}; \boldsymbol{w})$ denotes the Gibbs energy function with respect to the assignment of labels to the output variables $\boldsymbol{y} \in \mathcal{L}^N$. The parameters, $\boldsymbol{w}$, of the CRF can either be hand-crafted from prior knowledge or learnt from data. $Z(\boldsymbol{x})$ is the partition function given by

$$Z(\boldsymbol{x}) = \sum_{\boldsymbol{y} \in \mathcal{L}^N} \exp(-E(\boldsymbol{y}, \boldsymbol{x}; \boldsymbol{w})). \tag{2.15}$$

It is hence a normalization constant making the conditional probabilities sum to one. Note that the sum is over all possible combination of label assignments available, it is therefore computationally expensive to evaluate the value of the partition function.

For most image applications the Gibbs energy function decomposes over unary and pairwise terms, *i.e.* terms depending on only one and two variable respectively. The energy can be written as

$$E(\boldsymbol{y}, \boldsymbol{x}; \boldsymbol{w}) = \sum_{u \in \mathcal{V}} \psi_u(y_u, \boldsymbol{x}; \boldsymbol{w}) + \sum_{(u,v) \in \mathcal{E}} \psi_{uv}(y_u, y_v, \boldsymbol{x}; \boldsymbol{w}). \tag{2.16}$$

The terms $\psi_u(y_u, \boldsymbol{x}; \boldsymbol{w})$ and $\psi_{uv}(y_u, y_v, \boldsymbol{x}; \boldsymbol{w})$ are commonly referred to as unary and pairwise potentials respectively. Note that the graph structure defines what terms are present in this energy. For $\psi_{uv}(y_u, y_v, \boldsymbol{x}; \boldsymbol{w})$ to be non-zero node u and v must share an edge.

Potential Types

The unary potentials, also known as the data cost, of the CRF energy are often obtained from a pixelwise classifier estimating the class probabilities of each pixel. Commonly the term for each pixel is set to

$$\psi_u(y_u = l_p) = -w_1 \log(P(y_u = l_p|\boldsymbol{x})), \tag{2.17}$$

where $P(y_u = l_p|\boldsymbol{x})$ is an estimate of the probability of pixel u belonging to class l_p.

For the pairwise potential the connectivity or structure of our graph needs to be defined. This specifies what pairwise terms should be included in the energy, and also which output variables should depend on each other. A simple and commonly

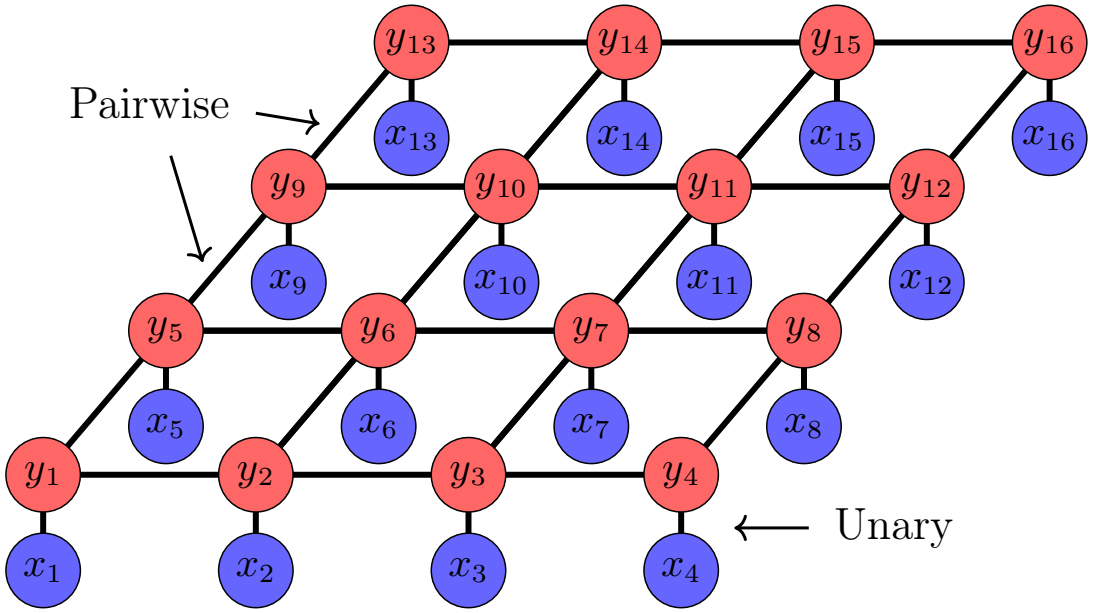

Figure 2.4: *CRF with a simple nearest neighbour connectivity, neighbourhood size four. The variables y_u are assigned class labels while the variables x_u represents the pixel values.*

used structure is the nearest neighbour connectivity where pixels are connected through an edge to its neighbours only. The size of the neighbourhood might vary but for 2D images a size of four or eight is common. An example of this structure can be seen in Figure 2.4.

The pairwise potentials, $\psi_{uv}(y_u = l_p, y_v = l_q, \boldsymbol{x}; \boldsymbol{w})$, defines the cost of assigning label l_p to pixel u and label l_q to pixel v. It can hence be used to enforce consistency and structure in the output. As an example, for semantic segmentation, we generally want neighbouring pixels to have the same labels. A type of pairwise term that enforces this is the Potts model given by

$$\psi_{uv}(y_u = l_p, y_v = l_q, \boldsymbol{x}; \boldsymbol{w}) = w_2 \mathbb{1}_{l_p \neq l_q}, \tag{2.18}$$

where $\mathbb{1}_{l_p \neq l_q}$ denotes the indicator function equaling one if $l_p \neq l_q$ and zero otherwise. This pairwise term can be generalized in several ways, for example we might want to weight the cost of assigning different labels to neighbouring pixel differently depending on if they have similar color or not. This can be achieved by adding a weighting term according to

$$\psi_{uv}(y_u = l_p, y_v = l_q, \boldsymbol{x}; \boldsymbol{w}) = w_3 \mathbb{1}_{l_p \neq l_q} e^{-(x_u - x_v)^2}, \tag{2.19}$$

where x_u and x_v are the pixel values of pixel u and v. This type of pairwise terms adds a lower energy if two neighbouring pixels differ a lot in color.

Both of these pairwise terms are constructed using prior knowledge, such that neighbouring pixel often have the same label unless there is a change in contrast. This is of course not true in all cases and several works have instead tried to learn the pairwise term from data [49, 50]. In Paper III we present a CRF model with more general pairwise potentials that can be learnt from data.

Using a neighbourhood only consisting of neighbouring pixels limits the extent on how far across the image information can propagate. A natural way to increase this limit is to increase the size of the neighbourhood, for example connecting all pixels closer than d pixels apart. The extreme of this would be to connect all pairs of pixels which is done for the denseCrf model. The denseCrf model were popularized by [5], that presented a method to perform efficient inference for these types of CRFs. The pairwise terms for dense CRFs also include a weighting on the distance between two pixels, hence the strength of the pairwise term decays exponentially with the distance between the pixels.

It is also possible to include potentials that depend on more than two pixel labels, *i.e.* higher order potentials. Higher order potentials can for example be used to enforce consistency within superpixels or utilize object detection results for semantic segmentation [51–54].

Inference

The inference problem, giving a semantic segmentation, equates to finding the maximum a posteriori labeling of the model in equation 2.14. Finding the minimizer to the Gibbs energy,

$$\boldsymbol{y}^* = \arg\min_{\boldsymbol{y}} E(\boldsymbol{y}, \boldsymbol{x}; \boldsymbol{w}), \tag{2.20}$$

is an equivalent problem. This problem is in general NP-hard [55], typical approaches to solving it can hence be divided into two categories, exact algorithms that only apply to special cases of the energy and approximate solutions. We will provide a few examples here but for an extensive overview of approaches we refer to [56, 57].

If we deal with a binary segmentation problem, *i.e.* only are interested in two classes, and if the energy is submodular the globally optimal solution can be found using the graph cuts method [57]. This approach can be extended to multi-label problems using the α-expansion [58], however we lose the guarantee of finding the global optimum.

Several popular methods are based on a relaxation of the original problem, these are usually the most efficient ones for performing inference in denser CRFs. One example is the mean-field method where the original distribution is $P(\boldsymbol{y}|\boldsymbol{x})$ is

approximated with a fully factorized one $Q(\boldsymbol{y})$. The optimization is then done by minimizing the Kullback–Leibler divergence between the two distributions. Other approaches rely on a continuous relaxation of the Gibbs energy, and then using local search methods to find a local minimum of the energy. This type of methods have been shown to outperform mean-field on several tasks [59] and is the approach used in Paper III.

Parameter Learning

The learning problem consists of estimating the parameters of the CRF, $\boldsymbol{w}$, based on a training set $(\boldsymbol{y}^{(k)}, \boldsymbol{x}^{(k)})_{k=1}^{N}$. The goal with the training is that if inference is performed for an input image from the data set, we want a solution close to the ground truth labeling. An intuitive approach to the learning problem is based on the maximum likelihood principle, *i.e.* finding the set of parameters that maximizes the probability of the training set.

A major difficulty when performing maximum likelihood training for CRFs is that it requires computation of the partition function for each training instance and for each iteration of a numerical optimization algorithm. This is of course computationally expensive and makes learning infeasible for CRF models used for semantic segmentation. Most popular learning methods therefore make approximations that simplify the computation of the partition function. Mean field is an example of this where the fully factorized distribution simplifies computation of the partition function [5]. Piece-wise training is also an option which only requires computation of local normalization factor over fewer variables [60,61]. Other methods instead try to estimate the partition function using sampling [62].

Another approach is to use a learning method that avoids the computation of the partition function, for example learning a model that maximizes the margin between the energy of the ground truth and any other output configuration [63,64]. This can be formulated as

$$\begin{aligned} &\max_{\boldsymbol{w}} \quad \zeta \\ s.t. \quad &E(\boldsymbol{y}, \boldsymbol{x}^{(k)}; \boldsymbol{w}) - E(\boldsymbol{y}^{(k)}, \boldsymbol{x}^{(k)}; \boldsymbol{w}) \geq \zeta \quad \forall\, k \text{ and } \boldsymbol{y} \neq \boldsymbol{y}^{(k)}. \end{aligned} \tag{2.21}$$

Since there is an exponential amount of constraints in this optimization problem it is not feasible to solve it as is. A solution to this is to iteratively add the constraints that currently is furthest away from being satisfied [65]. This learning method is utilized in Paper IV.

2.4 End-to-End Learning

Combining CNNs and CRFs is a powerful approach for dense classification tasks such as semantic segmentation. The CNNs ability to learn complex high-level image features paired with the CRFs ability to model output dependencies generally yields impressive results. However, many existing approaches use a two step training process to learn the weights of the CNN and CRF. Firstly, the CNN is trained to perform pixel-wise segmentation on the available data set. Secondly, the CRF is trained keeping the unary potentials fixed (although based on the output of the CNN). This is often referred to as piece-wise training and is non-ideal since the CNN is learnt while ignoring dependencies between output variables.

Instead, a better solution would be to perform end-to-end training. This means jointly training the CNN and the CRF at the same time. In this way the CNN and the CRF get the chance to learn how to interact and exploit complementary information to achieve as good of a result as possible. During recent years several examples of these deep structured model trained end-to-end have been proposed in the literature [7, 50, 66–68]. This section aims at providing a brief introduction to some of these methods, for more details we refer to [1].

2.4.1 CRF Inference as a Neural Network Layer

Given an iterative CRF inference method only consisting of differentiable operations, these operations can be implemented as neural network layers. Each step in the inference routine equaling one forward pass of a network layer. By implementing the back-propagation routines for this layer, which amounts to applying the chain rule for derivative, the error derivatives with respect to the parameters of the CRF can be computed during training. In addition, the error derivative with respect to the output of the CNN can be computed and the error can be propagated all the way back through the CNN. This enables the parameters of both the CRF and the CNN to be updated simultaneously during learning. This is usually referred to as unrolling inference algorithms and was shown to be possible for the mean-field inference algorithm [7]. In Paper III we show that this is possible for gradient-based CRF inference as well.

2.4.2 Back-propagating CRF Learning Objective

Many of the approaches for CRF parameter learning presented in Section 2.3.1 can be abstracted to minimizing a global objective L. This global objective depends on the samples of the data set, the parameters of the CRF as well as the output of the CNN, denoted $\boldsymbol{z}$, used to create the CRF potentials. If we are able to calculate the gradient of this global objective with respect to the CNN output, $\nabla_{\boldsymbol{z}} L$, we can back-propagate this gradient back through the CNN to calculate $\nabla_{\boldsymbol{\theta}} L$, where $\boldsymbol{\theta}$ are the weights of the CNN. The weights can then be updated using local search methods. The same thing is possible for the weights of the CRF, if $\nabla_{\boldsymbol{w}} L$ is calculated.

This approach of learning is usually formulated as a bi-level optimization problem [69–71] on the following form

$$\min_{\boldsymbol{\theta}} \sum_{i=k}^{N} l(\boldsymbol{y}^{(k)}, \boldsymbol{y}_k^*), \tag{2.22}$$

$$\text{subject to} \qquad \boldsymbol{y}_k^* = \arg\min_{\boldsymbol{y} \in C} E(\boldsymbol{y}, \boldsymbol{z}(\boldsymbol{\theta}), \boldsymbol{x}^{(k)}, \boldsymbol{w}) \qquad \forall\, k. \tag{2.23}$$

Here, C is the constraint set, E the CRF energy and $(\boldsymbol{y}^{(k)}, \boldsymbol{x}^{(k)})_{k=1}^{N}$ the training set. Since the optimal solution $\boldsymbol{y}_k^*$ of the inner optimization problem will depend on the output of the network, $\boldsymbol{z}$, and the weights of the CRF, $\boldsymbol{w}$, the gradients $\nabla_{\boldsymbol{z}} L$ and $\nabla_{\boldsymbol{w}} L$ can be calculated. In short, for the unconstrained case, this can be done by applying the implicit function theorem on the first order optimal conditions of the energy function ($\nabla_{\boldsymbol{y}} E = \boldsymbol{0}$), and using the fact that $\boldsymbol{y}$ is a function of $\boldsymbol{z}$. This enables the calculation of $\nabla_{\boldsymbol{z}} \boldsymbol{y}_k^*$ by solving a linear system consisting of second order derivatives of E. Having $\nabla_{\boldsymbol{z}} \boldsymbol{y}_k^*$ enables back-propagation through the energy minimization, in this case CRF inference, and hence end-to-end learning.

For more details, as well as the constrained cases, we refer to [70]. In Paper IV we present a method for doing this utilizing the max margin training approach for CRFs introduced in Section 2.3.1. Other examples of methods in this category are [50, 62, 66].

2.5 Learning Without Full Supervision

The learning approaches presented in previous sections all have one thing in common: they require an annotated dataset. Annotating data for semantic segmentation is a tedious task, creating one high-level annotated image can take around 90 minutes [72]. For 3D medical images, annotations are even more time-consuming and costly to acquire since there are a lot more pixels (or voxels) to be annotated and medical expertise is needed. There are several ways of getting around this problem, training neural networks without full supervision. In this section, three prominent categories of these methods will be briefly discussed, namely unsupervised learning, semi-supervised learning and weakly-supervised learning. Note that this is not an exhaustive text on this topic but serves as background to Paper I and II where tools from all three of these categories were utilized.

2.5.1 Unsupervised Learning

Unsupervised learning aims at extracting useful representations and patterns from unlabeled data. In the context of deep learning, this usually equals training the network to output informative features. This is similar to the motivation for pre-training the network on a larger, related dataset as discussed in Section 2.2.4. What counts as informative features depends on what the main goal of the neural network is, hence a common way to evaluate unsupervised learning methods are by applying the pre-trained network on a downstream task. This could, for example be semantic segmentation where the network that has been pre-trained in an unsupervised fashion is trained for semantic segmentation. The quality of the unsupervised pre-training is then decided by the accuracy and training time of the downstream task.

Autoencoder

An autoencoder is a neural network trained with backpropagation in an unsupervised manner [73]. Instead of using annotated labels as training target the autoencoder is trained to output a copy of the input image, *i.e.* an autoencoder learns the identity mapping for the training images.

Since this task is trivial, constraints need to be but on the structure of the autoencoder network. A simple way of doing this is to add a bottleneck in the network, *i.e.* a hidden layer with low dimensionality. In this way the network needs to learn a compact encoding of the input data that contains all information needed for it to be decoded to the original image. The features at the bottleneck of the autoencoder are often referred to as code or latent representation. In the

standard setup, the code has a lower dimensionality than the input data. However, it has been shown that autencoders where the dimensionality of the code is higher than the input also can learn meaningful encodings as well [74]. This by applying regularization to the latent representation.

Self-supervised Learning

In self-supervised learning the annotated labels are replaced by pseudo-labels that can be automatically generated from data. The network is then trained for this pretext task by learning to predict the pseudo-labels. For the network to learn to output useful features, the pretext task need to be meaningful considering the downstream task. Examples of pretext tasks are image colorization [75], image inpainting [76], image jigsaw puzzle solving [77, 78] and rotation prediction [79]. In [80] the pseudo-labels are created by k-means clustering of the output feature vectors. The network is then trained to predict the pseudo-labels. By iterating the k-means clustering and the pseudo-label prediction, the network learns to outputs features useful for several downstream tasks.

2.5.2 Semi-supervised Learning

Semi-supervised learning defines a middle ground between supervised learning and unsupervised learning. In the semi-supervised settings, one part of the available dataset has labels while the rest is unlabeled. Semi-supervised learning seeks to alleviate the need for a large amount of annotated data by allowing a network to leverage the unlabeled part of the dataset. This is generally achieved by adding another loss term during training that can be applied to unlabeled data. These losses can be divided into three main classes [81]: entropy minimization, consistency regularization and generic regularization. A short introduction to generic regularization can be found in Section 2.2.4

Entropy Minimization

Entropy minimization encourages the network to increase its confidence on unlabelled data. This is based on the assumption that the decision boundary should not pass by high density regions of the data distribution [82]. In [83], the entropy of the network on output data is minimized explicitly while in [84] this is done implicitly by letting the network create pseudo-labels for the unlabelled data which is then used as targets during training.

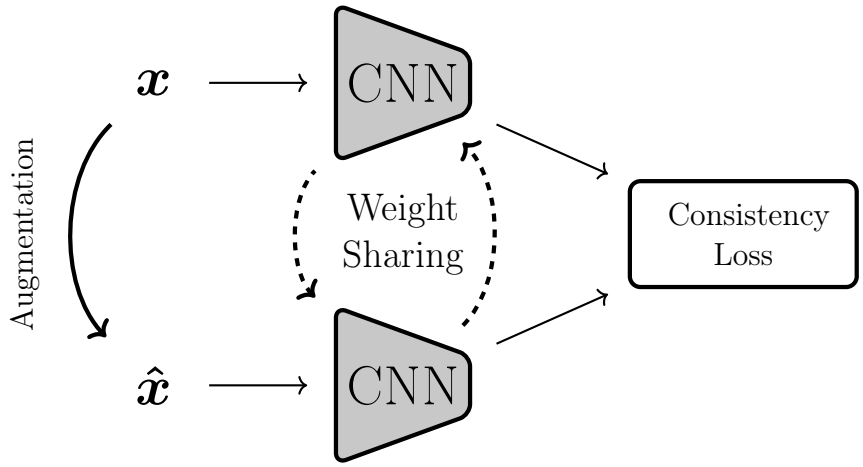

Figure 2.5: *A consistency regularization training schematic for semi-supervised learning. The network is trained to produce consistent output for the original input,* $\boldsymbol{x}$*, and an augmented version of the image,* $\hat{\boldsymbol{x}}$*. The assumption made is that the augmentation should not change the labels.*

Consistency Regularization

Consistency regularization leverages ideas from data augmentation, generally used as an regularization technique during supervised learning. In data augmentation, a transformation is applied to the input image before being fed to the network. The transformation is often randomized and is designed in a way such that the original labels can be used for the augmented image. A similar approach can be used in the semi-supervised setting for the unlabelled samples. The main idea being that the network should output consistent labels for the original image, $\boldsymbol{x}$, and an augmented image, $\hat{\boldsymbol{x}}$. This is enforced by minimizing a consistency loss over the two outputs, see Fig. 2.5. In this way, the network learns to be invariant to the image changes that can be inferred by the augmentation routines used. In [85] and [86], this consistency is enforced by minimizing the L^2 distance between the outputs of the network for $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$, while in [87] and [88] the network outputs are viewed as probability distributions and the Kullback–Leibler divergence between them is minimized.

Another type of consistency regularization is presented in [89], here a student network is trained to produce consistent output to a "mean teacher" whose weights are an ensemble of a student network's weights.

2.5.3 Weakly-supervised Learning

Weakly-supervised learning aims at utilizing weaker, more easily acquired, labels for training. For semantic segmentation weak labels such as image tags [90–95], object bounding boxes [91, 96, 97], points [98] or scribbles [99] have been used to alleviate the need for dense annotations. The general approach is to infer dense labels from the weak labels, the current network output and using additional information or assumptions. An example is [95] where dense labels are inferred by seeded region growing from discriminative regions in the image. The discriminative regions are found by looking for highly activated regions in intermediate feature maps of the network. In [91] both image tags and bounding boxes were utilized in a weakly- and semi-supervised setting. An expectation–maximization method was proposed for training the network using the weak labels.

Chapter 3

Summary

The topics of this thesis revolves around image segmentation, and can be divided into to main parts. The first one being development of methods to utilize 3D geometry to improve segmentation methods and the second one being development of DSMs for semantic segmentation. This chapter includes a summary of these two parts followed by short individual summaries of the papers included in this thesis.

Geometric Supervision for Segmentation

This work started as part of a project addressing the task of semantic localization, *i.e.* utilizing semantic cues to estimate the pose of a camera given the image taken and an 3D map. One step of the pipeline required accurate semantic segmentations for road-scenarios. What we noticed, when trying some state-of-the-art models trained on the cityscapes dataset [72], was that these performed very poorly on our images. Especially for images taken during different seasons or lighting conditions.

In the same project we had created large 3D models of the same localization at different seasons and time of the day. Paper I summarizes our effort to utilize these 3D models to train a CNN that performs well for all image conditions present in the localization dataset, being more robust to seasonal changes and weather conditions. In short, we create a dataset consisting of pairs of images with 2D-2D pixel correspondences. This is done by geometrically matching the 3D models created during different seasons or time of the day. Given the 3D point matches, the pixel positions for the 2D-2D correspondences in each image can be calculated using the camera positions available in the 3D models.

With the insight that two pixels in each 2D-2D correspondence pair should depict the same object, these can be used during training. To this end, we formulated a loss function that encourages the output of the CNN to be consistent over

every pair of pixel correspondences. This is related to the work on consistency regularization presented in Section 2.5.2, but instead of relying on augmentation methods for creating the input pairs these are created using the 3D models. In this way we can learn the network to be invariant to higher level visual changes, such as summer to winter or day to night.

Creating the dataset of 2D-2D correspondences are not completely automatic and requires some manual labours for the alignment of the different 3D models. Hence, the consistency training across the correspondences falls under weakly-supervised training. This makes the complete training routine used in Paper I both semi- and weakly-supervised. An estimated 30 hours of manual labor was required to create one of the 2D-2D correspondence dataset, which contains 28766 image pairs. This in comparison to the Cityscapes dataset [72], where each image required around 1.5 hours of annotation time.

The segmentation results in Paper I are quite convincing, by adding correspondence training we managed to improve upon several strong baselines, even for networks that have been trained on the Mapillary Vistas dataset [2] that is already quite diverse when it comes to variety in lighting conditions and seasons. However, we failed with one of the goals of the work, to show that the improved and more robust segmentation algorithm gave a significant performance boost for semantic visual localization algorithms. Even though there was a small improvement it was less than expected.

Having segmentations that are consistent across lighting conditions and seasonal changes is a crucial part of semantic long-term visual localization. This because the invariance of semantic meaning of the surrounding is one of the assumptions made when developing these localization methods. However, in many cases there are only a few classes available that is meaningful for localization. For example, the Cityscapes dataset [72] contains 19 classes, 8 of which cover dynamic objects such as cars or pedestrians that are not useful for localization. The Mapillary Vistas dataset [2] contains 66 classes, with 15 classes for dynamic objects. Hence using semantic labels for visual localization results in a loss of discriminative power. To combat this loss in discriminative power we present the Fine Grained Segmentation Network (FGSN) in Paper II.

In Paper II we draw inspiration from unsupervised learning and do k-means clustering of the output features of the CNN. Similar to the work in [80], the cluster indices can then be used as pseudo-labels during training. In addition, we use the 2D-2D correspondences from Paper I to train the network to output consistent labels across seasons. In this way, we can arbitrarily choose the number of output classes, by setting k, without having to do any additional annotating. During inference the FGSN will output a dense segmentation map consisting of k

classes, and even though the different classes might not have any specific semantic meaning, they will be useful for localization due to the consistency training. Using the FGSN output instead of standard semantic segmentation maps improves localization performance for several semantic visual localization algorithms.

Deep Structured Models

In the second part of the thesis, development of DSMs, emphasis has been put on creating models that are possible to train end-to-end as well as the methods needed for training. Paper III and Paper IV are obvious examples of this. In Paper V a robust segmentation method for abdominal organs using CNNs is developed. The original idea was to use the DSM and training routines developed in paper IV and add it to this framework as well. However, this did not actually improve the results much and was hence discarded. This brings up an important question, what types of CRFs are needed to improve on the results of a CNN? This is something that will be discussed in Chapter 4.

Regardless of using a DSM or not, Paper V presents a robust method for abdominal organ segmentation. The paper combines a robust organ localization with the use of specialized organ CNNs for segmentation. Since segmentation is a key problem in medical image analysis, a method for organ segmentation can be crucial for numerous applications in medical research and clinical care such as computer aided diagnosis and surgery assistance. In addition, robustness is something that is generally highly valuable for medical applications.

Paper IV introduces a method of training a DSM end-to-end using a max-margin objective. However, several restrictions to the CRF is needed to be able to do the training efficiently. The CRF used has a pairwise term where each pixel is only connected to its closest neighbours. This can easily be extended, however the method uses graph-cut inference of the CRF during training which becomes slow for densely connected CRFs.

In Paper III a framework for training DSMs with more expressive CRFs is presented, here a newly developed approximate inference method of the CRF is used. However, we would like to point out that this should not be seen as a strictly better approach than the one used in Paper IV. The max-margin approach of Paper IV has the advantage of using a fast and exact inference algorithm of the CRF. In addition the learning approach used have been shown to generalize well, even for smaller datasets. Something that suited the experiments on smaller medical data sets presented in the paper well.

The more expressive DSM presented in Paper III is however suitable for larger datasets with more semantic classes. This enabled us to do experiments on for ex-

ample the PASCAL VOC 2012 segmentation benchmark [100] consisting of several thousands of annotated images and 21 semantic classes.

At the time of development of these methods, segmentation was all about the evaluation numbers. The methods achieving the highest mean Intersection over Union on the major benchmarks get the most attention. This is of course positive in the manner of encouraging researchers to develop practically useful and accurate methods. The major benchmarks are also an invaluable tool for comparing different methods. However, the chase for better numbers combined with the use of very data-hungry deep learning brings on a research pipeline that is skewed towards parameter tuning instead of method development. This also impacts the possibility to reproduce previous works.

Unfortunately, the methods presented in Paper IV and Paper III fail to achieve state-of-the-art results on the major benchmarks. We have however successfully shown the usefulness of both approaches in each paper respectively. Perhaps most notable for Paper IV is the performance of the method on smaller medical datasets. For Paper III the model has the ability to improve on a strong CNN baseline trained on a lot of additional data, even though the DSM was only trained end-to-end on a subset of the data.

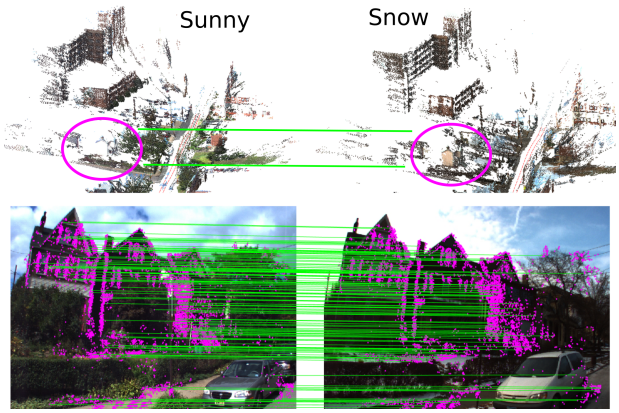

Figure 3.1: *In Paper I, pixel-to-pixel correspondences for images taken during different seasons are created by aligning 3D models. These correspondences can be used for training semantic segmentation networks robust to seasonal changes.*

3.1 Paper I

M. Larsson, E. Stenborg, C. Toft, L. Hammarstrand, M. Pollefeys, T. Sattler and F. Kahl "A Cross-Season Correspondence Dataset for Robust Semantic Segmentation". *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* 2019.

This paper presents a method to utilize 2D-2D point matches between images taken during different image conditions for training convolutional neural networks for semantic segmentation. Enforcing label consistency across the matches makes the final segmentation algorithm robust to seasonal changes. The 2D-2D matches are generated with little human interaction by geometrically matching points from 3D models built from images. Since only geometric information is used to align the models, accurate alignment of models created during day/night or summer/winter is possible. Adding the correspondences as extra supervision during training improves the segmentation performance of the convolutional neural network, making it more robust to seasonal changes and changes in weather conditions.

Enabling the use of 2D-2D correspondences, that can be created with little human supervision, for training provides an alternative to manually creating pixel-level annotations that is both cheaper and faster. For the training setup used in the paper, parallels can be drawn to recent work on semi-supervised learning. For example, in [88], a similar training setup is used, enforcing consistency between an image from the training set and an augmented version of the same image. The approach in Paper I differs by creating cross-season correspondences for the training pairs, instead of relying on data augmentation.

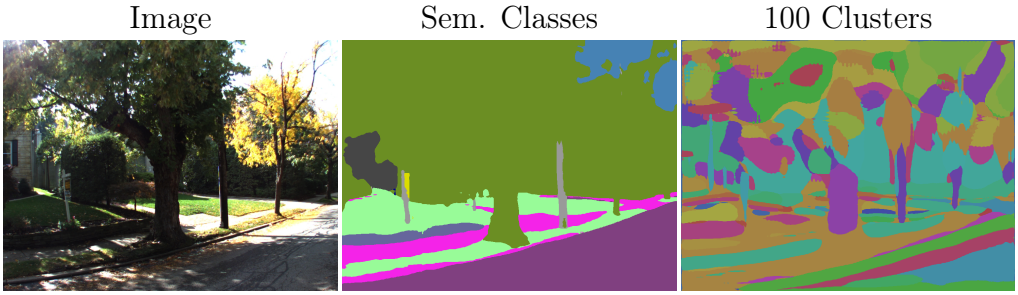

Figure 3.2: *In Paper II, a neural network that automatically discovers a large set of fine-grained clusters is trained. It is experimentally shown that using a larger number of clusters, instead of a small set of human-defined semantic classes, improves localization performance.*

3.2 Paper II

M. Larsson, E. Stenborg, C. Toft, L. Hammarstrand, T. Sattler and F. Kahl "Fine-Grained Segmentation Networks: Self-Supervised Segmentation for Improved Long-Term Visual Localization". *IEEE International Conference on Computer Vision (ICCV)* 2019.

A big challenge for long-term visual localization is handling large appearance changes. In order to gain robustness to such changes, many approaches use semantic segmentations as an invariant scene representation, as the semantic meaning of each scene part should not be affected by seasonal and other changes. However, these representations are typically not very discriminative due to the limited number of available classes. To provide more discriminative semantic representation, this paper proposes the use of Fine-Grained Segmentation Networks (FGSN).

FGSNs are a novel type of convolutional neural networks that output dense fine-grained segmentations. Using k-means clustering, FGSNs can be trained in a self-supervised manner, using the cluster assignments of image features as labels. This enables the use of arbitrarily many output classes without having to create annotations manually. In addition, the 2D-2D correspondence dataset from Paper I is used to ensure that the classes are stable under seasonal changes.

Altough the fine-grained segmentations are not trained to specifically contain any semantic information, the fact that they are consistent across seasonal changes make them useful for visual localization. In fact, using fine-grained segmentations instead of standard semantic segmentations improves performance for several visual localization methods utilizing segmentations. The improvement is most prevalent for cases without much semantic information, such as park areas where the majority of the image is segmented as vegetation, see Fig. 3.2.

3.3 Paper III

M. Larsson, A. Arnab, F. Kahl, S. Zheng, and P. Torr. "Revisiting Deep Structured Models in Semantic Segmentation with Gradient-Based Inference". *SIAM Journal on Imaging Sciences.* 2018.

In this paper we move from binary label CRFs with short spatial pairwise interactions to CRFs being able to handle multiple labels and learn pairwise interactions on larger distances. We present an inference technique based on gradient descent on the Gibbs energy of the CRF. This inference method consists only of differentiable operations which enables us to unroll the CRF inference as a number of update steps of a Recurrent Neural Network (RNN). During learning, we can also back-propagate through the RNN and do end-to-end training of the entire model.

Two different types of CRF models are presented in the paper. The first one consists of a spatial pairwise term as well as a high-dimensional bilateral kernels. In contrast to many previous works we do not restrict these two kernels to have Gaussian shape but allow for arbitrary shape of the spatial and bilateral kernels. In addition, we introduce a new type of potential function which is image-dependent like the bilateral kernel, but an order of magnitude faster to compute since only spatial convolutions are employed. The major contributions of the paper are

- A new model for a pairwise CRF potential which is image-dependent like the bilateral kernel, but does not require high-dimensional filtering. It is based on a learned 2D filter bank which makes both inference and learning an order of magnitude faster than high-dimensional filtering approaches.
- A new optimization method for CRF inference based on gradient descent that enables end-to-end training.
- We show that our inference method supports learning pairwise kernels of arbitrary shape. The learned kernels are empirically analyzed and it is demonstrated that in many cases non-Gaussian potentials are preferred.

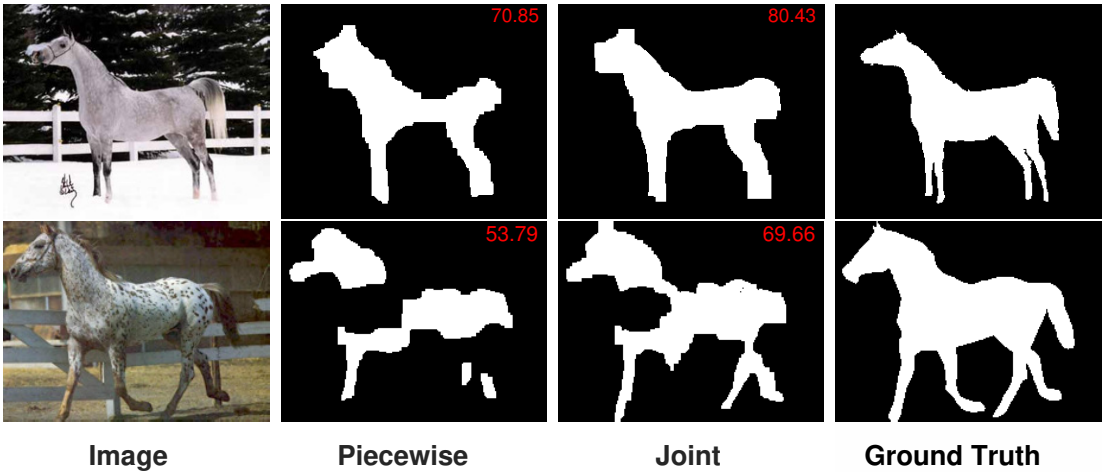

Figure 3.3: *Comparison of piecewise versus joint training of a deep structured model for some hand-picked example. The number shown in the upper right corner is the Jaccard index (%).*

3.4 Paper IV

M. Larsson, J. Alvén and F. Kahl. "Max-Margin Learning of Deep Structured Models for Semantic Segmentation". *Scandinavian Conference on Image Analysis (SCIA)*. 2017.

This paper presents a method for learning the parameters of a Deep Structured Model used for semantic segmentation. The learning problem is formulated as a Structured Support Vector Machine (SSVM) and we show that it is possible to calculate the derivative of the objective with respect to the output of the CNN. This enables us to back-propagate all through the layers of the CNN and learn the weights of the CNN and the CRF at the same time. Since the SSVM uses a max-margin loss function that generally gives good generalization capabilities of the trained model this method is especially suitable for application where labelled data is limited.

Figure 3.3 shows a comparison of the piecewise and jointly trained models. As can be seen the model where the CNN and CRF have been trained jointly performs better, avoiding error such as cutting of the legs of the horse. This is because the CNN has learnt to compensate for the slight shrinking effect of the CRF.

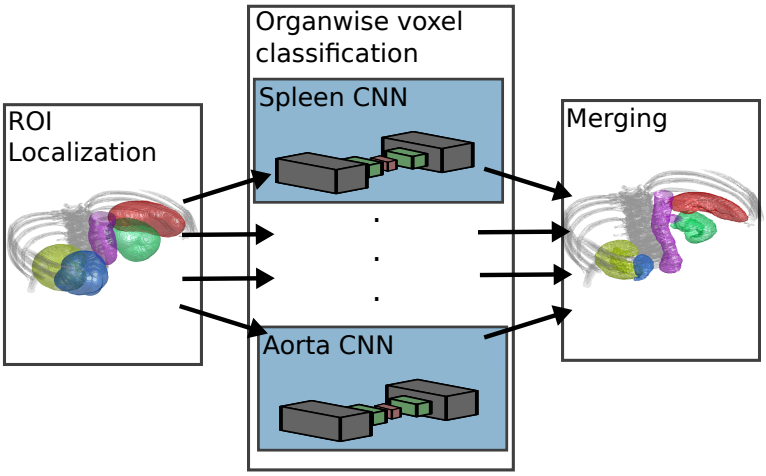

Figure 3.4: *Graphical representation of the method presented in Paper V.*

3.5 Paper V

M. Larsson, Y. Zhang and F. Kahl "Robust Abdominal Organ Segmentation Using Regional Convolutional Neural Networks". *Applied Soft Computing.* 2018.

This paper presents a method for segmenting 13 different abdominal organs utilizing CNNs. The method can be divided into two main steps. Firstly, an efficient and robust feature registration method is applied estimating the center-point of each organ. Secondly, a convolutional neural network performing voxelwise classification is applied to a region, defined by a prediction mask placed at the estimated organ center-point. The prediction mask is created using the ground truths of each organ in the training set. The approach of first localizing a region of interest for each organ transforms the problem the CNN has to solve from a large multi-label problem to 13 smaller binary-labels problems. We can therefore train smaller CNNs and more specialized, or regional, networks that only need to differentiate between a certain organ and the background.

During the development of this method and at the writing of the first draft of this paper there were very few examples of deep learning methods applied to medical 3D segmentation tasks and none for this specific task. Since then, there has been a lot of development in this area and several papers have been published [101–103] further showing that deep learning methods can perform really well on these types of tasks.

Chapter 4

Outlook

The field of semantic segmentation has moved at a high pace during the last few years, especially when it comes to methods based on CNNs. Almost on a monthly basis, there has been a new CNN with a different architecture pushing state-of-the-art further. During 2015 and 2016 the results of CNNs presented for semantic segmentation could be greatly increased by adding a CRF [7, 28]. This is mainly due to the fact that the architectures of the networks used during this time did not allow the CNN to learn interactions over long ranges. In addition, the downsampling of the pooling layers resulted in a loss of spatial information that prohibited the accurate segmentation of fine edges between classes, hence the output usually became "blobby". Both of these errors are something that the most commonly used CRF models excel at. However, during late 2016 and 2017 several new CNNs have been proposed, raising state-of-the-art, that do not use a CRF for post-processing [47, 104]. This indicates that the type of CRFs commonly used in semantic segmentation might not be necessary for these large scale problems with a lot of annotated data. This trend has continued during the last few years.

An additional detail with the CRFs is that inference for most of the commonly used models is still fairly slow, especially for dense models with edge-aware pairwise potentials. Some work has been done on creating alternatives to these CRFs that are less computationally demanding but still has the ability to refine segmentations near edges [105, 106]. This is also adressed as part of Paper III.

So, for CRFs and DSMs to be really useful for large scale segmentation problems in the future there are two improvements needed. Firstly, the inference needs to be faster, adding a CRF should not speed down the inference or training considerably. Unless, of course, the gain in performance is worth it. Secondly, there might be a need to rethink the type of models we are using and try to create CRFs that are better suited to correct the errors that the new state-of-the-art CNNs do.

Moving away from the large scale segmentation benchmarks there are still a lot of applications where adding a CRF gives a big increase in performance. Looking at datasets with slightly smaller training set, DSMs tend to perform better in general. CRFs are also a good way to include prior knowledge in you segmentation pipeline. Previous work has shown that geometric constraints, such as convex or star shaped fore-ground objects only can be enforced by a CRF [107, 108].

As discussed in section 2.5.3, there exists a lot of previous work on weakly-supervised learning. However, when it comes to unsupervised learning and semi-supervised learning the majority of work done in vision is related to image classification. Since the annotations needed for supervised training of segmentation CNNs is especially time-consuming to acquire, developing methods for weak-, semi- or un-supervised training of segmentation CNNs is an interesting and important research direction.

4.1 Future Work

4.1.1 Structured Output

At the moment many of the top entries of the major segmentation benchmarks train CNNs with a pixel-wise loss function, disregarding the fact that the output is structured. Modern CNNs have the capability to, and probably do, implicitly learn that there is structure in the output. However, actually taking the output structure into account, whether by using a CRF or in some other way, could be beneficial. There is hence interest in continuing the work on end-to-end training of DSMs, both trying to improve computation speed and to design more expressive models. In addition it would be interesting to do more work on DSMs for medical image segmentation where more application-specific types of CRFs might be needed.

Another interesting approach to taking output structure into account was introduced in [109], which used a Generative Adversarial Network (GAN) for semantic segmentation. The idea was that the discriminator would be able to learn how an ground truth segmentation should look like. Hence during training, the generator, that also performs the actual segmentation, would have to output realistic segmentation to trick the discriminator. In this way the output of the segmentation CNN would have to follow, and hence learn, the output structure of the segmentations. This introduces an interesting opportunity to learn output structure without having to perform CRF inference. This idea has been further refined in [110] and used in medical applications in [111, 112]. Since these approaches are able to take output structure into account during training without having to deal with the drawback of using CRFs, this could be an interesting research direction.

4.1.2 Weak Supervision

The underlying method used in Paper I is fairly simple. Given a way to create pairs of input samples that should have the same class or output vector, consistency regularization training can be used to utilize these pairs during training. As previously mentioned, the pairs can be created completely unsupervised by utilizing data augmentation methods [88]. Creating more advanced training pairs, which enables the networks to learn to be invariant to higher levels changes, is an interesting research direction. This has been explored to some extent, for example in [113] corresponding day/night image pairs are used for training of a segmentation network. However, there is still a lot left to explore. In the self-driving car scenario, most cars are equipped with several sensor. This opens up for ways of utilizing data from different sensors, *e.g.* LIDAR, to create weak labels for semantic segmentation, or vice-versa. In addition, the images are collected sequentially which opens up the opportunity to utilize temporal consistency during training.

4.1.3 Visual Localization

In Paper II, we show how 3D models of the same location created at different seasons and time of the day can be utilized to train segmentation networks useful for long-term visual localization. An important step in creating these 3D models were to localize each camera in the coordinate system of an initial 3D model. At the time of writing the paper, some manual labour was needed to align the 3D model correctly, but with the improvement of long-term visual localization methods this might be possible to do automatically. The work in Paper I shows that semantic algorithms can be improved via visual localization and Paper II shows that improved segmentations can lead to improved and more robust localization results. This lead to an interesting question, is it possible to create a positive feedback loop, iteratively improving both segmentation and localization?

The concept of bi-level learning was introduced in Section 2.4.2 and was used in Paper IV to train a DSM. Many interesting applications can be formulated as bi-level learning problems, for some examples in conjunction with CNNs see [70]. For visual localization, pose refinement is usually done by local optimization. Hence, if we want to learn a good pose optimizer we can formulate it as a bi-level learning problem, where the inner optimization problem is that of pose optimization and the outer optimization problem is that of minimizing the learning loss. This is one of our current projects, by utilizing the bi-level optimization formulation we can directly train a CNN to output features for each pixel in an image that, given a specific pose optimization method, minimizes a loss based on the distance between the estimated camera pose and the ground truth camera pose.

Bibliography

[1] A. Arnab, S. Zheng, S. Jayasumana, B. Romera-Paredes, M. Larsson, A. Kirillov, B. Savchynskyy, C. Rother, F. Kahl, and P. H. S. Torr, "Conditional random fields meet deep neural networks for semantic segmentation: Combining probabilistic graphical models with deep learning for structured prediction," *Signal Processing Magazine*, 2018.

[2] 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 (ICCV)*, 2017.

[3] Z. Xu, "Multi-atlas labeling beyond the cranial vault - workshop and challenge," 2016, [Online; accessed 10-January-2017]. [Online]. Available: https://www.synapse.org/\#!Synapse:syn3193805/wiki/217752

[4] D. G. Lowe, "Object recognition from local scale-invariant features," in *International Conference on Computer Vision (ICCV)*, 1999.

[5] P. Krähenbühl and V. Koltun, "Efficient inference in fully connected CRFs with gaussian edge potentials," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2011.

[6] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected CRFs," in *International Conference on Learning Representations (ICLR)*, 2015.

[7] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. Torr, "Conditional random fields as recurrent neural networks," in *International Conference on Computer Vision (ICCV)*, 2015.

[8] L. Ladicky, C. Russell, P. Kohli, and P. H. Torr, "Associative hierarchical crfs for object class image segmentation," in *International Conference on Computer Vision (ICCV)*, 2009.

[9] P. Krähenbühl and V. Koltun, "Efficient inference in fully connected crfs with gaussian edge potentials," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2011.

[10] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[11] A. Hanson, "Visions: A computer system for interpreting scenes," *Computer Vision Systems*, 1978.

[12] Y.-i. Ohta, T. Kanade, and T. Sakai, "An analysis system for scenes containing objects with substructures," in *International Joint Conference on Pattern Recognitions*, 1978.

[13] H. Zhu, F. Meng, J. Cai, and S. Lu, "Beyond pixels: A comprehensive survey from bottom-up to semantic image segmentation and cosegmentation," *Journal of Visual Communication and Image Representation*, 2016.

[14] 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 (IJCV)*, 2009.

[15] 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 (ECCV)*, 2016.

[16] Z. Liu, X. Li, P. Luo, C. C. Loy, and X. Tang, "Deep learning markov random field for semantic segmentation," *Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2017.

[17] F. Rosenblatt, "Principles of neurodynamics. perceptrons and the theory of brain mechanisms," Cornell Aeronautical Lab, Tech. Rep., 1961.

[18] K. Fukushima and S. Miyake, "Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition," in *Competition and Cooperation in Neural Nets*, 1982.

[19] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2012.

[20] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet large scale visual recognition challenge," *International Journal of Computer Vision (IJCV)*, 2015.

[21] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, 2015.

[22] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, 1989.

[23] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *International Conference on Machine Learning (ICML)*, 2010.

[24] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, 1998.

[25] M. Matsugu, K. Mori, Y. Mitari, and Y. Kaneda, "Subject independent facial expression recognition with robust face detection using a convolutional neural network," *Neural Networks*, 2003.

[26] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *International Conference on Learning Representations (ICLR)*, 2015.

[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[28] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs." *Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2018.

[29] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *International Conference on Learning Representations (ICLR)*, 2016.

[30] M. Ranzato, F. J. Huang, Y. L. Boureau, and Y. LeCun, "Unsupervised learning of invariant feature hierarchies with applications to object recognition," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[31] D. E. Rumelhart, G. E. Hinton, R. J. Williams *et al.*, "Learning representations by back-propagating errors," *Cognitive Modeling*, 1988.

[32] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *International Conference on Machine Learning (ICML)*, 2013.

[33] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research (JMLR)*, 2011.

[34] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," *International Conference on Learning Representations (ICLR)*, 2015.

[35] Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio, "Identifying and attacking the saddle point problem in high-dimensional non-convex optimization," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

[36] A. Choromanska, M. Henaff, M. Mathieu, G. B. Arous, and Y. LeCun, "The loss surfaces of multilayer networks," in *Conference on Artificial Intelligence and Statistics (AISTATS)*, 2015.

[37] N. Sivastava, G., A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research (JMLR)*, 2014.

[38] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning (ICML)*, 2015.

[39] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[40] A. Ahmed, K. Yu, W. Xu, Y. Gong, and E. Xing, "Training hierarchical feed-forward visual recognition models using transfer learning from pseudo-tasks," *European Conference on Computer Vision (ECCV)*, 2008.

[41] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[42] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

[43] M. D. Zeiler, G. W. Taylor, and R. Fergus, "Adaptive deconvolutional networks for mid and high level feature learning," in *International Conference on Computer Vision (ICCV)*, 2011.

[44] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," *arXiv preprint arXiv:1412.6806*, 2014.

[45] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2017.

[46] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2015.

[47] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[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 (ECCV)*, 2018.

[49] S. Chandra, N. Usunier, and I. Kokkinos, "Dense and low-rank gaussian crfs using deep embeddings," in *International Conference on Computer Vision (ICCV)*, 2017.

[50] G. Lin, C. Shen, A. van den Hengel, and I. Reid, "Efficient piecewise training of deep structured models for semantic segmentation," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[51] V. Vineet, J. Warrell, and P. H. S. Torr, "Filter-based mean-field inference for random fields with higher-order terms and product label-spaces," *International Journal of Computer Vision (IJCV)*, 2014.

[52] C. Wojek and B. Schiele, "A dynamic conditional random field model for joint labeling of object and scene classes," *European Conference on Computer Vision (ECCV)*, 2008.

[53] P. Kohli, P. H. Torr *et al.*, "Robust higher order potentials for enforcing label consistency," *International Journal of Computer Vision (IJCV)*, 2009.

[54] 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 (ECCV)*, 2016.

[55] M. Li, A. Shekhovtsov, and D. Huber, "Complexity of discrete energy minimization problems," in *European Conference on Computer Vision (ECCV)*. Springer, 2016.

[56] J. Kappes, B. Andres, F. Hamprecht, C. Schnorr, S. Nowozin, D. Batra, S. Kim, B. Kausler, J. Lellmann, N. Komodakis *et al.*, "A comparative study of modern inference techniques for discrete energy minimization problems," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[57] V. Kolmogorov and R. Zabin, "What energy functions can be minimized via graph cuts?" *Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2004.

[58] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2001.

[59] A. Desmaison, R. Bunel, P. Kohli, P. H. Torr, and M. P. Kumar, "Efficient continuous relaxations for dense crf," in *European Conference on Computer Vision (ECCV)*. Springer, 2016.

[60] C. Sutton and A. McCallum, "Piecewise training for undirected models," *arXiv preprint arXiv:1207.1409*, 2012.

[61] A. Kolesnikov, M. Guillaumin, V. Ferrari, and C. H. Lampert, "Closed-form training of conditional random fields for large scale image segmentation," *arXiv preprint arXiv:1403.7057*, 2014.

[62] A. Kirillov, D. Schlesinger, S. Zheng, B. Savchynskyy, P. H. Torr, and C. Rother, "Joint training of generic cnn-crf models with stochastic optimization," in *Asian Conference on Computer Vision (ACCV)*. Springer, 2016.

[63] B. Taskar, V. Chatalbashev, D. Koller, and C. Guestrin, "Learning structured prediction models: A large margin approach," in *International Conference on Machine Learning (ICML)*, 2005.

[64] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, "Large margin methods for structured and interdependent output variables," *Journal of Machine Learning Research (JMLR)*, 2005.

[65] M. Szummer, P. Kohli, and D. Hoiem, "Learning CRFs using graph cuts," in *European Conference on Computer Vision (ECCV)*, 2008.

[66] L.-C. Chen, A. Schwing, A. Yuille, and R. Urtasun, "Learning deep structured models," in *International Conference on Machine Learning (ICML)*, 2015.

[67] M. Larsson, A. Arnab, F. Kahl, S. Zheng, and P. H. S. Torr, "A projected gradient descent method for crf inference allowing end-to-end training of arbitrary pairwise potentials," in *International Conference on Energy Minimization Methods in Computer Vision and Pattern Recognition, (EMMCVPR)*, 2017.

[68] J. Tompson, A. Jain, Y. LeCun, and C. Bregler, "Joint training of a convolutional network and a graphical model for human pose estimation," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

[69] J. Geiping and M. Moeller, "Parametric majorization for data-driven energy minimization methods," in *International Conference on Computer Vision (ICCV)*, 2019.

[70] S. Gould, R. Hartley, and D. Campbell, "Deep declarative networks: A new hope," *arXiv preprint arXiv:1909.04866*, 2019.

[71] K. G. Samuel and M. F. Tappen, "Learning optimized map estimates in continuously-valued mrf models," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[72] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[73] A. Ng *et al.*, "Sparse autoencoder," *CS294A Lecture notes*, 2011.

[74] Y. Bengio *et al.*, "Learning deep architectures for ai," *Foundations and trends in Machine Learning*, 2009.

[75] G. Larsson, M. Maire, and G. Shakhnarovich, "Colorization as a proxy task for visual understanding," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[76] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[77] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *European Conference on Computer Vision (ECCV)*, 2016.

[78] C. Wei, L. Xie, X. Ren, Y. Xia, C. Su, J. Liu, Q. Tian, and A. L. Yuille, "Iterative reorganization with weak spatial constraints: Solving arbitrary jigsaw puzzles for unsupervised representation learning," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[79] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *International Conference on Learning Representations (ICLR)*, 2018.

[80] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *European Conference on Computer Vision (ECCV)*, 2018.

[81] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. A. Raffel, "Mixmatch: A holistic approach to semi-supervised learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[82] O. Chapelle and A. Zien, "Semi-supervised classification by low density separation." in *Conference on Artificial Intelligence and Statistics (AISTATS)*, 2005.

[83] Y. Grandvalet and Y. Bengio, "Semi-supervised learning by entropy minimization," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2005.

[84] D.-H. Lee, "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," in *Workshop on Challenges in Representation Learning, ICML*, 2013.

[85] S. Laine and T. Aila, "Temporal ensembling for semi-supervised learning," *arXiv preprint arXiv:1610.02242*, 2016.

[86] M. Sajjadi, M. Javanmardi, and T. Tasdizen, "Regularization with stochastic transformations and perturbations for deep semi-supervised learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[87] T. Miyato, S.-i. Maeda, M. Koyama, and S. Ishii, "Virtual adversarial training: a regularization method for supervised and semi-supervised learning," *Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2018.

[88] Q. Xie, Z. Dai, E. Hovy, M.-T. Luong, and Q. V. Le, "Unsupervised data augmentation," *arXiv preprint arXiv:1904.12848*, 2019.

[89] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[90] D. Pathak, E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional multi-class multiple instance learning," *arXiv preprint arXiv:1412.7144*, 2014.

[91] G. Papandreou, L.-C. 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 (ICCV)*, 2015.

[92] P. O. Pinheiro and R. Collobert, "From image-level to pixel-level labeling with convolutional networks," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[93] F. Saleh, M. S. A. Akbarian, M. Salzmann, L. Petersson, S. Gould, and J. M. Alvarez, "Built-in foreground/background prior for weakly-supervised semantic segmentation," in *European Conference on Computer Vision (ECCV)*. Springer, 2016.

[94] A. Kolesnikov and C. H. Lampert, "Seed, expand and constrain: Three principles for weakly-supervised image segmentation," in *European Conference on Computer Vision (ECCV)*. Springer, 2016.

[95] Z. Huang, X. Wang, J. Wang, W. Liu, and J. Wang, "Weakly-supervised semantic segmentation network with deep seeded region growing," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[96] J. Dai, K. He, and J. Sun, "Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation," in *International Conference on Computer Vision (ICCV)*, 2015.

[97] A. Khoreva, R. Benenson, J. H. Hosang, M. Hein, and B. Schiele, "Simple does it: Weakly supervised instance and semantic segmentation." in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[98] 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 (ECCV)*, 2016.

[99] D. Lin, J. Dai, J. Jia, K. He, and J. Sun, "Scribblesup: Scribble-supervised convolutional networks for semantic segmentation," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[100] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *International Journal of Computer Vision (IJCV)*, 2015.

[101] E. Gibson, F. Giganti, Y. Hu, E. Bonmati, S. Bandula, K. Gurusamy, B. Davidson, S. P. Pereira, M. J. Clarkson, and D. C. Barratt, "Automatic multi-organ segmentation on abdominal ct with dense v-networks," *Transactions on Medical Imaging (TMI)*, 2018.

[102] X. Li, H. Chen, X. Qi, Q. Dou, C.-W. Fu, and P.-A. Heng, "H-denseunet: hybrid densely connected unet for liver and tumor segmentation from ct volumes," *Transactions on Medical Imaging (TMI)*, 2018.

[103] F. Isensee, J. Petersen, S. A. Kohl, P. F. Jäger, and K. H. Maier-Hein, "nnu-net: Breaking the spell on successful medical image segmentation," *arXiv preprint arXiv:1904.08128*, 2019.

[104] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[105] L.-C. Chen, J. T. Barron, G. Papandreou, K. Murphy, and A. L. Yuille, "Semantic image segmentation with task-specific edge detection using cnns and a discriminatively trained domain transform," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[106] G. Bertasius, L. Torresani, S. X. Yu, and J. Shi, "Convolutional random walk networks for semantic image segmentation," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[107] O. Veksler, "Star shape prior for graph-cut image segmentation," *European Conference on Computer Vision (ECCV)*, 2008.

[108] L. Gorelick, O. Veksler, Y. Boykov, and C. Nieuwenhuis, "Convexity shape prior for segmentation," in *European Conference on Computer Vision (ECCV)*, 2014.

[109] P. Luc, C. Couprie, S. Chintala, and J. Verbeek, "Semantic segmentation using adversarial networks," in *NIPS Workshop on Adversarial Training*, 2016.

[110] L. Samson, N. van Noord, O. Booij, M. Hofmann, E. Gavves, and M. Ghafoorian, "I bet you are wrong: Gambling adversarial networks for structured semantic segmentation," in *International Conference on Computer Vision (ICCV)*, 2019.

[111] Y. Xue, T. Xu, H. Zhang, L. R. Long, and X. Huang, "Segan: Adversarial network with multi-scale L1 loss for medical image segmentation," *Neuroinformatics*, 2018.

[112] F. Mahmood, D. Borders, R. Chen, G. N. McKay, K. J. Salimian, A. Baras, and N. J. Durr, "Deep adversarial training for multi-organ nuclei segmentation in histopathology images," *Transactions on Medical Imaging (TMI)*, 2019.

[113] C. Sakaridis, D. Dai, and L. V. Gool, "Guided curriculum model adaptation and uncertainty-aware evaluation for semantic nighttime image segmentation," in *International Conference on Computer Vision (ICCV)*, 2019.