InstanceCut: from Edges to Instances with MultiCut

Alexander Kirillov[1]   Evgeny Levinkov[2]   Bjoern Andres[2]   Bogdan Savchynskyy[1]   Carsten Rother[1]

[1]TU Dresden, Dresden, Germany   [2]MPI for Informatics, Saarbrücken, Germany

[1]name.surname@tu-dresden.de   [2]surname@mpi-inf.mpg.de

Abstract

This work addresses the task of instance-aware semantic segmentation. Our key motivation is to design a simple method with a new modelling-paradigm, which therefore has a different trade-off between advantages and disadvantages compared to known approaches. Our approach, we term InstanceCut, represents the problem by two output modalities: (i) an instance-agnostic semantic segmentation and (ii) all instance-boundaries. The former is computed from a standard convolutional neural network for semantic segmentation, and the latter is derived from a new instance-aware edge detection model. To reason globally about the optimal partitioning of an image into instances, we combine these two modalities into a novel MultiCut formulation. We evaluate our approach on the challenging CityScapes dataset. Despite the conceptual simplicity of our approach, we achieve the best result among all published methods, and perform particularly well for rare object classes.

1. Introduction

This work addresses the task of segmenting each individual instance of a semantic class in an image. The task is known as *instance-aware semantic segmentation*, in short *instance segmentation*, and is a more refined task than semantic segmentation, where each pixel is only labeled with its semantic class. An example of semantic segmentation and instance segmentation is shown in Fig. 1a-1b. While semantic segmentation has been a very popular problem to work on in the last half decade, the interest in instance segmentation has significantly increased recently. This is not surprising since semantic segmentation has already reached a high level of accuracy, in contrast to the harder task of instance segmentation. Also, from an application perspective there are many systems, such as autonomous driving or robotics, where a more detailed understanding of the surrounding is important for acting correctly in the world.

In recent years, Convolutional Neural Networks (CNN) have tremendously increased the performance of many computer vision tasks. This is also true for the task of in-
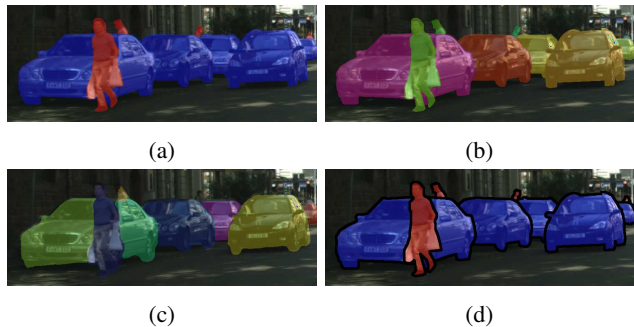

(a)　　　　　　(b)

(c)　　　　　　(d)

Figure 1: An image from the CityScapes dataset [13]: (a) Ground truth semantic segmentation, where all cars have the same label. (b) The ground truth instance segmentation, where each instance, i.e. object, is highlighted by a distinct color. In this work we use a "limiting" definition of instance segmentation, in the sense that each instance must be a connected component. Despite this limitation, we will demonstrate high-quality results. (c) Shows the result of our InstanceCut method. As can be seen, the front car is split into two instances, in contrast to (b). (d) Our connected-component instances are defined via two output modalities: (i) the semantic segmentation, (ii) all instance-boundaries (shown in bold-black).

stance segmentation, see the benchmarks [13, 37]. However, for this task it is, in our view, not clear whether the best modelling-paradigm has already been found. Hence, the motivation of this work is to explore a new, and very different, modelling-paradigm. To be more precise, we believe that the problem of instance segmentation has four core challenges, which any method has to address. Firstly, the label of an instance, e.g. "car number 5", does not have a meaning, in contrast to semantic segmentation, e.g. class "cars". Secondly, the number of instances in an image can vary greatly, e.g. between 0 and 120 for an image in the CityScapes dataset [13]. Thirdly, in contrast to object detection with bounding boxes, each instance (a bounding box) cannot simply be described by four numbers (corners of bounding box), but has to be described by a set of pixels. Finally, in contrast to semantic segmentation, a more refined labeling of the training data is needed, i.e. each instance has

to be segmented separately. Especially for rare classes, e.g. motorcycles, the amount of training data, which is available nowadays, may not be sufficient. Despite these challenges, the state of the art techniques for instance segmentation are CNN-based. As an example, [16, 53] address these challenges with a complex multi-loss cascade CNN architectures, which are, however, difficult to train. In contrast, our modelling-paradigm is very different to standard CNN-based architectures: assume that each pixel is assigned to one semantic class, and additionally we insert some edges (in-between pixels) which form loops – then we have solved the problem of instance segmentation! Each connected region, enclosed by a loop of instance-aware edges is an individual instance where the class labels of the interior pixels define its class. These are exactly the ingredients of our approach: (i) a standard CNN that outputs an instance-agnostic semantic segmentation, and (ii) a new CNN that outputs all boundaries of instances. In order to make sure that instance-boundaries encircle a connected component, and that the interior of a component has the same class label, we combine these two outputs into a novel multi-cut formulation. We call our approach *InstanceCut*.

Our InstanceCut approach has some advantages and disadvantages, which we discuss next. With respect to this, we would like to stress that these pros and cons are, however, quite different to existing approaches. This means that in the future we envision that our approach may play an important role, as a subcomponent in an "ultimate" instance segmentation system. Let us first consider the limitations, and then the advantages. The minor limitation of our approach is that, obviously, we cannot find instances that are formed by disconnected regions in the image (see Fig. 1b-1c). However, despite this limitation, we demonstrate results that exceed all published methods in terms of accuracy. In the future, we foresee various ways to overcome this limitation, e.g. by reasoning globally about shape.

We see the following major advantages of our approach. Firstly, all the four challenges for instance segmentation methods, listed above, are addressed in an elegant way: (i) the multi-cut formulation does not need a unique label for an instance; (ii) the number of instances arises naturally from the solution of the multi-cut; (iii) our formulation is on the pixel (superpixel) level; (iv) since we do not train a CNN for segmenting instances globally, our approach deals very well with instances of rare classes, as they do not need special treatment. Finally, our InstanceCut approach has another major advantage, from a practical perspective. We can employ any semantic segmentation method, as long as it provides pixel-wise log-probabilities for each class. Therefore, advances in this field may directly translate to an improvement of our method. Also, semantic segmentation, here a Fully-Convolutional-Neural-Network (FCN) [52], is part of our new edge-detection approach. Again, advances

in semantic segmentation may improve the performance of this component, as well.

Our Contributions in short form are:

• We propose a novel paradigm for instance-aware semantic segmentation, which has different pros and cons than existing approaches. In our approach, we only train classifiers for semantic segmentation and instance-edge detection, and not directly any classifier for dealing with global properties of an instance, such as shape.

• We propose a novel MultiCut formulation that reasons globally about the optimal partitioning of an image into instances.

• We propose a new FCN-based architecture for instance-aware edge detection.

• We validate experimentally that our approach achieves the best result, among all published methods, and performs particularly well for rare object classes.

2. Related Work

Proposal-based methods. This group of methods uses detection or a proposal generation mechanism as a subroutine in the instance-aware segmentation pipeline.

Several recent methods decompose the instance-aware segmentation problem into a detection stage and a foreground/background segmentation stage [16, 23]. These methods propose an end-to-end training that incorporates all parts of the model. In addition, non-maximal suppression (NMS) may be employed as a post-processing step. A very similar approach generates proposals using e.g. MCG [3] and then, in the second stage, a different network classifies these proposals [13, 22, 15, 11].

Several methods produce proposals for instance segmentations and combine them, based on learned scores [33, 41, 42] or generate parts of instances and then combine them [14, 38].

Although the proposal-based methods show state-of-the-art performance on important challenges, Pascal VOC2012 [18] and MSCOCO [37], they are limited by the quality of the used detector or proposal generator. Our method is, in turn, dependent on the quality of the used semantic segmentation. However, for the latter a considerable amount of research exists with high quality results.

Proposal-free methods. Recently, a number of alternative techniques to proposal-based approaches have been suggested in the literature. These methods explore different decompositions of instance-aware semantic segmentation followed by a post-processing step that assembles results.

In [48] the authors propose a template matching scheme for instance-aware segmentation based on three modalities: predicted semantic segmentation, depth estimation, and per-pixel direction estimation with respect to the center of the corresponding instance. The approach requires depth data for training and does not perform well on highly occluded

objects.

Another work, which focuses on instance segmentation of cars [55, 54] employs a conditional random field that reasons about instances using multiple overlapping outputs of an FCN. The latter predicts a fixed number of instances and their order within the receptive field of the FCN, i.e. for each pixel, the FCN predicts an ID of the corresponding instance or background label. However, in these methods the maximal number of instances per image must be fixed in advance. A very large number may have a negative influence on the system performances. Therefore, this method may not be well-suited for the CityScapes dataset, where the number of instances varies considerably among images.

In [50] the authors predict the bounding box of an instance for each pixel, based on instance-agnostic semantic segmentation. A post-processing step filters out the resulting instances.

Recurrent approaches produce instances one-by-one. In [43] an attention-based recurrent neural network is presented. In [44] an LSTM-based [24] approach is proposed.

The work [34] presents a proposal-free network that produces an instance-agnostic semantic segmentation, number of instances for the image, and a per-pixel bounding box of the corresponding instance. The resulting instance segmentation is obtained by clustering. The method is highly sensitive to the right prediction of the number of instances.

We also present a proposal-free method. However, ours is very different in paradigm. To infer instances, it combines semantic segmentation and object boundary detection via global reasoning.

3. InstanceCut

3.1. Overview of the proposed framework

We begin with presenting a general pipeline of our new InstanceCut framework (see Fig. 2) and then describe each component in detail. The first two blocks of the pipeline are processed independently: *semantic segmentation* and *instance-aware edge detection* operate directly on the input image. The third, *image partitioning block*, reasons about instance segmentation on the basis of the output provided by the two blocks above.

More formally, the semantic segmentation block (Section 3.2) outputs a log-probability of a semantic class $a_{i,l}$ for each class label $l \in \mathcal{L} = \{0, 1 \ldots, L\}$ and each pixel i of the input image. We call $a_{i,l}$, *per-pixel semantic class scores*. Labels $1, \ldots, L$ correspond to different semantic classes and 0 stands for background.

Independently, the instance-aware edge detection (Section 3.3) outputs log-probabilities b_i of an object boundary for each pixel i. In other words, b_i indicates how likely it is that pixel i touches an object boundary. We term b_i as *a per-pixel instance-aware edge score*. Note that these scores are class-agnostic.
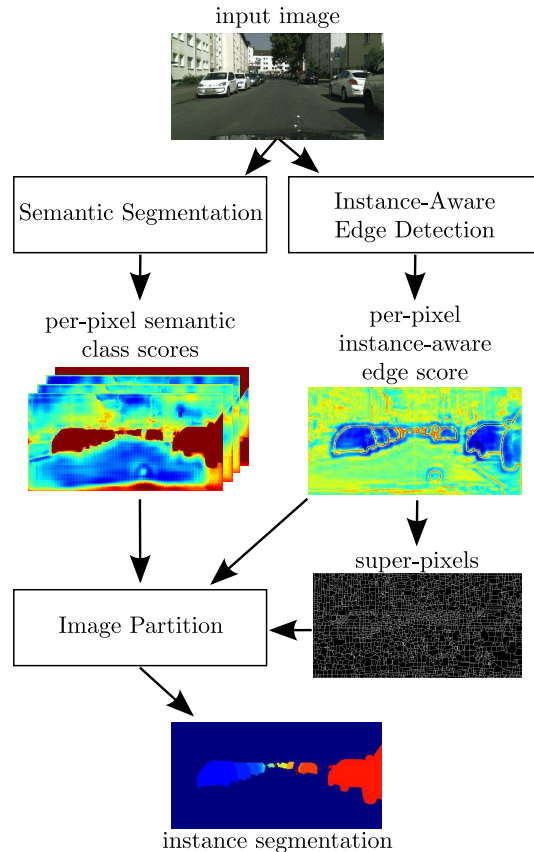

Figure 2: **Our InstanceCut pipeline - Overview.** Given an input image, two independent branches produce the per-pixel semantic class scores and per-pixel instance-aware edge scores. The edge scores are used to extract superpixels. The final image partitioning block merges the superpixels into connected components with a class label assigned to each component. The resulting components correspond to object instances and background.

Finally, the image partitioning block outputs the resulting instance segmentation, obtained using the semantic class scores and the instance-aware edge scores. We refer to Section 3.4 for a description of the corresponding optimization problem. To speed-up optimization, we reduce the problem size by resorting to a superpixel image. For the superpixel extraction we utilize the well-known watershed technique [49], which is run directly on the edge scores. This approach efficiently ensures that the extracted superpixel boundaries are aligned with boundaries of the instance-aware edge scores.

3.2. Semantic Segmentation

Recently proposed semantic segmentation frameworks are mainly based on the fully convolution network (FCN) architecture. Since the work [40], many new FCN architectures were proposed for this task [52, 20]. Some

of the methods utilize a conditional random field (CRF) model on top of an FCN [10, 36], or incorporate CRF-based mechanisms directly into a network architecture [39, 56, 45]. Current state-of-the-art methods report around 78% mean Intersection-over-Union (IoU) for the CityScapes dataset [13] and about 82% for the PASCAL VOC2012 challenge [18]. Due to the recent progress in this field, one may say that with a sufficiently large dataset, with associated dense ground truth annotation, an FCN is able to predict semantic class for each pixel with high accuracy.

In our experiments, we employ two publicly available pre-trained FCNs: Dilation10 [52] and LRR-4x [20]. These networks have been trained by the respective authors and we can also use them as provided, without any fine-tuning. Note, that we also use the CNN-CRF frameworks [56, 10] with dense CRF [31], since dense CRF's output can also be treated as the log-probability scores $a_{i,l}$.

Since our image partitioning framework works on the superpixel level we transform the pixel-wise semantic class scores $a_{i,l}$ to the superpixel-wise ones $a_{u,l}$ (here u indexes the superpixels) by averaging the corresponding pixels' scores.

3.3. Instance-Aware Edge Detection

Let us first review existing work, before we describe our approach. Edge detection (also know as *boundary detection*) is a very well studied problem in computer vision. The classical results were obtained already back in the 80's [9]. More recent methods are based on spectral clustering [47, 1, 3, 26]. These methods perform global inference on the whole image. An alternative approach suggests to treat the problem as a per-pixel classification task [35, 17]. Recent advances in deep learning have made this class of methods especially efficient, since they automatically obtain rich feature representation for classification [19, 30, 46, 6, 7, 51, 8].

The recent per-pixel classification method [8] constructs features, which are based on an FCN trained for semantic segmentation on Pascal VOC 2012 [18]. The method produces state-of-the-art edge detection performance on the BSD500 dataset [2]. The features for each pixel are designed as a concatenation of intermediate FCN features, corresponding to that particular pixel. The logistic regression trained on these features, followed by non-maximal suppression, outputs a per-pixel edge probability map. The paper suggests that the intermediate features of an FCN trained for semantic segmentation form a strong signal for solving the edge detection problem. Similarly constructed features also have been used successfully for other dense labelling problems [23].

For datasets like BSDS500 [2] most works consider general edge detection problem, where annotated edges are class- and instance-agnostic contours. In our work the instance-aware edge detection outputs a probability for each pixel, whether it touches a boundary. This problem is more challenging than canonical edge detection, since it requires to reason about contours and semantics jointly, distinguishing the true objects' boundaries and other not relevant edges, e.g. inside the object or in the background. Below (see Fig. 3), we describe a new network architecture for this task that utilizes the idea of the intermediate FCN features concatenation.

As a base for our network we use an FCN that is trained for semantic segmentation on the dataset that we want to use for object boundary prediction. In our experiments we use a pre-trained Dilation10 [52] model, however, our approach is not limited to this architecture and can utilize any other FCN-like architectures. We form a per-pixel feature representation by concatenating the intermediate feature maps of the semantic segmentation network. This is based on the following intuition: during inference, the semantic segmentation network is able to identify positions of transitions between semantic classes in the image. Therefore, its intermediate features are likely to contain a signal that helps to find the borders between classes. We believe that the same features can be useful to determine boundaries between objects.

Commonly used approaches [8, 23] suggest upscaling feature maps that have a size which is smaller than the original image to get per-pixel representation. However, in our experiments such an approach produces thick and oversmooth edge scores. This behavior can be explained by the fact that the most informative feature maps have an 8 times smaller scale than the original image. Hence, instead of upscaling, we downscale all feature maps to the size of the smallest map. Since the network was trained with rectified linear unit (ReLU) activations, the active neurons tends to output large values, therefore, we use max-pooling with a proper stride for the downscaling, see Fig. 3.

The procedure outputs the downscaled feature maps (of a *semantic segmentation FCN*, see Fig. 3) that are *concatenated* to get the downscaled per-pixel *feature map*. We utilize a *2-layer perceptron* that takes this feature map as input and outputs log-probabilities for edges (*smooth instance-aware edge map*, see Fig. 3). The perceptron method is the same for all spatial positions, therefore, it can be represented as two layers of 1×1 convolutions with the ReLU activation in between.

In our experiments we have noticed that the FCN gives smooth edge scores. Therefore, we apply *a context network* [52] that refines the scores making them sharper. The new architecture is an FCN, i.e. it can be applied to images of arbitrary size, it is differentiable and has a single loss at the end. Hence, straightforward end-to-end training can be applied for the new architecture. We upscale the resulting output map to match an input image size.

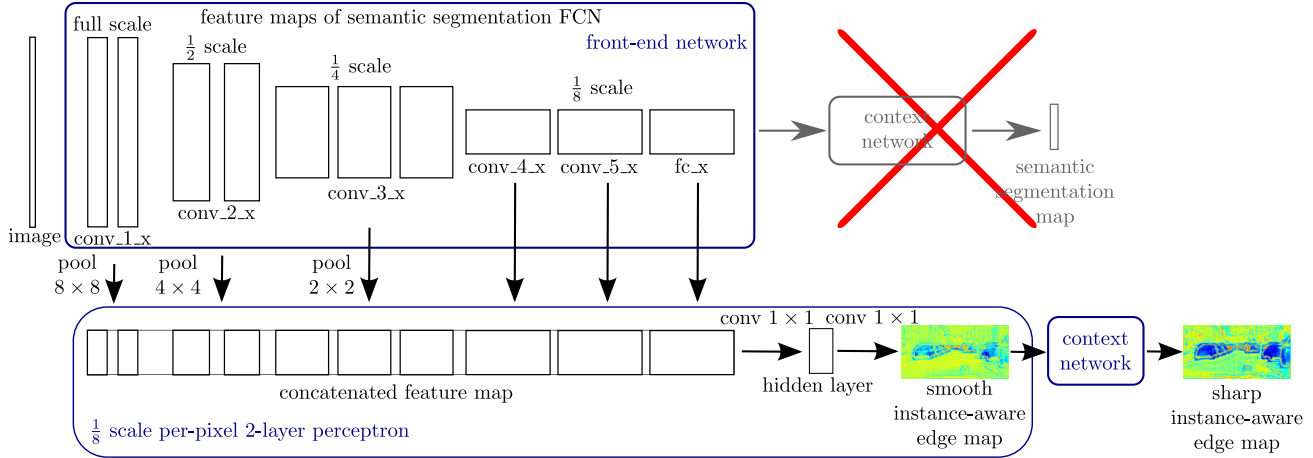Since the image partition framework, that comes next,

Figure 3: **Instance-aware edge detection block.** The semantic segmentation FCN is the front-end part of the network [52] trained for semantic segmentation on the same dataset. Its intermediate feature maps are downsampled, according to the size of the smallest feature map, by a max-pooling operation with an appropriate stride. The concatenation of the downsampled maps is used as a feature representation for a per-pixel 2-layer perceptron. The output of the perceptron is refined by a context network of Dilation10 [52] architecture.

operates on super-pixels, we need to transform the per-pixel edge scores b_i to edge scores $b_{u,v}$ for each pair $\{u, v\}$ of neighboring superpixels. We do this by averaging all scores of of those pixels that touch the border between u and v.

In the following, we describe an efficient implementation of the 2-layer perceptron and also discuss our training data for the boundary detection problem.

Efficient implementation. In our experiments, the input for the 2-layer perceptron contains about 13k features per pixel. Therefore, the first layer of the perceptron consumes a lot of memory. It is, however, possible to avoid this by using a more efficient implementation. Indeed, the first layer of the perceptron is equivalent to the summation of outputs of multiple 1×1 convolutions, which are applied to each feature map independently. For example, `conv_1` is applied to the feature maps from the `conv_1_x` intermediate layer, `conv_2` is applied to the feature maps from `conv_2_x` and its output is summed up with the output of `conv_1`, etc. This approach allows reducing the memory consumption, since the convolutions can be applied during evaluation of the front-end network.

Training data. Although it is common for ground truth data that object boundaries lie in-between pixels, we will use in the following the notion that a boundary lies on a pixel. Namely, we will assume that a pixel i is labeled as a boundary if there is a neighboring pixel j, which is assigned to a different object (or background). Given the size of modern images, this boundary extrapolation does not affect performance. As a ground truth for boundary detection we use the boundaries of object instances presented in CityScapes [13].

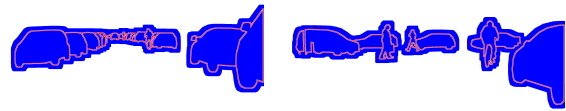As mentioned in several previous works [51, 7], highly

Figure 4: Ground truth examples for our instance-aware edge detector. Red indicates pixels that are labeled as edges, blue indicates background, i.e. no edge and white pixels are ignore.

unbalanced ground truth (GT) data heavily harms the learning progress. For example, in BSDS500 [2] less than 10% of pixels on average are labeled as edges. Our ground truth data is even more unbalanced: since we consider the object boundaries only, less than 1% of pixels are labeled as being an edge. We employ two techniques to overcome this problem of training with unbalanced data: *a balanced loss function* [51, 25] and *pruning of the ground truth data*.

The balanced loss function [51, 25] adds a coefficient to the standard log-likelihood loss that decreases the influence of errors with respect to classes that have a lot of training data. That is, for each pixel i the balanced loss is defined as

$$loss(p_{edge}, y^{GT}) = [\![y^{GT} = 1]\!] \log(p_{edge}) \qquad (1)$$
$$+ \alpha [\![y^{GT} = 0]\!] \log(1 - p_{edge}),$$

where $p_{edge} = 1/(1 - e^{-b_i})$ is the probability of the pixel i to be labeled as an edge, y^{GT} is the ground truth label for i (the label 1 corresponds to an edge), and $\alpha = N_1/N_0$ is the balancing coefficient. Here, N_1 and N_0 are numbers of pixels labeled, respectively, as 1 and 0 in the ground truth.

Another way to decrease the effect of unbalanced GT data is to subsample the GT pixels, see e.g. [8]. Since we are interested in instance-aware edge detection and combine its

output with our semantic segmentation framework, a wrong edge detection, which is far from the target objects (for example, in the sky) does not harm the overall performance of the InstanceCut framework. Hence, we consider a pixel to be labeled as background for the instance-aware edge detection if and only if it lies inside the target objects, or in an area close to it, see Fig. 4 for a few examples of the ground truth data for the CityScapes dataset [13]. In our experiments, only 6.8% of the pixels are labeled as object boundaries in the pruned ground truth data.

3.4. Image Partition

Let V be the set of superpixels extracted from the output of the instance-aware edge detection block and $E \subseteq \binom{V}{2}$ be the set of neighboring superpixels, i.e., those having a common border.

With the methods described in Sections 3.2 and 3.3 we obtain:

• Log-probabilities $\alpha_{u,l}$ of all semantic labels $l \in \mathcal{L}$ (including background) for each superpixel $u \in V$.

• Log-probabilities $b_{u,v}$ for all pairs of neighbouring superpixels $\{u, v\} \in E$, for having a cutting edge.

• Prior log-probabilities of having a boundary between any two (also equal) semantic classes $\beta_{l,l'}$, for any two labels $l, l' \in \mathcal{L}$. In particular, the weight $\beta_{l,l}$ defines, how probable it is that two neighboring super-pixel have the same label l and belong to different instances. We set $\beta_{0,0}$ to $-\infty$, assuming there are no boundaries between superpixels labeled both as background.

We want to assign a single label to each superpixel and have close-contour boundaries, such that if two neighboring superpixels belong to different classes, there is always a boundary between them.

Our problem formulation consists of two components: (i) a conditional random field model [27] and (ii) a graph partition problem, known as MultiCut [12] or correlation clustering [4]. In a certain sense, these two problems are coupled together in our formulation. Therefore, we first briefly describe each of them separately and afterwards consider their joint formulation.

Conditional Random Field (CRF). Let us, for now, assume that all $\beta_{l,l} = -\infty$, $l \in \mathcal{L}$, i.e., there can be no boundary between superpixels assigned the same label. In this case our problem is reduced to the following famous format: Let $G = (V, E)$ be an undirected graph. A finite set of labels $\mathcal{L}$ is associated with each node. With each label l in each node v a vector $\alpha_{v,l}$ is associated, which denotes *the score* of the label assigned to the node. Each pair of labels l, l' in neighbouring nodes $\{u, v\}$ is assigned a score

$$c_{u,v,l,l'} := \begin{cases} b_{u,v} + \beta_{l,l'}, & l \neq l' \\ 0, & l = l' \end{cases}.$$

The vector $\mathbf{l} \in \mathcal{L}^{|V|}$ with coordinates l_u, $u \in V$ being labels assigned to each node is called *a labeling*. *The maximum a posteriori inference* problem for the CRF is defined above reads

$$\max_{\mathbf{l} \in L^{|V|}} \sum_{u \in V} \alpha_{u,l_u} + \sum_{uv \in E} c_{u,v,l_u,l_v}. \qquad (2)$$

A solution to this problem is a usual (non-instance-aware) semantic segmentation, if we associate the graph nodes with superpixels and the graph edges will define their neighborhood.

For the MultiCut formulation below, we will require a different representation of the problem (2), in a form of *an integer quadratic problem*. Consider binary variables $x_{u,l} \in \{0, 1\}$ for each node $u \in V$ and label $l \in \mathcal{L}$. The equality $x_{u,l} = 1$ means that label l is assigned to the node u. The problem (2) now can be rewritten as follows:

$$\max_{\mathbf{x}} \sum_{u \in V} \sum_{l \in \mathcal{L}} \alpha_{u,l} x_{u,l} + \sum_{uv \in E} \sum_{l \in \mathcal{L}} \sum_{l' \in \mathcal{L}} c_{u,v,l,l'} x_{u,l} x_{v,l'}$$

$$\text{s.t.} \begin{cases} x_{u,l} \in \{0, 1\}, & u \in V, l \in \mathcal{L} \\ \sum_{l \in L} x_{u,l} = 1, & u \in V. \end{cases} \qquad (3)$$

The last constraint in (3) is added to guarantee that each node is assigned exactly one label. Although the problem (3) is NP-hard in general, it can be efficiently (and often exactly) solved for many practical instances appearing in computer vision, see [27] for an overview.

MultiCut Problem. Let us now assume a different situation, where all nodes have already got an assigned semantic label and all that we want is to *partition* each connected component (labeled with a single class) into connected regions corresponding to instances. Let us assume, for instance, that all superpixels of the component have a label l. This task has an elegant formulation as *a MultiCut* problem [12]:

Let $G = (V, E)$ be an undirected graph, with the scores $\theta_{u,v} := b_{u,v} + \beta_{l,l}$ assigned to the graph edges. Let also $\dot{\cup}$ stand for a disjoint union of sets. *The MultiCut* problem (also known as correlation clustering) is to find a partitioning $(\Pi_1, \ldots, \Pi_k)$, $\Pi_i \subseteq V$, $V = \dot{\cup}_{i=1}^{k} \Pi_i$ of the graph vertices, such that the total score of edges connecting different components is maximized. The number k of components is not fixed but is determined by the algorithm itself. Although the problem is NP-hard in general, there are efficient approximate solvers for it, see e.g. [5, 28, 29].

In the following, we will require a different representation of the MultiCut problem, in form of *an integer linear problem*. To this end, we introduce a binary variable $y_e = y_{u,v} \in \{0, 1\}$ for each edge $e = \{u, v\} \in E$. This variable takes the value 1, if u and v belong to different components, i.e. $u \in \Pi_i$, $v \in \Pi_j$ for some $i \neq j$. Edges $\{u, v\}$ with $y_{u,v} = 1$ are called *cut* edges. The vector $\mathbf{y} \in \{0, 1\}^{|E|}$ with coordinates y_e, $e \in E$ is called *a MultiCut*. Let C be the set of all cycles of the graph G. It is a known result from combinatorial optimization [12] that the MultiCut problem can be written in the following form:

$$\max_{\mathbf{y}\in\{0,1\}^{|E|}} \sum_{\{u,v\}\in E} \theta_{u,v}y_{u,v}, \ \text{ s.t. } \forall C \ \forall e' \in C: \sum_{e\in C\setminus\{e'\}} y_e \geq y_{e'}.$$
(4)

Here, the objective directly maximizes the total score of the edges and the inequality constraints basically force each cycle to have none or at least two cut edges. These *cycle constraints* ensure that the set of cut edges actually defines a partitioning. Obviously, the cut edges correspond to boundaries in our application.

Our InstanceCut Problem. Let us combine both subproblems: We want to *jointly* infer both the semantic labels *and* the partitioning of each semantic segment, with each partition component defining an object instance. To this end, consider our InstanceCut problem (5)-(8) below:

$$\max_{\substack{\mathbf{x}\in\{0,1\}^{|V||\mathcal{L}|} \\ \mathbf{y}\in\{0,1\}^{|E|}}} \sum_{u\in V}\sum_{l\in\mathcal{L}} \alpha_{u,l}x_{u,l}$$
(5)

$$+ w \sum_{uv\in E}\sum_{l\in\mathcal{L}}\sum_{l'\in\mathcal{L}} (b_{u,v}+\beta_{l,l'})x_{u,l}x_{v,l'}y_{u,v}$$

$$\sum_{l\in\mathcal{L}} x_{u,l} = 1, \ u \in V$$
(6)

$$\forall e' \in C: \sum_{e\in C\setminus\{e'\}} y_e \geq y_{e'}$$
(7)

$$\left.\begin{array}{l} x_{u,l} - x_{v,l} \leq y_{uv} \\ x_{v,l} - x_{u,l} \leq y_{uv} \end{array}\right\}, \ \{u,v\}\in E, \ l\in\mathcal{L}.$$
(8)

Objective (5) and inequalities (6)-(7) are obtained directly from merging problems (3) and (4). We also introduced the parameter w that balances the modalities. Additional constraints (8) are required to guarantee that as soon as two neighboring nodes u and v are assigned different labels, the corresponding edge $y_{u,v}$ is cut and defines a part of an instance boundary. Two nodes u and u are assigned different labels if at most one of the variables $x_{u,l}$, $x_{v,l}$ takes value 1. In this case, the largest left-hand side of one of the inequalities (8) is equal to 1 and therefore $y_{u,v}$ must be cut. The problem related to (5)-(8) was considered in [21] for foreground/background segmentation.

Although the problem (5)-(8) is NP-hard and it contains a lot of hard constraints, there exists an efficient approximate solver for it [32], which we used in our experiments. For solving the problem over 3000 nodes (superpixels) and 9 labels (segment classes) it required less than a second on average.

4. Experiments

Dataset. There are three main datasets with full annotation for the instance-aware semantic segmentation problem: PASCAL VOC2012 [18], MS COCO [37] and CityScapes [13]. We select the last one for our experimental evaluation for several reasons: (i) CityScapes has very fine annotation with precise boundaries for the annotated objects, whereas MS COCO has only coarse annotations,
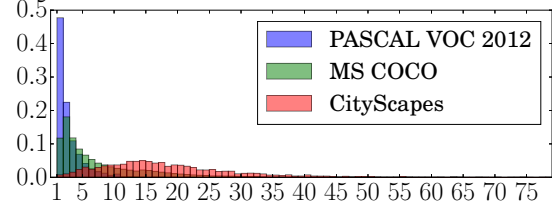

Figure 5: The histograms shows distribution of number of instances per image for different datasets. For illustrative reasons we cut long tails of CityScapes and MS COCO. We use CityScapes dataset since it contains significantly more instances per image.

for some objects, that do not coincide with the true boundaries. Since our method uses an edge detector, it is important to to have precise object boundaries for training. (ii) The median number of instances per image in CityScapes is 16, whereas PASCAL VOC has 2 and MS COCO has 4. For this work a larger number is more interesting. The distribution of the number of instances per image for different datasets is shown in Fig. 5. (iii) Unlike other datasets, CityScapes' annotation is dense, i.e. all foreground objects are labeled.

The CityScape dataset has 5000 street-scene images recorded by car-mounted cameras: 2975 images for training, 500 for validation and 1525 for testing. There are 8 classes of objects that have an instance-level annotation in the dataset: person, rider, car, truck, bus, train, motorcycle, bicycle. All images have the size of 1024×2048 pixels.

Training details. For the semantic segmentation block in our framework we test two different networks, which have publicly available trained models for CityScapes: Dilation10 [52] and LRR-4x [20]. The latter is trained using the additional coarsely annotated data, available in CityScapes. Importantly, CityScapes has 19 different semantic segmentation classes (and only 8 out of them are considered for instance segmentation) and both networks were trained to segment all these classes. We do not retrain the networks and directly use the log-probabilities for the 8 semantic classes, which we require. For the background label we take the maximum over the log-probabilities of the remaining semantic classes.

As an initial semantic segmentation network for the instance-aware edge detection block we use Dilation10 [52] pre-trained on the CityScapes. We exactly follow the training procedure described in the original paper [52]. That is, we pre-train first the front-end module with the 2-layer perceptron on top. Then we pre-train the context module of the network separately and, finally, train the whole system end-to end. All the stages are trained with the same parameters as in [52]. In our experiments the 2-layer perceptron has 16 hidden neurons. On the validation set the trained detector achieves 97.2% AUC ROC.

Parameters w (see (5)) and $\beta_{l,l'}$, for all $l,l'\in\mathcal{L}$, in our

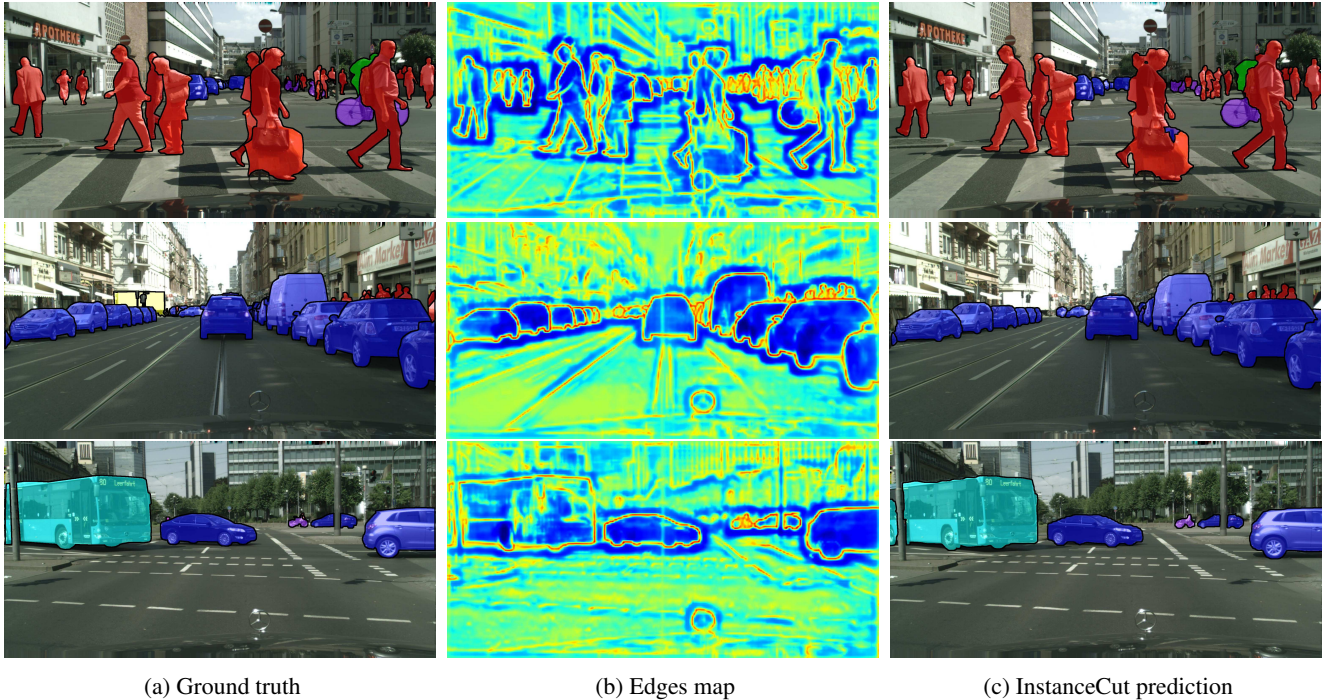(a) Ground truth   (b) Edges map   (c) InstanceCut prediction

Figure 6: Qualitative results of InstanceCut framework. Left column contains input images with the highlighted ground truth instances. Middle column depicts per-pixel instance-aware edge log-probabilities and the last column shows the results of our approach. Note that in the last example the bus and a car in the middle are separated by a lamp-post, therefore, our method returns two instances for the objects.

	AP	AP50%	AP100m	AP50m
MCG+R-CNN [13]	4.6	12.9	7.7	10.3
Uhrig et al. [48]	8.9	21.1	15.3	16.7
InstanceCut	**13.0**	**27.9**	**22.1**	**26.1**

Table 1: CityScapes results for instance-aware semantic segmentation on the test set.

InstanceCut formulation (5) are selected via 2-fold cross-validation. Instead of considering different $\beta_{l,l'}$ for all pairs of labels, we group them into two classes: 'big' and 'small'. All $\beta_{l,l'}$, where either l or l' corresponds to a (physically) big object, i.e., train, bus, or truck, are set to β_{big}. All other $\beta_{l,l'}$ are set to β_{small}. Therefore, our parameter space is only 3 dimensional and is determined by the parameters w, β_{small} and β_{big}.

Instance-level results - quantitative and qualitative. We evaluated our method using 4 metrics that are suggested by the CityScapes benchmark: AP, AP50%, AP100m and AP50m. We refer to the webpage of the benchmark for a detailed description.

The InstanceCut framework with Dilation10 [52] as the semantic segmentation block gives AP $= 14.8$ and AP50% $= 30.7$ on the validation part of the dataset. When we replace Dilation10 by LRR-4x [20] for this block the performance improves to AP $= 15.8$ and AP50% $= 32.4$, on the validation set.

Quantitative results for the test set are provided in Table 1. We compare our approach to all published methods that have results for this dataset. Among them our method shows the best performance, despite its simplicity.

5. Conclusion

We have proposed an alternative paradigm for instance-aware semantic segmentation. The paradigm represents the instance segmentation problem by a combination of two modalities: instance-agnostic semantic segmentation and instance-aware boundaries. We have presented a new framework that utilize this paradigm. The modalities are produced by FCN networks. The standard FCN model is used for semantic segmentation, whereas a new architecture is proposed for object boundaries. The modalities are combined are combined by a novel MultiCut framework, which reasons globally about instances. The proposed framework achieves the best results amongst all published methods for the challenging CityScapes dataset[13].

References

[1] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 33(5):898–916, 2011. 4

[2] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5):898–916, May 2011. 4, 5

[3] P. Arbeláez, J. Pont-Tuset, J. T. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 328–335, 2014. 2, 4

[4] N. Bansal, A. Blum, and S. Chawla. Correlation clustering. *Machine Learning*, 56(1):89–113, 2004. 6

[5] T. Beier, T. Kroeger, J. H. Kappes, U. Köthe, and F. A. Hamprecht. Cut, glue, & cut: A fast, approximate solver for multicut partitioning. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 73–80. IEEE, 2014. 6

[6] G. Bertasius, J. Shi, and L. Torresani. Deepedge: A multiscale bifurcated deep network for top-down contour detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4380–4389, 2015. 4

[7] G. Bertasius, J. Shi, and L. Torresani. High-for-low and low-for-high: Efficient boundary detection from deep object features and its applications to high-level vision. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 504–512, 2015. 4, 5

[8] G. Bertasius, J. Shi, and L. Torresani. Semantic segmentation with boundary neural fields. *arXiv preprint arXiv:1511.02674*, 2015. 4, 5

[9] J. Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698, 1986. 4

[10] 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. *arXiv preprint arXiv:1606.00915*, 2016. 4

[11] Y.-T. Chen, X. Liu, and M.-H. Yang. Multi-instance object segmentation with occlusion handling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3470–3478, 2015. 2

[12] S. Chopra and M. R. Rao. The partition problem. *Mathematical Programming*, 59(1):87–115, 1993. 6

[13] 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 *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1, 2, 4, 5, 6, 7, 8

[14] J. Dai, K. He, Y. Li, S. Ren, and J. Sun. Instance-sensitive fully convolutional networks. *arXiv preprint arXiv:1603.08678*, 2016. 2

[15] J. Dai, K. He, and J. Sun. Convolutional feature masking for joint object and stuff segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3992–4000, 2015. 2

[16] J. Dai, K. He, and J. Sun. Instance-aware semantic segmentation via multi-task network cascades. *arXiv preprint arXiv:1512.04412*, 2015. 2

[17] P. Dollár and C. L. Zitnick. Fast edge detection using structured forests. *IEEE transactions on pattern analysis and machine intelligence*, 37(8):1558–1570, 2015. 4

[18] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. 2, 4, 7

[19] Y. Ganin and V. Lempitsky. Nˆ 4-fields: Neural network nearest neighbor fields for image transforms. In *Asian Conference on Computer Vision*, pages 536–551. Springer, 2014. 4

[20] G. Ghiasi and C. C. Fowlkes. Laplacian pyramid reconstruction and refinement for semantic segmentation. In *European Conference on Computer Vision*, pages 519–534. Springer, 2016. 3, 4, 7, 8

[21] F. A. Hamprecht. Asymmetric cuts: Joint image labeling and partitioning. In *Pattern Recognition: 36th German Conference, GCPR 2014, Münster, Germany, September 2-5, 2014, Proceedings*, volume 8753, page 199. Springer, 2014. 7

[22] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Simultaneous detection and segmentation. In *European Conference on Computer Vision*, pages 297–312. Springer, 2014. 2

[23] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Hypercolumns for object segmentation and fine-grained localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 447–456, 2015. 2, 4

[24] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 3

[25] J.-J. Hwang and T.-L. Liu. Pixel-wise deep learning for contour detection. *arXiv preprint arXiv:1504.01989*, 2015. 5

[26] P. Isola, D. Zoran, D. Krishnan, and E. H. Adelson. Crisp boundary detection using pointwise mutual information. In *European Conference on Computer Vision*, pages 799–814. Springer, 2014. 4

[27] 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. *International Journal of Computer Vision*, pages 1–30, 2015. 6

[28] B. W. Kernighan and S. Lin. An efficient heuristic procedure for partitioning graphs. *Bell system technical journal*, 49(2):291–307, 1970. 6

[29] M. Keuper, E. Levinkov, N. Bonneel, G. Lavou, T. Brox, and B. Andres. Efficient decomposition of image and mesh graphs by lifted multicuts. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1751–1759. IEEE, 2015. 6

[30] J. J. Kivinen, C. K. Williams, N. Heess, and D. Technologies. Visual boundary prediction: A deep neural prediction network and quality dissection. In *AISTATS*, volume 1, page 9, 2014. 4

[31] V. Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. *Adv. Neural Inf. Process. Syst*, 2011. 4

[32] E. Levinkov, J. Uhrig, S. Tang, M. Omran, E. Insafutdinov, A. Kirillov, C. Rother, T. Brox, B. Schiele, and B. Andres. Joint graph decomposition & node labeling: Problem, algorithms, applications. *arXiv preprint arXiv:1611.04399*, 2016. 7

[33] X. Liang, Y. Wei, X. Shen, Z. Jie, J. Feng, L. Lin, and S. Yan. Reversible recursive instance-level object segmentation. *arXiv preprint arXiv:1511.04517*, 2015. 2

[34] X. Liang, Y. Wei, X. Shen, J. Yang, L. Lin, and S. Yan. Proposal-free network for instance-level object segmentation. *arXiv preprint arXiv:1509.02636*, 2015. 3

[35] J. J. Lim, C. L. Zitnick, and P. Dollár. Sketch tokens: A learned mid-level representation for contour and object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3158–3165, 2013. 4

[36] G. Lin, C. Shen, I. Reid, et al. Efficient piecewise training of deep structured models for semantic segmentation. *arXiv preprint arXiv:1504.01013*, 2015. 4

[37] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer, 2014. 1, 2, 7

[38] S. Liu, X. Qi, J. Shi, H. Zhang, and J. Jia. Multi-scale patch aggregation (mpa) for simultaneous detection and segmentation. 2

[39] Z. Liu, X. Li, P. Luo, C.-C. Loy, and X. Tang. Semantic image segmentation via deep parsing network. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1377–1385, 2015. 4

[40] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015. 3

[41] P. O. Pinheiro, R. Collobert, and P. Dollr. Learning to segment object candidates. In *NIPS*, 2015. 2

[42] P. O. Pinheiro, T.-Y. Lin, R. Collobert, and P. Dollr. Learning to refine object segments. In *ECCV*, 2016. 2

[43] M. Ren and R. S. Zemel. End-to-end instance segmentation and counting with recurrent attention. *arXiv preprint arXiv:1605.09410*, 2016. 3

[44] B. Romera-Paredes and P. H. Torr. Recurrent instance segmentation. *arXiv preprint arXiv:1511.08250*, 2015. 3

[45] A. G. Schwing and R. Urtasun. Fully connected deep structured networks. *arXiv preprint arXiv:1503.02351*, 2015. 4

[46] W. Shen, X. Wang, Y. Wang, X. Bai, and Z. Zhang. Deepcontour: A deep convolutional feature learned by positive-sharing loss for contour detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3982–3991, 2015. 4

[47] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000. 4

[48] J. Uhrig, M. Cordts, U. Franke, and T. Brox. Pixel-level encoding and depth layering for instance-level semantic labeling. *arXiv preprint arXiv:1604.05096*, 2016. 2, 8

[49] L. Vincent and P. Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE transactions on pattern analysis and machine intelligence*, 13(6):583–598, 1991. 3

[50] Z. Wu, C. Shen, and A. v. d. Hengel. Bridging category-level and instance-level semantic image segmentation. *arXiv preprint arXiv:1605.06885*, 2016. 3

[51] S. Xie and Z. Tu. Holistically-nested edge detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1395–1403, 2015. 4, 5

[52] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015. 2, 3, 4, 5, 7, 8

[53] S. Zagoruyko, A. Lerer, T.-Y. Lin, P. O. Pinheiro, S. Gross, S. Chintala, and P. Dollár. A multipath network for object detection. *arXiv preprint arXiv:1604.02135*, 2016. 2

[54] Z. Zhang, S. Fidler, and R. Urtasun. Instance-level segmentation with deep densely connected mrfs. *arXiv preprint arXiv:1512.06735*, 2015. 3

[55] Z. Zhang, A. G. Schwing, S. Fidler, and R. Urtasun. Monocular object instance segmentation and depth ordering with cnns. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2614–2622, 2015. 3

[56] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr. Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1529–1537, 2015. 4