Conditional Random Fields Meet Deep Neural Networks for Semantic Segmentation

Anurag Arnab*, Shuai Zheng*, Sadeep Jayasumana, Bernardino Romera-Paredes, Måns Larsson, Alexander Kirillov, Bogdan Savchynskyy, Carsten Rother, Fredrik Kahl and Philip Torr

***Abstract*—Semantic Segmentation is the task of labelling every pixel in an image with a pre-defined object category. It has numerous applications in scenarios where the detailed understanding of an image is required, such as in autonomous vehicles and medical diagnosis. This problem has traditionally been solved with probabilistic models known as Conditional Random Fields (CRFs) due to their ability to model the relationships between the pixels being predicted. However, Deep Neural Networks (DNNs) have recently been shown to excel at a wide range of computer vision problems due to their ability to learn rich feature representations automatically from data, as opposed to traditional hand-crafted features. The idea of combining CRFs and DNNs have achieved state-of-the-art results in a number of domains. We review the literature on combining the modelling power of CRFs with the representation-learning ability of DNNs, ranging from early work that combines these two techniques as independent stages of a common pipeline to recent approaches that embed inference of probabilistic models directly in the neural network itself. Finally, we summarise future research directions.**

***Keywords*—Conditional Random Fields, Deep Learning, Semantic Segmentation.**

I. Introduction

Scene Understanding is a long-standing problem in the field of computer vision and involves developing algorithms to interpret the contents of images with the level of comprehension of a human. Perceptual tasks, such as visual scene understanding, are performed effortlessly by humans. However, replicating the visual cortex on a computer has proven to be a challenging problem, that is still yet to be completely solved, almost 50 years after it was posed to an undergraduate student at MIT as a summer research project [1].

There are many ways of describing a scene, and current computer vision research addresses most of these problems independently, as illustrated in Figure 1. A high-level summary of a scene can be obtained by predicting image tags that describe the objects in the picture (such as "person") or the scene (such as "city" or "office"). This task is known as *image classification*. The *object detection* task, on the other hand, aims to localise different objects in an image by placing bounding boxes around each instance of a pre-defined object category. *Semantic Segmentation*, the main focus of this article, aims for a more precise understanding of the scene by assigning an object category label to each pixel within the image. Recently, researchers have also begun tackling new scene understanding problems such as *instance segmentation*, which aims to assign a unique identifier to each segmented object in the image, as well as bridging the gap between natural language processing and computer vision with tasks such as *image captioning* and *visual question answering*, which aim at describing an image in words, and answering textual questions from images respectively.

Scene Understanding tasks, such as semantic segmentation, enable computers to extract information from real world scenarios, and to leverage this information to accomplish given tasks. Semantic Segmentation has numerous applications such as in autonomous vehicles which need a precise, pixel-level understanding of their environment, developing robots which can navigate and manipulate objects in their environment, diagnosing medical conditions by segmenting cells, tissues and organs of interest, image- and video-editing and developing "smart-glasses" which describe the scene to the blind.

Semantic Segmentation has traditionally been approached using probabilistic models known as a Conditional Random Fields (CRFs), which explicitly model the correlations among the pixels being predicted. However, in recent years, deep neural networks have been shown to excel at a wide range of computer vision and machine learning problems as they can automatically learn expressive feature representations from massive datasets. Despite the representational power of deep neural networks, state-of-the-art segmentation algorithms, which are benchmarked on public computer vision datasets and evaluation servers where the test set is withheld, all include a CRF within their pipelines. Some approaches include CRFs as a separate stage of the pipeline whilst the leading ones incorporate it within the neural network itself.

In this article, we review CRFs and deep neural networks in the context of dense, pixel-wise prediction tasks, and how CRFs can be incorporated into neural networks to combine the advantages of these two models. Markov Random Fields (MRFs), Conditional Random Fields (CRFs) and more generally, probabilistic graphical models are ubiquitous tools with a long history of applications in a variety of domains spanning computer vision, computer graphics and image processing [2]. This is due to their ability to model correlations in the variables being predicted. Deep Neural Networks (DNNs), on the other hand, are also fast becoming the de facto method of choice in a variety of machine learning tasks as they can learn rich

A. Arnab* and S. Zheng* are the joint first authors. A. Arnab, S. Zheng, S. Jayasuamana, B. Romera-Paredes, and P. Torr are with the Department of Engineer Science, University of Oxford, United Kingdom.

A. Kirillov, B. Savchynskyy, C. Rother are with the Visual Learning Lab Heidelberg University, Germany.

F. Kahl and M. Larsson are with the Department of Signals and Systems, Chalmers University of Technology.

Manuscript received Oct 10, 2017.

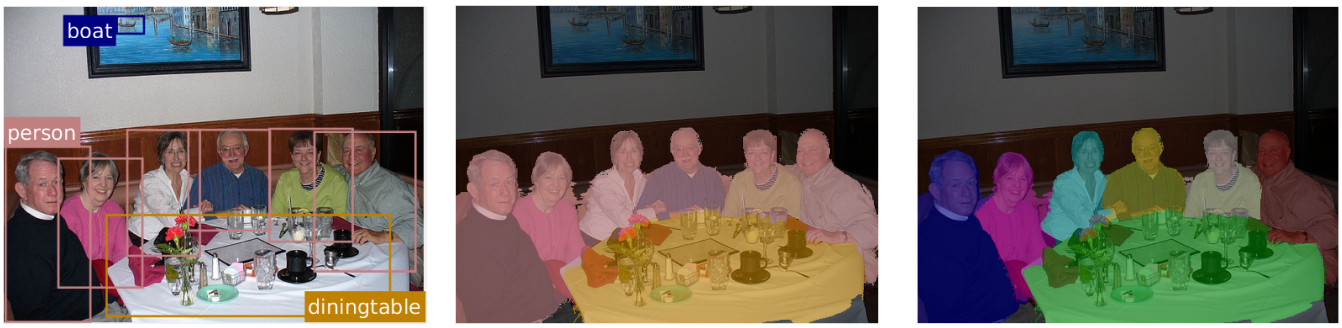


Fig. 1: **Example of various Scene Understanding tasks**. Some tasks, such as image classification, provide a high-level description of the image by classifying whether certain tags exist. Other tasks like object detection, semantic segmentation and instance segmentation provide more detailed and localised information about the scene. Researchers have also begun to bridge the gap between natural language processing and computer vision with tasks such as image captioning and visual question-answering.

feature representations automatically from data. It is therefore a natural idea to combine CRFs with neural networks in a joint framework, an approach which has been successful in a number of domains. From a theoretical perspective, it is interesting to investigate the connections between CRFs and DNNs, and to explore how inference algorithms for probabilistic graphical models can be framed as neural networks themselves. We review CRFs, DNNs and their integration in the rest of this paper, which is organised as follows:

- Section II reviews Conditional Random Fields along with their applications and history in Semantic Segmentation.
- Section III discusses the use of Fully Convolutional Networks for dense, pixel-wise prediction tasks such as Semantic Segmentation.
- Section IV describes how mean-field inference on a CRF, with a particular form of potential function, can be embedded into the neural network itself. In more general terms, this is an example of how iterative algorithms can be represented as neural networks, with similar ideas recently being applied to other domains as well.
- Section V describes methods which are able to learn more complex potential functions in CRFs embedded in neural networks, allowing them to capture more complex relationships in data.
- We then conclude in Section VII with unsolved problems and possible future directions in the area of scene understanding.

II. Conditional Random Fields

A naïve way of performing dense prediction tasks like semantic segmentation is to classify each pixel independently using some features derived from the image. However, such independent pixel-wise classification often produces unsatisfactory results that are inconsistent with the visual features in the image. For example, an independent pixel-wise classification can predict a few spurious incorrect labels in the middle of a blob of pixels that are classified to have the same label (*e.g.* a few "dog" pixels in the middle of a blob that is classified as "cat", as shown in Fig. 2). In order to predict the label of each pixel, a local classifier uses a small spatial context in the image (as shown by the patch in Fig 2c), and this often leads to noisy predictions. Better results can be obtained by acknowledging that we are predicting a structured output and explicitly modelling the problem to include our prior knowledge about a good pixel-wise prediction result. For example, we know that objects are usually continuous and thus we expect nearby pixels to be assigned the same object label. Conditional Random Fields (CRFs) are models that are widely used to achieve this. In the following, we provide a tutorial introduction to CRFs in the semantic image segmentation setting.

Conditional Random Fields are a classical tool for modelling complex structures consisting of a large number of interrelated parts. In the above example of image segmentation, these parts correspond to separate pixels. Each pixel u is associated with a finite set of its possible states $L = \{l_1, l_2, \ldots, l_L\}$, modelled by a variable $X_u \in L$. In the example in Fig. 2, these finite states are the labels that can be assigned to each pixel, i.e. $L = \{\text{person, cat, dog, background}\}$. Each state has an associated *unary* cost $\psi_u(X_u = x|I)$, which has to be paid to assign label x to the pixel u, given the image I. This unary cost is typically obtained from a classifier, as described in the previous paragraph, and with only the

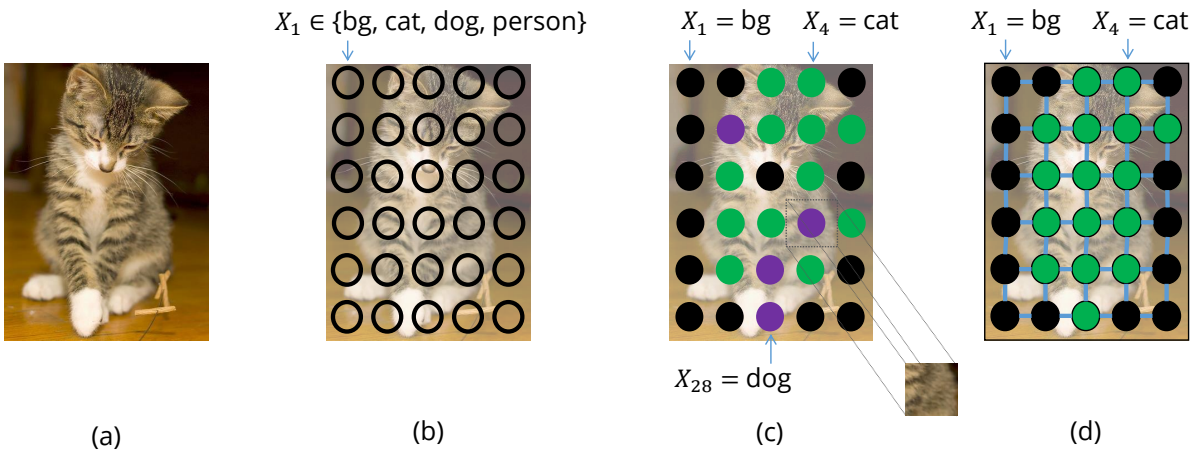


Fig. 2: **Overview of Semantic Segmentation.** Every pixel u is associated with a random variable X_u which takes on a label from a pre-defined label set (b). A naïve method of performing this would be to train a classifier to predict the semantic label of each pixel, using features derived from the image. However, as we can see from (c), this tends to result in very noisy segmentations. This is because in order to predict a pixel, the classifier would only have access to local pixel features, which are often not discriminative enough. By taking the relationship between different pixels into account (such as the fact that if a pixel has been labelled "cat", nearby ones are likely to take the same label since cats are continuous objects) with a CRF, we can improve our labelling (d).

unary cost, we would be performing independent, per-pixel predictions. To model interactions between pixels, pairwise costs are introduced. The value $\psi_{u,v}(X_u = x, X_v = y|I)$ is the *pairwise* cost for assigning a pair of labels x and y to the pixels u and v respectively. In semantic segmentation, a common pairwise cost to use is the Potts model (derived from statistical mechanics) where the cost is 0 when two neighbouring pixels have the same label, and λ (a real, positive scalar) if they have different labels. This cost encourages nearby pixels to take on the same label, and is based on our prior knowledge that objects are generally continuous. This pairwise weight, λ, typically depends on other features in the image – for example, the cost is higher if pixels are closer to each other in terms of spatial coordinates or if the appearance of two pixels is similar (by comparing pixel intensity values in an appropriate colour space). Costs can also be defined over more than two simultaneously interacting variables, in which case they are known as *higher order potentials*. However, we ignore these in the rest of this section for simplicity.

In terms of graph theory, a CRF can be understood as a graph $(V, \mathcal{E})$, with nodes V corresponding to the image pixels, and edges $\mathcal{E}$ connecting those node pairs for which a pairwise cost is defined. The following graphs are common in segmentation literature: (a) 4- or 8-grid graphs (which we will denote as "Grid CRF"), where only neighbouring pixels in the image are connected by graph edges and (b) fully connected graphs, where all pairs of pixels are connected by edges (Row 2 and 3 of Fig 3). Intuitively, grid graphs (such as the 4-grid graph in Fig. 2) can only propagate information to a limited number of neighbours. By contrast, fully-connected graphs enable long-range interactions between pixels which can lead to more precise segmentations as shown in Fig. 3. However, grid graphs were traditionally favoured in segmentation systems [10], [3], [9] since there exist efficient inference algorithms to solve the corresponding segmentation problem.

Let X_u be the variable associated with the node $u \in V$ and let $\mathbf{X}$ be the vector formed by the X_u variables under some ordering of V. An assignment $\mathbf{x}$ to $\mathbf{X}$ is known as a *configuration* or a *labelling*, i.e., a configuration assigns a label to each node in the CRF. *Inference* of the CRF involves finding a configuration $\mathbf{x}$, such that the total unary and pairwise costs, also called the *energy*, are minimised. The corresponding problem is called *energy minimization for CRFs*, where the energy is given by:

$$E(\mathbf{x}, I) := \sum_{u \in V} \psi_u(X_u = x_u|I) + \sum_{\{u,v\} \in \mathcal{E}} \psi_{u,v}(X_u = x_u, X_v = x_v|I). \quad (1)$$

Although the energy minimization problem is NP-hard [11], a number of exact and approximate algorithms exist to obtain acceptable solutions (see [12] for an overview). Exact algorithms typically apply to only special cases of the energy, whilst approximate algorithms efficiently find a solution to a simplification of the original problem. In particular, the most popular methods for image segmentation were initially based on the reduction of the energy minimization problem or its parts to the st-min-cut problem [13]. However, the complexity of these algorithms grow as the graph becomes more dense.

By contrast, fully-connected graphs with a specific type of pairwise cost can be efficiently (albeit approximately) addressed by *mean-field* algorithms, as detailed in Section II-A.

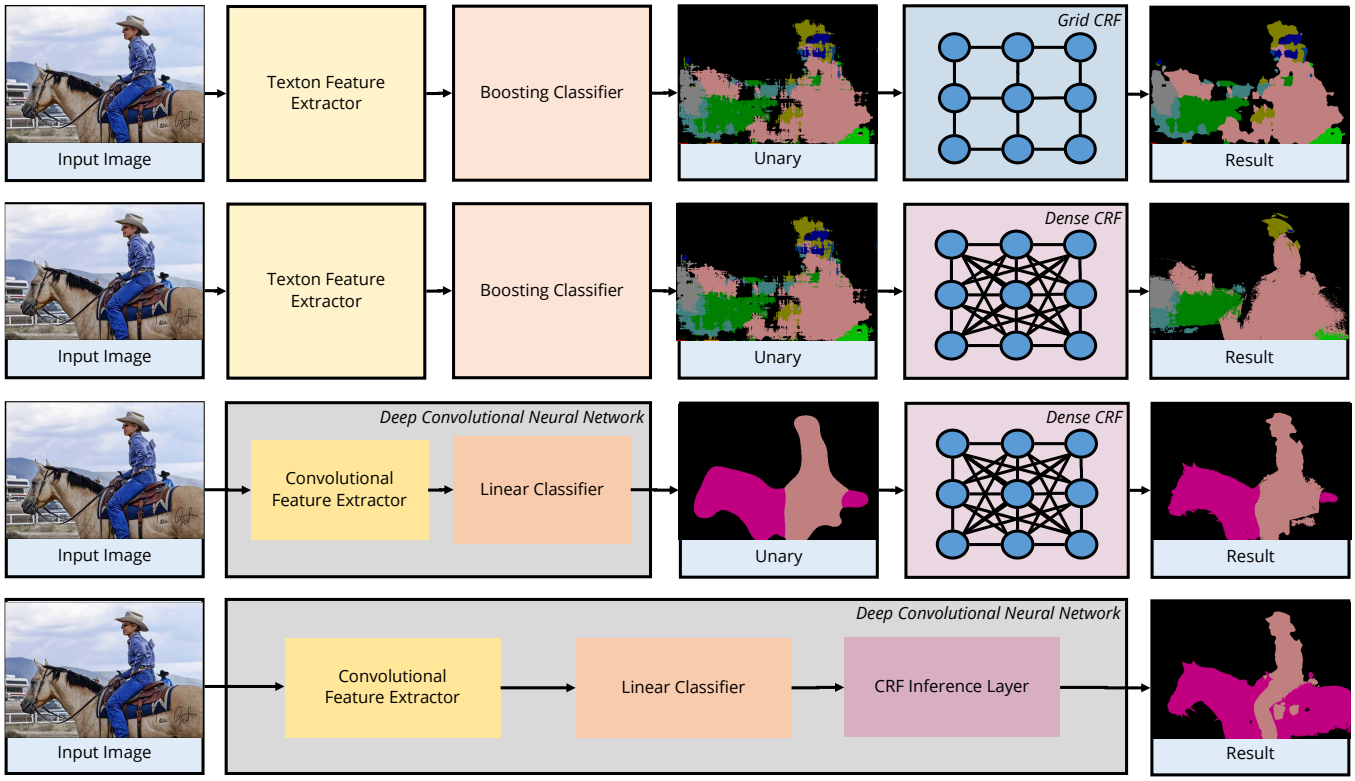


Fig. 3: **The evolution of Semantic Segmentation systems**. The first row shows the early "TextonBoost" work [3] that computed unary potentials using Texton [3] features, and a CRF with limited 8-connectivity. The DenseCRF work of Krahenbuhl and Koltun [4] used densely connected pairwise potentials and approximate mean-field inference. The more expressive model achieved significantly improved performance. Numerous works, such as [5] have replaced the early hand-crafted features with Deep Neural Networks which can learn features from large amounts of data, and used the outputs of a CNN as the unary potentials of a DenseCRF model. In fact, works such as [6] showed that unary potentials from CNNs (without any CRF) on their own achieved significantly better results than previous methods. Current state-of-the-art methods have all combined inference of a CRF within the deep neural network itself, allowing for end-to-end training [7], [8]. Result images for this figure were obtained using the publicly available code of [9], [4], [5], [7], [6].

These fully-connected graphs are now the most common model in segmentation.

The *learning* problem involves estimating cost functions based on a *training set* $(I^{(k)}, \mathbf{x}^{(k)})_{k=1}^{n}$ consisting of n pairs of images and corresponding *ground truth* labellings. The costs must be designed in a way that the inference performed for the image I returns a labelling that is close to the ground truth $\mathbf{x}$. Since statistical parameter estimation builds a basis for the learning theory, one defines a probability distribution $p(\mathbf{x}|I) = \frac{1}{Z(I)} \exp\{-E(\mathbf{x}, I)\}$ on the set of labellings. Here $Z(I) = \sum_{\mathbf{x}} \exp\{-E(\mathbf{x}, I)\}$ is a normalization factor known as the *partition function*, which ensures that the distribution sums to 1 (to make it a probability distribution).

A popular learning problem formulation is based on the maximum likelihood principle and consists in finding costs ψ_u and $\psi_{u,v}$ that maximize the probability of the training set $(I^{(k)}, \mathbf{x}^{(k)})_{k=1}^{n}$. The crucial subproblem, which determines the computational complexity of such estimation, consists in computing *marginal probabilities* for each label x_u in each node u and the pair (x_u, x_v) of labels in each pair of nodes (u, v), connected by an edge in the associated graph. The marginal probabilities build sufficient statistics for the distribution p and therefore need to be computed to perform learning. To compute the marginal probability $p_u(x_u)$ for the label x_u in the graph node u, one has to perform the summation $\sum_{\mathbf{x}' :\, x'_u = x_u} p(\mathbf{x}')$ over *all* possible labellings where node u takes on label x_u Combinatorial complexity of this problem arises, because the number of such labellings is exponentially large and the explicit summation is typically computationally infeasible. Moreover, it is a well-known result [14] stating that this problem is #P-hard, or, loosely speaking, there is no hope for a reasonably fast algorithm being able to perform such computations in general.

A number of approaches have been proposed to approximate these computations. Below we review the most popular ones:

A. Mean-field inference facilitating dense pairwise terms

Although computing the marginal probabilities is a hard problem in general, it can be easily done in certain special cases. In particular, it is an easy task if the graphical model only contains unary terms (and therefore no edges). In this case all marginal probabilities are simply inversely proportional to the unary costs. Distributions corresponding to such graphs without edges are called *fully factorized*. The *mean-field* approach approximates the distribution $p(\mathbf{x}|I)$ for a given image I with a fully factorized distribution, $Q(\mathbf{x})$. The approximation is done by minimising the Kullback-Leibler divergence between these two distributions. This minimization constitutes a non-convex problem, therefore only a local minimum is typically found by optimisation algorithms.

In practice, it is particularly important that there is an efficient algorithm to this end in the special case when the graphical model is associated with a fully-connected graph and pairwise costs $\psi_{uv}(x_u, x_v|I)$ for each pair of labels form a Gaussian distribution (up to a normalizing constant) for $x_u \neq x_v$ and equal to 0 otherwise. This model was first introduced by Krähenbühl and Koltun [4], and is known as DenseCRF. The pairwise costs were the sum of two Gaussian kernels: a Gaussian blurring filter, $\exp(-\frac{\|p_u-p_v\|^2}{2\sigma_\gamma^2})$, and an edge-preserving bilateral filter, $\exp(-\frac{\|p_u-p_v\|^2}{2\sigma_\alpha^2} - \frac{\|I_u-I_v\|^2}{2\sigma_\beta^2})$, where p_u and p_v denote the positions of pixels u and v, I_u and I_v the colour intensity vectors of these pixels, and σ the bandwidth of these filters. Although approximate mean-field inference was used, the model was more expressive compared to previous Grid CRFs and thus achieved significantly improved results with the faster runtime (facilitated by fast Gaussian filtering techniques [15]). As a result, this has become the de facto CRF model for most segmentation tasks, and has achieved the best performance on multiple public benchmarks. Section III describes how DenseCRF has been used in conjunction with CNNs where a CNN-based classifier provides the unary potentials. Section IV details how the mean-field inference algorithm for DenseCRF can be incorporated within the neural network itself.

B. Stochastic sampling-based estimation of marginals

Another approach for approximating marginal probabilities is based on sampling, in particular, on the Gibbs sampling method [16]. In this case, instead of computing the sum over all labellings, one samples such labellings and computes frequencies of all configurations in each node. The advantage of this approach is that in theory, the estimated marginals eventually converge to the true ones.

C. Variational approximations

The problem of computing marginals can be reformulated as a minimization problem [17]. Although this minimization problem is still as difficult as computing marginal probabilities, efficient approximations of the objective function exist. And this approximation of the objective function can be minimised quickly. In Section V we will get back to these approximations in the context of joint training of DNN and CRF models.

In this section, we have introduced CRFs, a common probabilistic graphical model used in semantic segmentation. The performance of these models are, however, influenced heavily by the unary term produced by the classifier. Convolutional Neural Networks have proved to be excellent classifiers in a variety of tasks, and we review their application to dense prediction tasks in the next section.

III. Convolutional Neural Networks for Dense Prediction

Deep Neural Networks have recently been shown to excel in a number of machine learning tasks, and most problems within the computer vision community are now solved by a class of these neural networks known as Convolutional Neural Networks (CNNs). A detailed overview of neural networks can be found in [18], with which we share our terminology. In contrast to previous classification algorithms, which require the user to manually design discriminative features, neural networks are able to automatically learn features from data when trained with Stochastic Gradient Descent (SGD) (or one of its variants) to minimise a training objective function. Whilst the backpropagation algorithm (a method for efficiently computing gradients of the parameters of a neural network with respect to an objective function, for use in conjunction with optimisation via SGD) for training neural networks has been used since the 1980's [19], it was the emergence of large-scale datasets such as ImageNet [20] and the parallel computational power of graphics processing units (GPUs) that have enabled neural networks to become very successful in most machine learning tasks. This became apparent to the computer vision community during the 2012 ImageNet challenge where the only entry using a neural network, AlexNet [21], achieved the best performance by a significant margin.

AlexNet [21], like many subsequent neural network architectures, is composed of a sequence of convolutional layers followed by ReLU non-linearities and pooling layers. A sequence of convolutional filters allows a neural network to learn hierarchical representations of images where complex patterns are composed of simpler patterns captured in earlier stages of the network. Convolutional layers are common in computer vision tasks since they preserve spatial information, and also since they are translationally equivariant – they have the same response at different parts of the image.

The last layers of successful CNN architectures [21], [22], [23] are typically inner-product or *fully-connected* layers (as common in traditional multi-layer perceptrons [24], [19] which used these layers throughout the network). These layers consider all features in the input to make the final prediction in the case of image classification, and the final fully-connected layer can be thought of as a linear classifier (operating on the non-linear features produced by preceding layers of the neural network). The final output of the network is a C-dimensional vector where C is the number of classes and each element in the vector represents the probability of each class appearing in the image. Hence, a typical CNN designed for

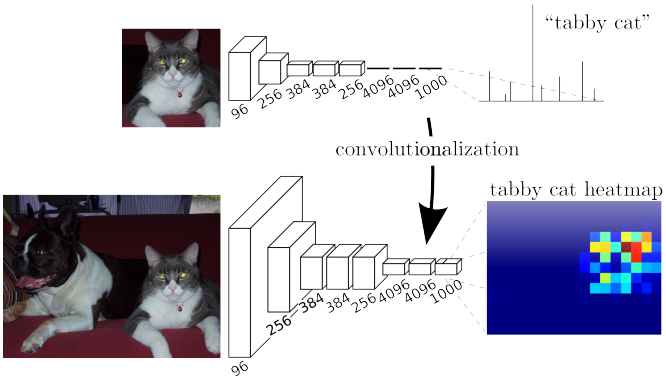


Fig. 4: **Fully convolutional networks.** Fully connected layers can easily be converted into convolutional layers by recognising that a fully-connected layer is simply a convolutional layer where the size of the convolutional filter and input feature map are identical. This enables CNNs trained for image classification to output a coarse segmentation when input a larger image. This simple method enables good initialisation and efficient training of CNNs for pixelwise prediction. (Figure from [6]).

image classification can be thought of as a function which maps an image of a fixed size to a C-dimensional vector of probabilities of each class appearing in the image.

Girshick *et al.* [25] showed that CNN architectures designed to excel in ImageNet classification could, with minimal modifications, be adapted for other scene understanding tasks such as object detection and semantic segmentation. Furthermore, it was possible for Girschick *et al.* to *fine-tune* their network from a network already trained on ImageNet since most of the layers were the same. Fine-tuning from an existing ImageNet-pretrained model provided better parameter-initialisation for training via backpropagation, and has been found to improve performance in many computer vision tasks. Therefore, the work of Girshick *et al.* suggested that CNNs for semantic segmentation should be based on ImageNet trained architectures as well.

A key idea to extending CNNs designed for image classification to other more complex tasks such as semantic segmentation is realising that a fully connected layer can be considered as a convolutional layer, where the filter size is the same as the size of the input feature map [26], [6]. Long *et al.* [6] converted the fully-connected layers of common architectures such as AlexNet [21] and VGG [22] into convolutional layers, and named these *Fully Convolutional Networks* (FCNs). Since these networks consist of only convolutional-, pooling- and ReLU non-linearity layers, they can operate on any arbitrarily sized image. However due to max-pooling in the network, the output would be a downsampled version of the input, as shown in Fig. 4. Common architectures such as AlexNet [21], VGG [22] and ResNet [23] all consist of five pooling layers of size 2x2, and hence the output is downsampled by a factor of 32 in these fully convolutional networks. Long *et al.* showed that even by simply bilinearly upsampling the coarse predictions up to the original size of the image, state-of-the-art performance at the time of publication could be achieved. This method is simple to implement, can be initialised with the parameters of a CNN trained on ImageNet, and then be fine-tuned on smaller datasets, which significantly improves results over initialising with random weights.

Although the fully-convolutional approach of Long *et al.* achieved state-of-the-art performance, the predictions of the model were still quite coarse and "blobby", since the max-pooling stages in earlier parts of the network resulted in a lot of spatial information being lost. As a result, fine structures and object boundaries were usually segmented poorly. This has led to a lot of follow-up work on improving the segmentation performance of neural networks.

Chen *et al.* [5] used the outputs of a CNN as the unary potentials of a DenseCRF model, and showed that applying a CRF as post-processing on these unaries could significantly improve results and provide sharper boundaries (as shown in Row 3 of Fig. 3). In fact, the absolute performance improvement from applying DenseCRF on CNN unaries was greater than that of Textonboost [3] unaries [5]. Other works have improved the CNN architecture by addressing the loss of resolution caused by max-pooling. It is not possible to completely remove max-pooling from a CNN architecture for segmentation, since it will mean that layers deeper down will not have sufficient context or *receptive field* to make a good prediction. To combat this issue, *Atrous* [5] or *Dilated* [27] convolutions have been proposed (inspired by the "algorithme à trous" used in computing the undecimated wavelet transform [28]), which enables the receptive field of a convolution filter to be increased without increasing the number of parameters in the filter. In these works, the last two max-pooling layers were removed, and Atrous convolutions were used thereafter to ensure a large receptive field. Note that it is not possible to remove all max-pooling layers in the network, due to the memory requirements of processing images at full resolution. Other works have learned more complex networks to upsample the low-resolution output of an FCN: In [29] an additional "decoder" network is learned which progressively "unpools" the initial prediction to obtain the final full-resolution output. Ghiasi and Fowlkes [30] learn the basis functions to upsample with in a coarse-to-fine architecture.

Although many architectural innovations have been proposed to improve the segmentation accuracy of neural networks , they have all benefited from additional refinement by a CRF. Furthermore, as Table I shows, algorithms which have achieved state-of-the-art results on public benchmarks such as Pascal VOC [31] have all incorporated CRFs as part of the neural network and trained it jointly with the unary part of the network end-to-end [32], [33], [8]. Similar trends are also being observed on the Cityscapes [34] and ADE20k [35] datasets which have been released in the last year. Intuitively, the improvement from these approaches stems from the fact that the parameters of the unary part of the network, and those of the CRF, may learn to optimally cooperate with each other.

The rest of this article focuses on these approaches which combine CRFs and CNNs in an end-to-end differentiable network: In Section IV, we elaborate on how mean-field inference

TABLE I: Results of recent algorithms on the Pascal VOC 2012 test set. Only the first submission, from 2012, does not use any deep learning. All the other methods use a base CNN architecture derived from an ImageNet pretrained network. Evaluation is performed by a public server on a withheld test-set. The performance metric is the Intersection over Union (IoU) [31].

Method	IoU [%]	Base Network
Methods not using deep learning		
O2P [36]	47.8	–
Methods not using a CRF		
SDS [37]	51.6	AlexNet
FCN [6]	67.2	VGG
Zoom-out [38]	69.6	VGG
Methods using CRF for post-processing		
DeepLab [5]	71.6	VGG
EdgeNet [39]	73.6	VGG
BoxSup [40]	75.2	VGG
Dilated Conv [27]	75.3	VGG
Centrale Boundaries [41]	75.7	VGG
DeepLab Attention [42]	76.3	VGG
LRR [30]	79.3	ResNet
DeepLab v2 [43]	79.7	ResNet
Methods with end-to-end CRFs		
CRF as RNN [7]	74.7	VGG
Deep Gaussian CRF [8]	75.5	VGG
Deep Parsing Network [44]	77.5	VGG
Context [32]	77.8	VGG
Higher Order CRF [33]	77.9	VGG
Deep Gaussian CRF [8]	80.2	ResNet

of CRFs can be unrolled and interpreted as a Recurrent Neural Network, and in Section V we describe other approaches which enable arbitrary potentials to be learned.

IV. Conditional Random Fields as Recurrent Neural Networks

Chen *et al.* [5] showed that state-of-the-art Semantic Segmentation results could be achieved by using the output of a fully convolutional network as the unary potentials of the DenseCRF model of [4]. However, the CRF was used as post-processing, and fully-convolutional network parameters were learnt by backpropagation whilst CRF parameters were cross-validated (the authors tried a large number of different CRF parameters, and finally selected those which gave the highest performance on a validation set).

This section details how mean-field inference of a DenseCRF model can be incorporated into the neural network itself, as a separate "mean-field inference module", an idea which was developed concurrently by Zheng *et al.* [7] and Schwing and Urtasun [45]. This enables joint training of both the CNN and CRF parameters by backpropagation. Intuitively, we can expect better results from this approach as the CNN and CRF learn parameters which are compatible with each other due to the joint training. The cross-validation strategy of other works, such as [5], cannot update the parameters of the CNN such that they are optimal for the chosen CRF parameters. Zheng *et al.* named their approach, "CRF-as-RNN", and this achieved the best results when that paper was published.

Mean-field is an iterative algorithm, and crucially for optimisation via SGD, the derivative of the output with respect to the input of each iteration can be calculated analytically. Therefore, we can unroll the inference algorithm across its time-steps, and form a Recurrent Neural Network (RNN) [18]. An RNN is a type of neural network, usually used to model sequential data, where the output of one iteration is used as the input of the next iteration and all iterations share the same parameters. In this case, the sequence is formed from the output of the iterative mean-field inference algorithm on each time step. When training the network, we can back-propagate through the RNN, and into the previous CNN to optimise all parameters jointly. Furthermore, as shown in [7] and described next, for the DenseCRF model, the inference algorithm turns out to consist of standard CNN operations, making its implementation simple and efficient in standard neural network libraries. In Sec. IV-B we describe how this idea can be extended beyond DenseCRF to other types of potentials, while Sec IV-E mentions how the idea of unrolling inference algorithms has subsequently been employed in other domains using deep learning.

A. CRF as RNN [7]

As mentioned in Sec. II-A, mean-field is an approximate inference method which approximates the true probability distribution, $P(\mathbf{X}|I)$, with a simpler distribution, $Q(\mathbf{X}|I)$. For notational simplicity, we omit the conditioning on the image, I, from here onwards. In mean-field, $Q(\mathbf{X})$ is assumed to be a product of independent marginals, $Q(\mathbf{X}) = \prod_u Q_u(X_u)$. The KL divergence between $P(\mathbf{X})$ and $Q(\mathbf{X})$ is then iteratively minimised. The Maximum a Posteriori (MAP) estimate of $P(\mathbf{X})$ is approximated as the MAP estimate of $Q(\mathbf{X})$. Since $Q(\mathbf{X})$ is fully-factorised, the MAP estimate is simply the label which maximises each independent marginal Q_u.

In the case of DenseCRF [4] (introduced in Sec. II-A), where the energy is of the form of Eq. 1, and the pairwise potentials are sums of Gaussian kernels,

$$\psi_{u,v}(x_u, x_v) = \mu(x_u, x_v)\, k(\mathbf{f}_u, \mathbf{f}_v)$$

$$k(\mathbf{f}_u, \mathbf{f}_v) = w^{(1)} \exp\left(-\frac{\|p_u - p_v\|^2}{2\sigma_\gamma^2}\right) + w^{(2)} \exp\left(-\frac{\|p_u - p_v\|^2}{2\sigma_\alpha^2} - \frac{\|I_u - I_v\|^2}{2\sigma_\beta^2}\right), \tag{2}$$

the mean-field update equations take the form:

$$Q_u(l) = \frac{1}{Z_u} \exp\{-\psi_u(l) - \sum_{l' \in L} \mu(l, l') \sum_{m=1}^{M} w^{(m)} \sum_{v \neq u} k^{(m)}(\mathbf{f_u}, \mathbf{f_v}) Q_v(l')\}. \tag{3}$$

The $\mu(\cdot, \cdot)$ function represents the compatibility of the labels assigned to variables X_u and X_v. In DenseCRF, the common Potts model (Sec. II) was used, where $\mu(x_u, x_v) = 0$ if $x_u =$

Algorithm 1 Mean field inference for Dense CRF [4], composed from common CNN operations.

$Q_u(l) \leftarrow \frac{1}{\sum_{l'} \exp(U_u(l'))} \exp\left(U_u(l)\right)$ ▷ Initialization

while not converged **do**

$\tilde{Q}_u^{(m)}(l) \leftarrow \sum_{v \neq u} k^{(m)}(\mathbf{f}_u, \mathbf{f}_v) Q_v(l)$ for all m ▷ Message Passing

$\check{Q}_u(l) \leftarrow \sum_m w^{(m)} \tilde{Q}_u^{(m)}(l)$ ▷ Weighting Filter Outputs

$\hat{Q}_u(l) \leftarrow \sum_{l' \in L} \mu(l, l') \check{Q}_u(l')$ ▷ Compatibility Transform

$\check{Q}_u(l) \leftarrow U_u(l) - \hat{Q}_u(l)$ ▷ Adding Unary Potentials

$Q_u(l) \leftarrow \frac{1}{\sum_{l'} \exp\left(\check{Q}_u(l')\right)} \exp\left(\check{Q}_u(l)\right)$ ▷ Normalizing

end while

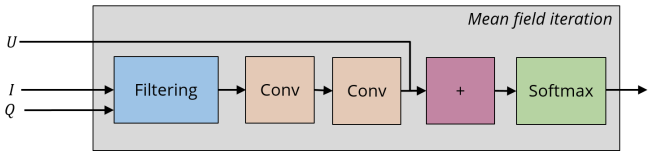


Fig. 5: **A mean-field iteration expressed as a sequence of common CNN operations.** The update equation of mean field inference of a DenseCRF model (Eq. 3), can be broken down into a series of smaller steps, as shown in Algorithm 1. Note that not only are these steps all differentiable, but they are all standard neural network operations as well. This allows a single mean-field iteration to be efficiently implemented as a neural network.

x_v and 1 otherwise. The DenseCRF model has parallels with Convolutional Neural Networks: In DenseCRF with Gaussian pairwise potentials [4], a Gaussian blurring filter, and an edge-preserving bilateral filter are used to compute the pairwise term. The coefficients of the bilateral filter depend on the image itself (pixel intensity values), which differs from a convolution layer in a CNN where the weights are fixed after training. Moreover, although the filter can potentially be as large as the image, it is parameterised only by its bandwidth. The pairwise potential is further parameterised by the weights of each filter, $w^{(1)}$ and $w^{(2)}$, and the label compatibility function, $\mu(\cdot, \cdot)$, which are both learned in the framework of [7].

Algorithm 1 shows how we can break this update equation down into simpler steps [7]. Moreover, we can see that these steps all consist of common CNN operations, and are all differentiable. Therefore, the authors were able to backpropagate through the mean-field inference algorithm, and into the original CNN. This allowed them to jointly optimise the parameters of the CNN and the CRF and achieve the best-published results on the VOC dataset at the time.

a) Expressing a mean-field iteration as a sequence of standard neural network operations: The "Message Passing" step, which involves filtering the approximated marginals, Q, can be computed efficiently using fast-filtering techniques that are common in signal processing literature [15]. This was leveraged by [7] to reduce the computational complexity of this step from $O(N^2)$ (the complexity of a naïve implementation of the bilateral filter where N is the number of pixels in the image) to $O(N)$. Computing the gradient of the output of the filtering step with respect to its input can also be performed using similar techniques. The next two steps, "Weighting Filter Outputs" and the "Compatibility Transform" can both be viewed as convolutions with a 1×1 kernel. In both cases, the parameters of these two steps were learnt as the neural network was trained. The addition step is another common operation that is trivial to implement in a neural network. Finally, note that both the "Normalising" and "Initialization" steps are equivalent to applying a softmax operation. This operation is ubiquitous in neural network literature as it is the activation function used in the multinomial logistic regression.

The fact that Zheng *et al.* [7] viewed the "Compatibility Transform" as a convolutional filter whose weights were learnt meant that they did not restrict themselves to the Potts model (Sec. II). This is in contrast to other methods such as [5] which cross-validated CRF parameters separately and assumed a Potts model, and is another reason for the improved performance of this method relative to the works published before it.

b) Mean-field inference as a Recurrent Neural Network: One iteration of the mean-field algorithm can be formulated as sequence of common CNN layers as shown in Fig. 5. By performing multiple mean-field iterations with the output of one iteration becoming the input of the next iteration, the mean-field inference algorithm can be formulated as a Recurrent Neural Network, as shown in Fig. IV-A. If we denote the unary potentials as U (the output of the initial CNN), then one mean-field iteration can be expressed as $Q^{t+1} = f_{\boldsymbol{\theta}}(U, Q^t, I)$ where Q^t are the current estimation of the marginal probabilities and I is the image. The vector, $\boldsymbol{\theta}$ denotes the parameters of the mean-field iteration which are shared among all iterations. In the case of Zheng *et al.* , they were the weights for the filter outputs, w, and the compatibility transform, $\mu(\cdot, \cdot)$ represented as a convolutional layer.

Q^0 is initialised as the softmax-normalised unary potentials (log probabilities) output by the initial CNN. Following the original DenseCRF work [4], Zheng *et al.* [7], computed a fixed number, T of mean-field iterations. Thus the final output of the module can be read off as Q^T. In practice, $T = 5$ iterations were used by [7] as it was empirically observed that mean-field had converged at this time. Recurrent Neural Networks are known to be susceptible to the vanishing/exploding gradients problem [18]: computing the gradient of the output of one iteration with respect to its input requires multiplying by the parameters being learned. This repeated multiplication can cause the gradients to explode (if the parameters are greater than 1) or vanish (if the parameters are less than 1). However, the fact that only five iterations need to be performed in practice means that this problem is averted.

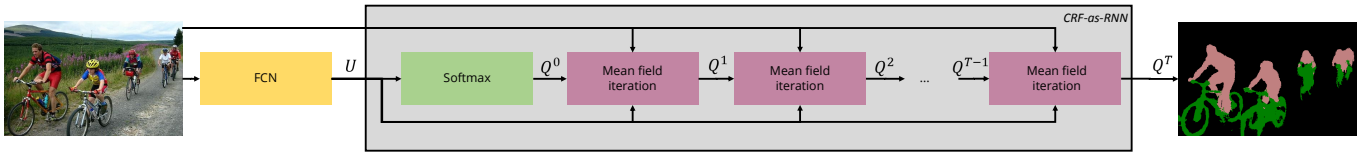
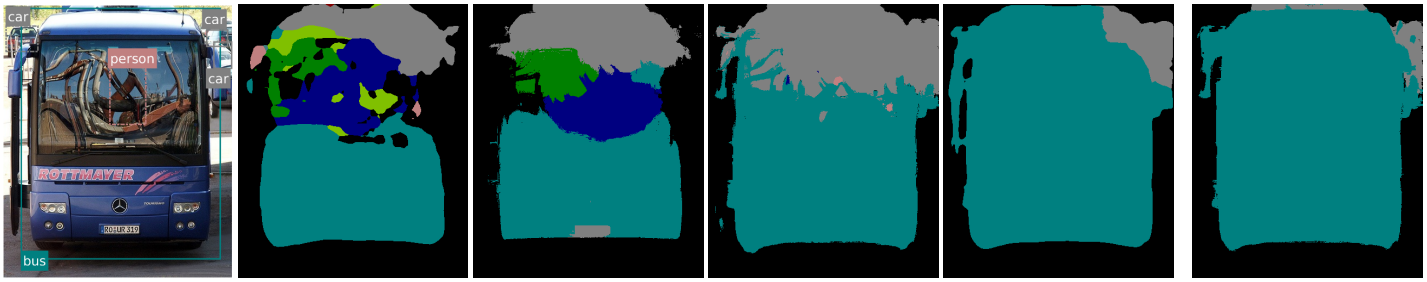


Fig. 6: **The final end-to-end trainable network of Zheng *et al.* [7]**. The final system consists of a Fully Convolutional Network (FCN) [6] followed by a CRF. The authors showed that the iterative mean-field inference algorithm could be unrolled and seen as a Recurrent Neural Network (RNN). This CRF inference module was named "CRF-as-RNN".



Input FCN [6] DeepLab [5] CRF-as-RNN [7] DPN [44] Higher Order CRF [33]

Fig. 7: **Comparison of various Semantic Segmentation methods**. FCN tends to produce "blobby" outputs which do not respect the edges of the image (from the Pascal VOC validation set). DeepLab, which refines outputs of a fully-convolutional network with DenseCRF produces an output which is consistent with edges in the image. CRF-RNN and DPN both train a CRF jointly within a neural network; achieving better results than DeepLab. Unlike other methods, the Higher Order CRF can recover from incorrect segmentation unaries since it also uses cues from an external object detector, while being robust to false-positive detections (like the incorrect "person" detection). Object detections produced by [46] have been overlaid on the input image, but only [33] uses this information.

As shown in Fig. 6, the final network implemented by Zheng *et al.* consists of a fully convolutional network [6], which predicts pixel-level labels without considering the structure of the output variables, followed by a CRF which can be trained end-to-end. The complete system therefore unites the strengths of CNNs – which can learn rich feature representations automatically from data – and CRFs – which can model the structure and correlations between the variables that are being predicted. Furthermore, parameters of both the CNN and CRF can be learned end-to-end via backpropagation.

In practice, Zheng *et al.* first trained a fully convolutional network (specifically the FCN8-s architecture of [6]) for semantic segmentation using the standard softmax cross-entropy loss. They then inserted the CRF-as-RNN layer and continued training their network, since it is necessary for the FCN part of the model to be initialised well before training the CRF. As shown in Table I, the CRF-as-RNN method outperformed DeepLab [47] (which also uses a fully convolutional network, but uses the CRF as a post-processing step) by 2%. In their paper, Zheng *et al.* [7] reported an improvement over just the fully convolutional network of 5.1%.

B. Incorporating Higher Order potentials

The CRF-RNN framework considered the particular case of unrolling mean-field inference as an RNN for the DenseCRF model. Arnab *et al.* [33] showed that this framework could be extended to different types of Higher Order potentials as well. Higher Order potentials (as mentioned in Section II), model correlations between cliques of pixels larger than two pixels as in DenseCRF.

Arnab *et al.* [33] considered two different higher order potentials: Firstly, a detection potential was formulated which encouraged consistency between the outputs of an object detector (*i. e.* Fig 1) and the final segmentation. This helped in cases where the segmentation unaries were poor and missed an object, but a complementary object detector had not. The potential was also formulated such that false detections could be ignored. A second potential was based on superpixels (a grouping of pixels into perceptually similar units, using specialised algorithms) and encouraged consistency over larger regions, helping to clean up spurious noise in the output. The mean-field updates for these more complex potentials were still differentiable, although they were no longer consisted of commonly-used neural network operations.

Higher Order potentials were shown to be effective in improving semantic segmentation performance before the adoption of deep learning [48]. In fact, potentials based on object detectors [49] and superpixels [50] had been proposed previously. However, by learning the parameters of these potentials jointly with the weights of an FCN, Arnab *et al.* [33] reduced the error of CRF-as-RNN by 12.6% on the VOC benchmark (Tab I).

TABLE II: Comparison of mean IoU (%) obtained on VOC 2012 reduced validation set from end-to-end and disjoint training (adapted from [33]).

Method	Mean IoU [%]
Unary only	68.3
Pairwise CRF trained disjointly	69.5
Pairwise CRF trained end-to-end	72.9
Higher Order CRF trained disjointly	73.6
Higher Order CRF trained end-to-end	75.8

Liu *et al.* [44] formulated another approach of incorporating higher order relations. Whilst [7] and [33] performed exact mean-field inference of a CRF as an RNN, Liu *et al.* approximated a single iteration of mean-field inference with a number of carefully designed convolution layers. Their "Deep Parsing Network" (DPN) was also trained end-to-end, although like [7], they first pre-trained the initial part of their network before finally training the entire network. As shown in Tab. I, this approach achieved very competitive results similar to [33]. Fig. 7 shows a comparison between FCN, CRF-as-RNN, DPN and the Higher Order CRF on a common image.

C. Benefits of end-to-end training

An alternative to unrolling mean-field inference of a CRF and training the whole network end-to-end is to train only the "unary" part of the network and use the CRF as a post-processing step whose parameters are determined via cross-validation. We refer to this approach as "disjoint" training of the CNN and the CRF, and show in Table II that end-to-end training outperforms disjoint training in the case of CRF-RNN [7], which only has pairwise potentials, and the Higher Order CRF of [33].

Many recent works in the literature have used CRFs as a post-processing step. However, as shown in Table I, the best performing ones have incorporated the CRF as part of the network itself. Intuitively, joint training of a CRF with a CNN allows the two modules to learn to optimally co-operate with each other.

D. Error analysis

It can be seen visually from Figures 3 and 7 that the densely-connected pairwise potentials of a CRF improve the segmentation quality at boundaries of objects. To quantify the improvements that CRFs make to the overall segmentation, we separately evaluate segmentation performance on the "boundary" and "interior" regions of the image, as done by [40] and [33]. As shown in Fig. 8c) and d), we consider a narrow band (trimap [50]) around the "void" labels annotated in the VOC 2012 reduced validation set. The mean IoU of pixels lying within this band is termed the "Boundary IoU" whilst the "Interior IoU" is evaluated outside this region.

Fig. 8 shows our results as the trimap width is varied for FCN [6], CRF-as-RNN [7], DPN [44] and Higher Order CRF [33]. We can see that all the CRF models improve the Boundary IoU over FCN significantly, although CRF-as-RNN and Higher Order CRF are almost identical. Moreover, all the methods incorporating CRFs also show an improvement in the Interior IoU as well, indicating that the spatial and appearance consistency encouraged by CRFs is not limited to only improving segmentation at object boundaries. Furthermore, the higher order potentials of [33] and [44] show a substantial increase in Interior IoU over CRF-as-RNN, and an even bigger improvement over FCN. Higher order potentials encourage consistency over larger regions of the image [33] and model contextual relationships between object classes [44]. As a result, they show larger improvements at the interior of objects being segmented.

E. Other examples of unrolling inference algorithms in neural networks

Unrolling inference algorithms as neural networks is a powerful idea beyond semantic image segmentation. Riegler *et al.* [51] presented a method for depth-map super-resolution by unrolling the steps of an optimisation algorithm and formulating it as an end-to-end trainable network. Recently, Wang *et al.* [52] extended deep structured models to continuous valued output variables, and addressed tasks such as image denoising and depth refinement.

V. Learning Arbitrary Potentials in CRFs

As the previous section shows, the joint training of a CNN and a CRF with Gaussian potentials is beneficial for the semantic segmentation problem. While it is able to output spatially- and appearance-consistent results, some applications may benefit even more from the usage of more generic potentials. Indeed, general potentials are able to incorporate more sophisticated knowledge about a problem than are Gaussian potentials. As an example, for the human body part segmentation problem, parametric potentials may enforce a high-level structural constraint, i.e. head should be located above torso as seen in Fig. 9a. Moreover, these potentials can acquire this knowledge automatically during training. And by training a pixel-level CNN jointly with these potentials, a synergistic effect can be obtained as observed in Sec. IV.

As mentioned in Section II, the main computational burden for probabilistic CNN-CRF learning consists in estimating marginal distributions for the configurations of individual variables and their pairs. Therefore, existing methods can be classified according to how they approximate these marginals. Each of the methods has different advantages and disadvantages and has a specific scope of problems where it is superior to the other. Below we briefly review three such methods, that are applicable to learn arbitrary pairwise and unary costs. Finally, we also describe methods that are able to learn arbitrary pairwise costs without computing marginals.

A. Stochastic sampling-based training

This approach to learning is based on the stochastic estimation of marginal probability distributions, as described in Section II-B. This method for training CNN-CRF models was proposed in [53], and applied to the problem of human

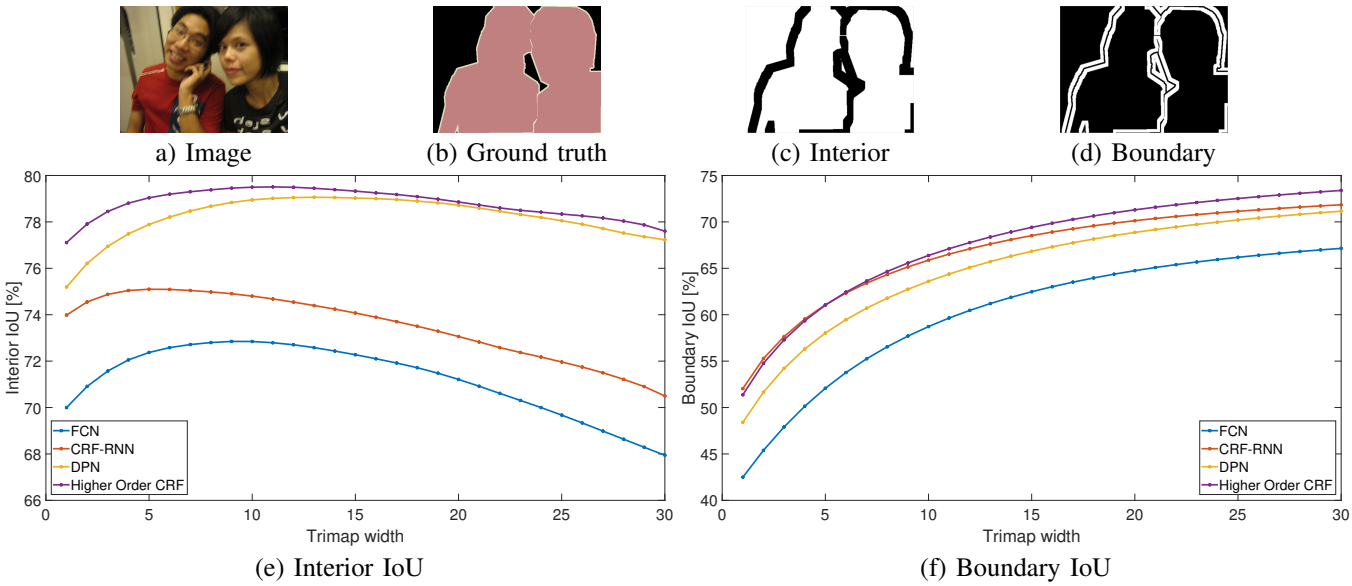


Fig. 8: **Error analysis on VOC 2012 reduced validation set.** The IoU is computed for boundary and interior regions for various trimap widths. An example of the Boundary and Interior regions for a sample image using a width of 9 pixels is shown in white in the top row. Black regions are ignored in the IoU calculation (adapted from [33])

.

body-part segmentation where it was shown to outperform techniques based on DenseCRF as displayed in Fig. 9. Advantages of this method are that it is simple, and easy to parallelize. Along with a long training time, the disadvantages include the necessity to use very slow sampling-based energy minimization methods [16] for image segmentation during the prediction stage, when the neural network has already been trained. The slow prediction time is due to the fact that the best predictive accuracy is obtained when the training and prediction procedures are identical.

B. Piece-wise training

For estimating marginal probabilities, this method replaces the marginal probabilities $p_v(x_v)$ (see Section II for details) with values equal to the corresponding exponentiated costs $\exp\{-\psi_v(x_v)\}$ up to a normalizing constant. Although such a replacement is not grounded theoretically, it allows one to train parameters for each node and edge of the graph independently, which is computationally very efficient and easily parallelizable. Despite the lack of theoretical justification, the method was employed by Lin *et al.* [32] to give good practical results (it was on top of the Pascal VOC and Cityscapes leaderboards for several months) on numerous segmentation datasets, as reflected by the `Context` entry in Tab. I.

C. Variational method based training

As referenced in Section II-C, the marginal probabilities were approximated with a variational technique during learning in [47]. In terms of theoretical justification the method can be positioned between stochastic training, as the most theoretically justified, and piece-wise training, which lacks such a justification. Practically, the running time in the classification regime is far more acceptable than those of the stochastic method of [53]. However, the scalability of the method is questionable because of its large memory footprint: along with the training set one has to store a set of current values of the working variables. The size of this set is proportional to the number of training samples multiplied by (i) the number of nodes in a graphical model corresponding to each sample (which can be roughly estimated as a number of pixels in the corresponding image), (ii) the number of edges in each graphical model and (iii) the number of possible variable configurations, which can be assigned to each graph node. In total, this typically requires one or even two orders of magnitude more storage than the training set itself. So far, this method has not been shown on large training sets.

D. Learning by backpropagating through inference

Formulating the steps of CRF inference as differentiable operations enable both the learning and inference of the CRF to be incorporated in a standard deep learning framework, without having to explicitly compute marginals. An example of this is "CRF-as-RNN", previously detailed Sec. IV-A.

However, in contrast to that approach, we can look at the *inference* problem from a discrete optimisation point of view. Here, it can be seen as an integer program where a given cost should be minimised while satisfying a set of constraints (each pixel should be assigned one label). An alternative inference approach is to do a continuous relaxation (allowing variables to be real-valued instead of integers) of this integer

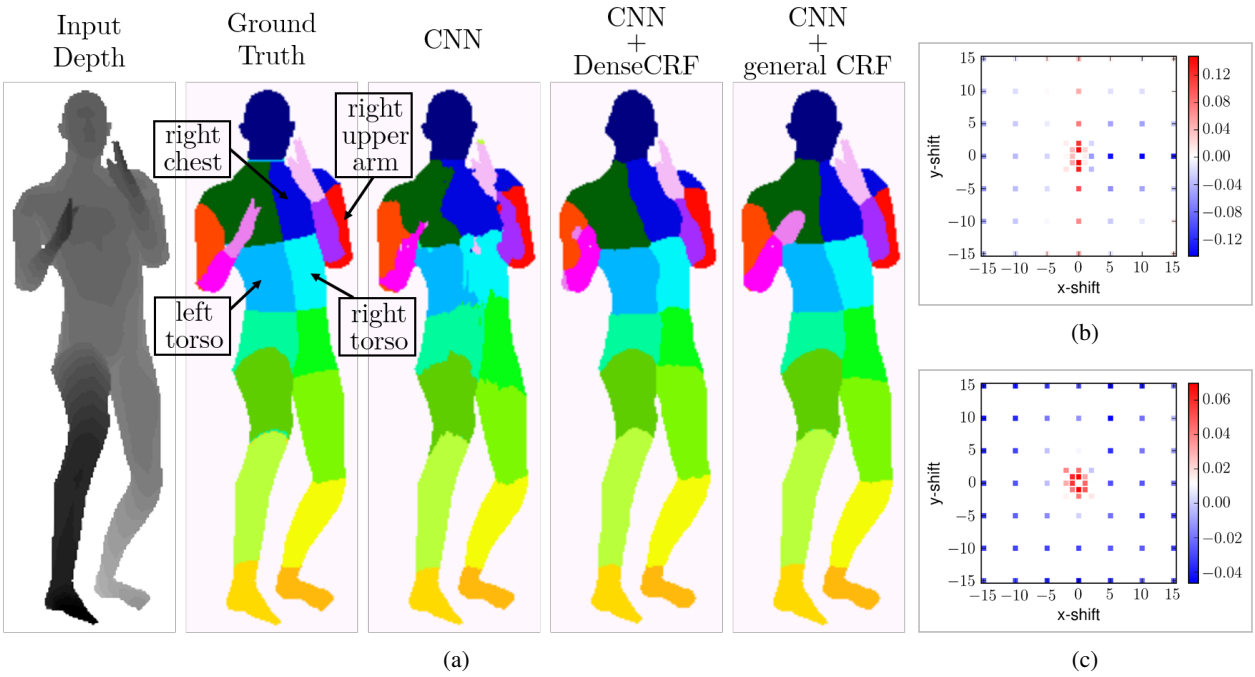


Fig. 9: **Human body parts segmentation with generic pairwise potentials.** (a) (From left to right). The input depth image. The corresponding ground truth labelling for all body parts. The result of a trained CNN model. The result of CRF-as-RNN [7] where the pairwise potentials are a sum of Gaussian kernels. The result of [53] that jointly train CNN and CRF with general parametric potentials. Note how the hands and elbows are segmented better. (b) Weights for pairwise potentials that connect the label "left torso" with the label "right torso". Red means a high energy value, i.e. a discouraged configuration, while blue means the opposite. The potentials enforce a straight, vertical border between the two labels, i.e. there is a large penalty for "left torso" on top (or below) of "right torso" (x-shift 0, y-shift arbitrary). Also, it is encouraged that "right torso" is to the right of the "left torso" (Positive x-shift and y-shift 0). (c) Weights for pairwise potentials that connect the label "right chest" with the label "right upper arm". It is discouraged that the "right upper arm" appears close to "right chest", but this configuration can occur at a certain distance. Since the training images have no preferred arm-chest configurations, all directions have similar weights. Such relations between different parts cannot by the Gaussian kernels used in CRF-as-RNN.

program and search for a feasible minimum. The solution to the original discrete problem can then be derived from the solution to the relaxed problem. Desmaison *et al.* [55] presented methods to solve several continuous relaxations of this problem. They showed that these methods generally achieve lower energies than mean-field based approaches. In the work of Larsson *et al.* [54], a projected gradient method based on only differential operations is presented. This inference method minimises a continuous relaxation of the CRF energy and the weights can be learned by backpropagating the error derivative through the gradient steps. An example of the pairwise potentials learned by this method is shown in Fig. 10. Chandra and Kokkinos [8] solve the energy minimization problem by formulating it as a convex quadratic program and finding the global minimum by solving a linear system. As seen in Tab. I, it is the best-published approach on the PASCAL VOC 2012 dataset at the time of writing.

E. Methods based on discriminative learning

There is another, discriminative learning technique, which does not require the computation of marginal distributions. This class of methods formulate the learning as a structured support vector machine (SSVM), which is an extension to the support vector machine allowing structured output. Since solving these usually requires doing inference for several setups of weights an efficient inference method is crucial. Larsson *et al.* [56] utilised this for medical image segmentation doing inference with a highly efficient graph-cut method. Knöbelreiter *et al.* [57] proposes a very efficient GPU-parallelized energy minimization solver to efficiently perform inference for the stereo problem. They show how learning can be made practically feasible by an SSVM formulation.

VI. Training Considerations

Optimising deep neural networks in general requires careful selection of training hyperparameters, most notably the learning rate. The networks described in this paper, which integrate CRFs and CNNs, are typically trained in a two-stage process [7], [8], [33], [44], [53], [54]: First, the "unary" part of the network is trained, and then the part of the network modelling inference of the CRF is appended and the network is trained end-to-end. It is not recommended to train from

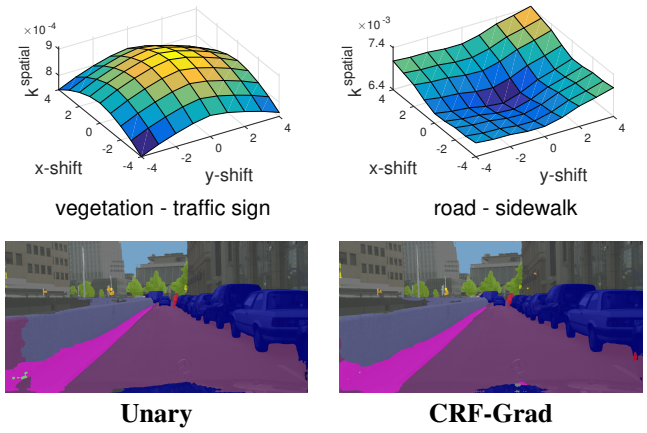


Fig. 10: **Pairwise potentials, represented as convolutional filters, learned by the method of [54].** These filters, shown on top, model contextual relationships between classes and their values can be understood as the energy added when setting one pixel to the first class (e.g., vegetation) and the other pixel with relative position (x-shift,y-shift) to the second class (e.g., traffic sign). The bottom row shows an example result on the Cityscapes dataset. The traffic lights and poles (which are challenging due to their limited spatial extent) are segmented better, and it does not label "road" being on top of the "sidewalk", due to the modelling of contextual relationships between classes.

the beginning with the CRF inference layer as the unaries produced by the first stage of the network are so poor that performing inference on them produces meaningless results whilst also increasing the computational time. An exception is Lin *et al.* [32] who train their entire network end-to-end. However, they have carefully chosen learning rate multipliers for different parts of their network. Liu *et al.* [44], on the other hand, initially train the unary part of the network and then have three separate training stages for each of the three modules in their network simulating mean-field inference of a CRF.

Another important training detail for these models is the selection of the initial unary network to finetune end-to-end with the CRF inference layer. In the case of [7], [33], [54], allowing the initial unary network to converge and finetuning off that model tends to not produce the best results. Instead, the best performance is usually obtained by finetuning from a unary network that has not fully converged, in the sense that its validation error has not yet completely plateaued. Our intuition about why this happens is that once the parameters of the unary network converge to a local optimum, it is difficult for the learning algorithm to get out of this region.

Since CRF-as-RNN [7] is an iterative method, it is also necessary to specify the number of mean-field iterations to perform. The authors used five iterations of mean-field for training, and then increased this to ten iterations at test time. This choice is motivated by Fig. 11 which shows empirical convergence results on the VOC 2012 reduced validation set – the majority of the improvement takes place after three

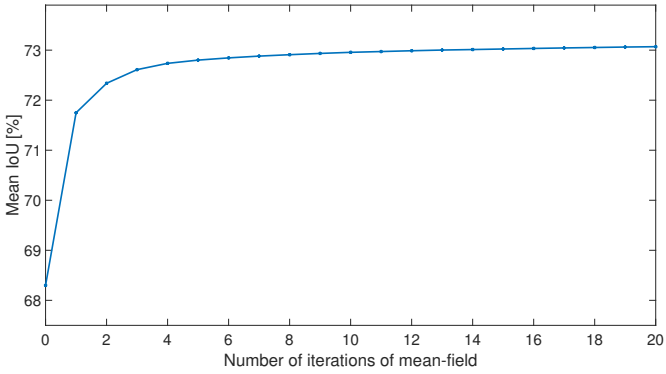


Fig. 11: **Empirical convergence of CRF-as-RNN [7]** The mean IoU on the Pascal VOC 2012 reduced validation set starts plateauing after five iterations of mean-field inference, supporting the authors' choice of training their network with five iterations of mean-field. The result at 0 iterations is the mean IoU of only the unary network.

iterations and the IoU begins to plateau after five iterations. Ten iterations are used at test time to obtain the best performance, whilst five iterations are used whilst training as these are enough iterations for the IoU to start plateauing and fewer iterations also decreases the training time.

VII. Conclusions and Future Directions

Conditional Random Fields (CRFs) have a long history in structured prediction tasks in computer vision, such as semantic segmentation. Deep Neural Networks (DNNs), on the other hand, have recently been shown to achieve outstanding results due to their ability to automatically learn features from large datasets. In this article, we have discussed various approaches of how CRFs have been combined with DNNs to tackle pixel-labelling tasks such as semantic segmentation. The most basic method is to simply use the outputs of a DNN as the unary potentials of a well studied CRF model (such as DenseCRF [4]) as a separate post-processing step *e.g.* [47]. We then discussed how the mean-field inference algorithm for CRFs could be formulated as a Recurrent Neural Network, and thus be incorporated as another module or "layer" of an existing neural network. This method, however, was limited to only pairwise potentials of a specific form. Finally, we described several approaches to learning arbitrary potentials in CRFs. We also noted that the idea of unrolling inference algorithms as neural networks has since been applied in other fields as well.

As neural networks are universal function approximators, it is possible that network architectures could be designed that do not require explicit CRFs to model smoothness priors and achieve the same performance. This, however, remains an open research question as our understanding of DNNs is still very limited. Moreover, it is not clear if the smoothness priors incorporated by a CRF could be modelled by generic neural network layers as efficiently (in terms of the number of parameters). It is also an open question as to whether we need

to develop more sophisticated training algorithms to achieve this.

The works described in this article have all been fully supervised learning scenarios, where large costs have been incurred in collecting datasets with per-pixel annotations. Given the substantial increase in segmentation performance on public benchmarks such as Pascal VOC [31], a future direction is to achieve similar accuracy levels using weakly supervised annotations (for example, image tags as annotation). In such scenarios with limited annotations, incorporating additional prior knowledge is of greater importance, and CRFs provide a method of doing so [58], [59].

Instance Segmentation (Fig 1) is another emerging area of scene understanding research, and early works have incorporated end-to-end CRFs within their systems [60].

For some tasks (such as face detection on cameras), rough bounding boxes suffice. However, pixel-level understanding of a scene is required for tasks such as autonomous vehicles and medical diagnosis where detailed information is required. Advances in pixel-level prediction, along with holistic models which address multiple scene understanding tasks, are bringing us closer to computers which understand our physical world and help enrich it.

ACKNOWLEDGMENT

This work was supported by grants ERC-2012-AdG 321162-HELIOS, ERC-647769, EPSRC Seebibyte EP/M013774/1, EPSRC/MURI EP/N019474/1, the Clarendon Fund, the Swedish Research Council (grant no. 2016-04445), the Swedish Foundation for Strategic Research (Semantic Mapping and Visual Navigation for Smart Robots) and Vinnova / FFI (Perceptron, grant no. 2017-01942)

REFERENCES

[1] M. Minsky and S. Papert, *Perceptrons: an introduction to computational geometry*. MIT Press, 1969.

[2] A. Blake, P. Kohli, and C. Rother, *Markov Random Fields for Vision and Image Processing*. MIT Press, 2011.

[3] J. Shotton, J. Winn, C. Rother, and A. Criminisi, "Textonboost for image understanding: Multiclass object recognition and segmentation by jointly modeling texture, layout, and context," *IJCV*, vol. 81, pp. 2–23, 2009.

[4] P. Krähenbühl and V. Koltun, "Efficient inference in fully connected crfs with gaussian edge potentials," in *NIPS*, 2011.

[5] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected crfs," in *ICLR*, 2015.

[6] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *IEEE CVPR*, 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 *IEEE ICCV*, 2015.

[8] S. Chandra and I. Kokkinos, "Fast, exact and multi-scale inference for semantic image segmentation with deep gaussian crfs," in *ECCV*, 2016, pp. 402–418.

[9] L. Ladicky, C. Russell, P. Kohli, and P. H. Torr, "Associative hierarchical crfs for object class image segmentation," in *IEEE ICCV*, 2009.

[10] X. He, R. Zemel, and M. Carreira-Perpinan, "Multiscale conditional random fields for image labeling," in *IEEE CVPR*, 2004.

[11] M. Li, A. Shekhovtsov, and D. Huber, "Complexity of discrete energy minimization problems," in *ECCV*, 2016, pp. 834–852.

[12] J. H. Kappes, B. Andres, F. A. Hamprecht, C. Schnörr, S. Nowozin, D. Batra, S. Kim, B. X. Kausler, T. Kröger, J. Lellmann, N. Komodakis, B. Savchynskyy, and C. Rother, "A comparative study of modern inference techniques for structured discrete energy minimization problems," *IJCV*, pp. 1–30, 2015.

[13] V. Kolmogorov and R. Zabin, "What energy functions can be minimized via graph cuts?" *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 2, pp. 147–159, 2004.

[14] A. Bulatov and M. Grohe, "The complexity of partition functions," *Theoretical Computer Science*, vol. 348, no. 2-3, pp. 148–186, 2005.

[15] A. Adams, J. Baek, and M. A. Davis, "Fast high-dimensional filtering using the permutohedral lattice," *Computer Graphics Forum*, vol. 29, no. 2, pp. 753–762, 2010.

[16] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," *IEEE Trans. Pattern Anal. Mach. Intell.*, no. 6, pp. 721–741, 1984.

[17] M. J. Wainwright, M. I. Jordan *et al.*, "Graphical models, exponential families, and variational inference," *Foundations and Trends® in Machine Learning*, vol. 1, no. 1–2, pp. 1–305, 2008.

[18] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT Press, 2016.

[19] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error-propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 1*. MIT Press, Cambridge, MA, 1986, vol. 1, no. 6088, pp. 318–362.

[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," *IJCV*, vol. 115, no. 3, pp. 211–252, 2015.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.

[22] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2015.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE CVPR*, 2016.

[24] F. Rosenblatt, "Principles of neurodynamics. perceptrons and the theory of brain mechanisms," DTIC Document, Tech. Rep., 1961.

[25] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *IEEE CVPR*, 2014.

[26] A. Giusti, D. C. Ciresan, J. Masci, L. M. Gambardella, and J. Schmidhuber, "Fast image scanning with deep max-pooling convolutional neural networks," in *IEEE ICIP*. IEEE, 2013, pp. 4034–4038.

[27] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *ICLR*, 2015.

[28] M. Holschneider, R. Kronland-Martinet, J. Morlet, and P. Tchamitchian, "A real-time algorithm for signal analysis with the help of the wavelet transform," in *Wavelets*. Springer, 1990, pp. 286–297.

[29] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *IEEE ICCV*, 2015.

[30] G. Ghiasi and C. C. Fowlkes, "Laplacian pyramid reconstruction and refinement for semantic segmentation," in *ECCV*. Springer, 2016, pp. 519–534.

[31] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The PASCAL visual object classes (VOC) challenge," *IJCV*, vol. 88, no. 2, pp. 303–338, 2010.

[32] G. Lin, C. Shen, A. van den Hengel, and I. D. Reid, "Exploring context with deep structured models for semantic segmentation," in *arXiv preprint arXiv:1603.03183*, 2016.

[33] A. Arnab, S. Jayasumana, S. Zheng, and P. Torr, "Higher order conditional random fields in deep neural networks," in *ECCV*, 2016.

[34] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *IEEE CVPR*, 2016.

[35] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Semantic understanding of scenes through the ade20k dataset," in *arXiv preprint arXiv:1608.05442*, 2016.

[36] J. Carreira, R. Caseiro, J. Batista, and C. Sminchisescu, "Free-form region description with second-order pooling," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2014.

[37] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, "Simultaneous detection and segmentation," in *ECCV*, 2014.

[38] M. Mostajabi, P. Yadollahpour, and G. Shakhnarovich, "Feedforward semantic segmentation with zoom-out features," in *IEEE CVPR*, 2015.

[39] 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 *IEEE CVPR*, 2016, pp. 4545–4554.

[40] J. Dai, K. He, and J. Sun, "Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation," in *IEEE ICCV*, 2015.

[41] I. Kokkinos, "Pushing the boundaries of boundary detection using deep learning," in *ICLR*, 2016.

[42] L.-C. Chen, Y. Yang, J. Wang, W. Xu, and A. L. Yuille, "Attention to scale: Scale-aware semantic image segmentation," in *IEEE CVPR*, 2016, pp. 3640–3649.

[43] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," in *arXiv preprint arXiv:1606.00915*, 2016.

[44] Z. Liu, X. Li, P. Luo, C.-C. Loy, and X. Tang, "Deep learning markov random field for semantic segmentation," in *IEEE Trans. Pattern Anal. Mach. Intell.*, 2017.

[45] A. G. Schwing and R. Urtasun, "Fully connected deep structured networks," in *arXiv:1503.02351*, 2015.

[46] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *NIPS*, 2015.

[47] L.-C. Chen, A. G. Schwing, A. L. Yuille, and R. Urtasun, "Learning deep structured models," in *ICML*, 2015.

[48] V. Vineet, J. Warrell, and P. H. Torr, "Filter-based mean-field inference for random fields with higher-order terms and product label-spaces," in *ECCV*, 2012, pp. 31–44.

[49] C. Wojek and B. Schiele, "A dynamic conditional random field model for joint labeling of object and scene classes," in *ECCV*. Springer, 2008, pp. 733–747.

[50] P. Kohli, L. Ladicky, and P. H. S. Torr, "Robust higher order potentials for enforcing label consistency," *IJCV*, vol. 82, no. 3, pp. 302–324, 2009.

[51] G. Riegler, M. Rüther, and H. Bischof, "Atgv-net: Accurate depth super-resolution," in *ECCV*. Springer, 2016, pp. 268–284.

[52] S. Wang, S. Fidler, and R. Urtasun, "Proximal deep structured models," in *NIPS*, 2016.

[53] A. Kirillov, D. Schlesinger, S. Zheng, B. Savchynskyy, P. Torr, and C. Rother, "Efficient likelihoood learning of a generic cnn-crf model for semantic segmentation," in *ACCV*, 2016.

[54] 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 *11th International Conference on Energy Minimization Methods in Computer Vision and Pattern Recognition, (EMMCVPR)*. Springer, 2017.

[55] A. Desmaison, R. Bunel, P. Kohli, P. H. Torr, and M. P. Kumar, "Efficient continuous relaxations for dense crf," in *ECCV*. Springer, 2016, pp. 818–833.

[56] M. Larsson, J. Alvén, and F. Kahl, "Max-margin learning of deep structured models for semantic segmentation," in *20th Scandinavian Conference Image Analysis, (SCIA 2017)*, 2017.

[57] P. Knöbelreiter, C. Reinbacher, A. Shekhovtsov, and T. Pock, "End-to-end training of hybrid CNN-CRF models for stereo," in *IEEE CVPR*, 2017.

[58] A. Kolesnikov and C. H. Lampert, "Seed, expand and constrain: Three principles for weakly-supervised image segmentation," in *ECCV*, 2016, pp. 695–711.

[59] 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 *ECCV*, 2016.

[60] A. Arnab and P. H. S. Torr, "Pixelwise instance segmentation with a dynamically instantiated network," in *IEEE CVPR*, 2017.

Anurag Arnab is a DPhil (PhD) student at the University of Oxford under the supervision of Prof. Philip H. S. Torr. His research interests are in deep learning and probabilistic graphical models, and their applications in Scene Understanding tasks such as semantic segmentation and instance segmentation.

Shuai Zheng did his Ph.D. degree at Oxford University under the guidance of Prof. Philip H. S. Torr. Formerly, he received his MEng from the Institute of Automation, Chinese Academy of Sciences, where he was supervised by Prof. Kaiqi Huang and Prof. Tieniu Tan. His research interests are in deep learning and its applications in computer vision such as semantic segmentation.

Sadeep Jayasuamana is a research scientist at Five AI, Cambridge, UK. Previously, he was a postdoctoral research fellow in the Torr Vision Group at the University of Oxford. He received his PhD from the Australian National University, under the supervision of Prof. Richard Hartley. Sadeep's research interests include deep learning, semantic segmentation, and machine learning on manifold-valued data.

Bernardino Romera-Paredes is a research scientist at Google DeepMind. He was a postdoctoral research fellow in the Torr Vision Group at the University of Oxford. He received his PhD degree from University College London in 2014, supervised by Prof. Massimiliano Pontil and Prof. Nadia Berthouze. He has published in top-tier machine learning conferences such as NIPS, ICML, and ICCV. His research focuses on multitask and transfer learning methods applied to Computer Vision tasks.

Måns Larsson is a PhD student from Chalmers University of Technology under the supervision of Professor Fredrik Kahl. His research interests include deep learning, probabilistic graphical models and semantic segmentation tasks in both medical image analysis and computer vision.

Alexander Kirillov is a PhD student at the Universität Heidelberg jointly supervised by Prof. Carsten Rother and Prof. Dmitry Vetrov. He has published in top Computer Vision and Machine Learning venues such as CVPR, ICCV and NIPS. His main research interests are deep learning models for structured output and energy minimization problems for computer vision applications.

Bogdan Savchynskyy is a senior researcher at the Universität Heidelberg, Germany. He received PhD from the Department of Image Recognition of IRTC ITS, Kiev, Ukraine. He was award the CVPR' 14, President of Ukraine Scholarship for Young Scientists' 03. He received funding on the project Exact Relaxation-Based Inference in Graphical Models from DFG. His research interests include Applied Mathematics and Computer Vision.

Carsten Rother is a full Professor at the Universität Heidelberg, Germany. His research interests are the field of Computer Vision and Machine Learning, such as segmentation, matching, object recognition, and pose estimation, with applications to Natural Science, in particular Biology and Medicine. He won best paper (honorable) mention awards at CVPR, ACCV, and CHI, and was awarded the DAGM Olympus prize in 2009.

Fredrik Kahl received his MSc degree in computer science and technology in 1995 and his PhD in mathematics in 2001. His thesis was awarded the Best Nordic Thesis Award in pattern recognition and image analysis 2001-2002 at the Scandinavian Conference on Image Analysis 2003. In 2005, he received the ICCV Marr Prize and in 2007, he was awarded an ERC Starting Grant from the European Research Council in 2008. He is currently a joint Professor at Chalmers University of Technology and Lund University.

Philip H. S. Torr received his PhD degree from Oxford University. After that he was at Oxford for another three years, then was a research scientist for Microsoft Research for six years, first in Redmond, then in Cambridge, founding the vision side of the Machine Learning and Perception Group. He is currently a professor at Oxford University. He has won awards from several top vision conferences, including ICCV, CVPR, ECCV, NIPS and BMVC.