Semi-convolutional Operators for Instance Segmentation

David Novotny*[1,2], Samuel Albanie*[1], Diane Larlus[2], and Andrea Vedaldi[1]

[1] Visual Geometry Group, Department of Engineering Science, University of Oxford
{david,albanie,vedaldi}@robots.ox.ac.uk

[2] Computer Vision Group, NAVER LABS Europe
diane.larlus@naverlabs.com

Abstract. Object detection and instance segmentation are dominated by region-based methods such as Mask RCNN. However, there is a growing interest in reducing these problems to pixel labeling tasks, as the latter could be more efficient, could be integrated seamlessly in image-to-image network architectures as used in many other tasks, and could be more accurate for objects that are not well approximated by bounding boxes. In this paper we show theoretically and empirically that constructing dense pixel embeddings that can separate object instances cannot be easily achieved using convolutional operators. At the same time, we show that simple modifications, which we call semi-convolutional, have a much better chance of succeeding at this task. We use the latter to show a connection to Hough voting as well as to a variant of the bilateral kernel that is spatially steered by a convolutional network. We demonstrate that these operators can also be used to improve approaches such as Mask RCNN, demonstrating better segmentation of complex biological shapes and PASCAL VOC categories than achievable by Mask RCNN alone.

Keywords: Instance embedding, object detection, instance segmentation, coloring, semi-convolutional

1 Introduction

State-of-the-art methods for detecting objects in images, such as R-CNN [19,18,47], YOLO [45], and SSD [39], can be seen as variants of the same paradigm: a certain number of candidate image regions are proposed, either dynamically or from a fixed pool, and then a convolutional neural network (CNN) is used to decide which of these regions tightly enclose an instance of the object of interest. An important advantage of this strategy, which we call *propose & verify* (P&V), is that it works particularly well with standard CNNs. However, P&V also has several significant shortcomings, starting from the fact that rectangular proposals can only approximate the actual shape of objects; segmenting objects, in particular, requires a two-step approach where, as in Mask R-CNN [23], one first

* Equal contribution

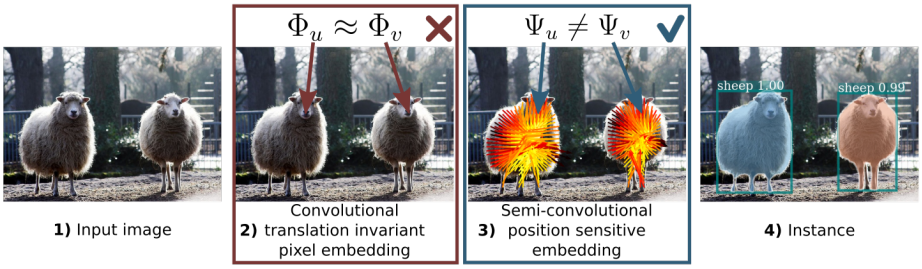


Fig. 1. Approaches for instance segmentation based on dense coloring via convolutional pixel embeddings cannot easily distinguishing identical copies of an object. In this paper, we propose a novel semi-convolutional embedding that is better suited for instance segmentation.

detects object instances using simple shapes such as rectangles, and only then refines the detections to pixel-accurate segmentations.

An alternative to P&V that can overcome such limitations is to label directly individual pixels with an identifier of the corresponding object occurrence. This approach, which we call *instance coloring* (IC), can efficiently represent any number of objects of arbitrary shape by predicting a single label map. Thus IC is in principle much more efficient than P&V. Another appeal of IC is that it can be formulated as an image-to-image regression problem, similar to other image understanding tasks such as denoising, depth and normal estimation, and semantic segmentation. Thus this strategy may allow to more easily build *unified architectures* such as [27,25] that can solve instance segmentations together with other problems.

Despite the theoretical benefits of IC, however, P&V methods currently dominate in terms of overall accuracy. The goal of this paper is to explore some of the reasons for this gap and to suggest workarounds. Part of the problem may be in the nature of the dense labels. The most obvious way of coloring objects is to number them and "paint" them with their corresponding number. However, the latter is a global operation as it requires to be aware of all the objects in the image. CNNs, which are *local and translation invariant*, may therefore be ill-suited for direct enumeration. Several authors have thus explored alternative coloring schemes more suitable for convolutional networks. A popular approach is to assign an arbitrary color (often in the guise of a real vector) to each object occurrence, with the only requirement that different colors should be used for different objects [15,6,28]. The resulting color *affinities* can then be used to easily enumerate object a posteriori via a non-convolutional algorithm.

In this paper, we argue that even the latter technique is insufficient to make IC amenable to computation by CNNs. The reason is that, since CNNs are translation invariant, they must still assign the same color to identical copies of an object, making replicas indistinguishable by convolutional coloring. This argument, which is developed rigorously in sec. 3.6, holds in the limit since in practice the receptive field size of most CNNs is nearly as large as the whole

image; however, it suggests that the convolutional structure of the network is at least an unnatural fit for IC.

In order to overcome this issue, we suggest that an architecture used for IC should not be translation invariant; while this may appear to be a significant departure from convolutional networks, we also show that a small modification of standard CNNs can overcome the problem. We do so by defining *semi-convolutional* operators which mix information extracted from a convolutional network with information about the global location of a pixel (sec. 3.1 and fig. 1). We train the latter (sec. 3.2) so that the response of the operator is similar for all pixels that belong to the same object instance, making this embedding naturally suited for IC. We show that, if the mixing function is additive, then the resulting operator bears some resemblance to Hough voting and related detection approaches. After extending the embedding to incorporate standard convolutional responses that capture appearance cues (sec. 3.3), we use it to induce pixel affinities and show how the latter can be interpreted as a steered version of a bilateral kernel (sec. 3.4). Finally, we show how such affinities can also be integrated in methods such as Mask RCNN (sec. 3.5).

We assess our method with several experiments. We start by investigating the limit properties of our approach on simple synthetic data. Then, we show that our semi-convolutional feature extractor can be successfully combined with state-of-the-art approaches to tackle parsing of biological images containing overlapping and articulated organisms (sec. 4.2). Finally, we apply the latter to a standard instance segmentation benchmark PASCAL VOC (sec. 4.3). We show in all such cases that the use of semi-convolutional features can improve the performance of state-of-the-art instance segmentation methods such as Mask RCNN.

2 Related work

The past years have seen large improvements in object detection, thanks to powerful baselines such as Faster-RCNN [47], SSD [39] or other similar approaches [11,45,35], all from the *propose & verify* strategy.

Following the success of object detection and semantic segmentation, the challenging task of instance-level segmentation has received increasing attention. Several very different families of approaches have been proposed.

Proposal-based instance segmentation. While earlier methods relied on bottom-up segmentations [18,9], the vast majority of recent instance-level approaches combine segment proposals together with powerful object classifiers. In general, they implement a multi-stage pipeline that first generates region proposals or class agnostic boxes, and then classifies them [30,20,7,43,10,44,33]. For instance DeepMask [43] and follow-up approaches [44,8] learn to propose segment candidates that are then classified. The MNC approach [10], based on Faster-RCNN [47], repeats this process twice [10] while [33] does it multiple times. [22] extends [10] to model the shape of objects. The fully convolutional instance segmentation method of [32] also combines segmentation proposal and object detection using a position sensitive score map.

Some methods start with semantic segmentation first, and then cut the regions obtained for each category into multiple instances [26,4,38], possibly involving higher-order CRFs [3].

Among the most successful methods to date, Mask-RCNN [23] extends Faster R-CNN [47] with a small fully convolutional network branch [41] producing segmentation masks for each region of interest predicted by the detection branch. Despite its outstanding results, Mask-RCNN does not come without shortcomings: it relies on a small and predefined set of region proposals and non-maximum suppression, making it less robust to strong occlusions, crowded scenes, or objects with fundamentally non-rectangular shapes (see detailed discussion in sec. 3.6).

Instance-sensitive embeddings. Some works have explored the use of pixel-level embeddings in the context of clustering tasks, employing them as a soft, differentiable proxy for cluster assignments [54,21,15,12,42,28]. This is reminiscent of unsupervised image segmentation approaches [49,16]. It has been used for body joints [42], semantic segmentation [1,21,6] and optical flow [1], and, more relevant to our work, to instance segmentation [15,12,6,28].

The goal of this type of approaches is to bring points that belong to the same instance close to each other in an embedding space, so that the decision for two pixels to belong to the same instance can be directly measured by a simple distance function. Such an embedding requires a high degree of invariance to the interior appearance of objects.

Among the most recent methods, [15] combines the embedding with a greedy mechanism to select seed pixels, that are used as starting points to construct instance segments. [6] connects embeddings, low rank matrices and densely connected random fields. [28] embeds the pixels and then groups them into instances with a variant of mean-shift that is implemented as a recurrent neural network. All these approaches are based on convolutions, that are local and translation invariant by construction, and consequently are inherently ill-suited to distinguish several identical instances of the same object (see more details about the convolutional coloring dilemma in sec. 3.6). A recent work [25] employs position sensitive convolutional embeddings that regress the location of the centroid of each pixel's instance. We mainly differ by allowing embeddings to regress an unconstrained representative point of each instance.

Among other approaches using a clustering component, [50] leverages a coverage loss and [56,51,52] make use of depth information. In particular, [52] trains a network to predict each pixel direction towards its instance center along with monocular depth and semantic labeling. Then template matching and proposal fusion techniques are applied.

Other instance segmentation approaches. Several methods [43,44,34,24] move away from box proposals and use Faster-RCNN [47] to produce "centerness" scores on each pixel instead. They directly predict the mask of each object in a second stage. An issue with such approaches is that objects do not necessarily fit in the receptive fields.

Recurrent approaches sequentially generate a list of individual segments. For instance, [2] uses an LSTM for detection with a permutation invariant loss while

[48] uses an LSTM to produce binary segmentation masks for each instance. [46] extends [48] by refining segmentations in each window using a box network. These approaches are slow and do not scale to large and crowded images.

Some approaches use watershed algorithms. [4] predicts pixel-level energy values and then partition the image with a watershed algorithm. [26] combines a watershed algorithm with an instance aware boundary map. Such methods create disconnected regions, especially in the presence of occlusion.

3 Method

3.1 Semi-convolutional networks for instance coloring

Let $\mathbf{x} \in \mathcal{X} = \mathbb{R}^{H \times W \times 3}$ be an image and $u \in \Omega = \{1, \dots, H\} \times \{1, \dots, W\}$ a pixel. In instance segmentation, the goal is to map the image to a collection $\mathcal{S}_{\mathbf{x}} = \{S_1, \dots, S_{K_{\mathbf{x}}}\} \subset 2^{\Omega}$ of image regions, each representing an occurrence of an object of interest. The symbol $S_0 = \Omega - \cup_k S_k$ will denote the complementary region, representing background. The regions as well as their number are a function of the image and the goal is to predict both.

In this paper, we are interested in methods that reduce instance segmentation to a pixel-labeling problem. Namely, we seek to learn a function $\Phi : \mathcal{X} \to \mathcal{L}^{\Omega}$ that associates to each pixel u a certain label $\Phi_u(\mathbf{x}) \in \mathcal{L}$ so that, as a whole, labels encode the segmentation $\mathcal{S}_{\mathbf{x}}$. Intuitively, this can be done by painting different regions with different "colors" (aka pixel labels) making objects easy to recover in post-processing. We call this process *instance coloring* (IC).

A popular IC approach is to use real vectors $\mathcal{L} = \mathbb{R}^d$ as colors, and then require that the colors of different regions are sufficiently well separated. Formally, there should be a margin $M > 0$ such that:

$$\forall u, v \in \Omega : \quad \begin{cases} \|\Phi_u(\mathbf{x}) - \Phi_v(\mathbf{x})\| \leq 1 - M, & \exists k : u, v \in S_k, \\ \|\Phi_u(\mathbf{x}) - \Phi_v(\mathbf{x})\| \geq 1 + M, & \text{otherwise.} \end{cases} \tag{1}$$

If this is the case, clustering colors trivially reconstructs the regions.

Unfortunately, it is difficult for a convolutional operator Φ to satisfy constraint (1) or analogous ones. While this is demonstrated formally in sec. 3.6, for now an intuition suffices: if the image contains replicas of the same object, then a convolutional network, which is translation invariant, must assign the same color to each copy.

If convolutional operators are inappropriate, then, we must abandon them in favor of non-convolutional ones. While this sounds complex, we suggest that very simple modifications of convolutional operators, which we call *semi-convolutional*, may suffice. In particular, if $\Phi_u(\mathbf{x})$ is the output of a convolutional operator at pixel u, then we can construct a non-convolutional response by mixing it with information about the pixel location. Mathematically, we can define a semi-convolutional operator as:

$$\Psi_u(\mathbf{x}) = f(\Phi_u(\mathbf{x}), u) \tag{2}$$

where $f : \mathcal{L} \times \Omega \to \mathcal{L}'$ is a suitable mixing function. As our main example of such an operator, we consider a particularly simple type of mixing function, namely addition. With it, eq. (2) specializes to:

$$\Psi_u(\mathbf{x}) = \Phi_u(\mathbf{x}) + u, \qquad \Phi_u(\mathbf{x}) \in \mathcal{L} = \mathbb{R}^2. \tag{3}$$

While this choice is restrictive, it has the benefit of having a very simple interpretation. Suppose in fact that the resulting embedding can perfectly separate instances, in the sense that $\Psi_u(\mathbf{x}) = \Psi_v(\mathbf{x}) \Leftrightarrow \exists k : (u, v) \in S_k$. Then for all the pixels of the region S_k we can write in particular:

$$\forall u \in S_k : \quad \Phi_u(\mathbf{x}) + u = c_k \tag{4}$$

where $c_k \in \mathbb{R}^2$ is an *instance-specific point*. In other words, we see that the effect of learning this semi-convolutional embedding for instance segmentation is to predict a *displacement field* $\Phi(\mathbf{x})$ that maps all pixels of an object instance to an instance-specific centroid c_k. An illustration of the displacement field can be found fig. 2.

Relation to Hough voting and implicit shape models. Eq. (3) and (4) are reminiscent of well known detection methods in computer vision: Hough voting [13,5] and implicit shape model (ISM) [31]. Recall that both of these methods map image patches to votes for the parameters θ of possible object occurrences. In simple cases, $\theta \in \mathbb{R}^2$ can be the centroid of an object, and casting votes may have a form similar to eq. (4).

This establishes, a clear link between voting-based methods for object detection and coloring methods for instance segmentation. At the same time, there are significant differences. First, the goal here is to group pixels, not to reconstruct the parameters of an object instance (such as its centroid and scale). Eq. (3) may have this interpretation, but the more general version eq. (2) does not. Second, in methods such as Hough or ISM the centroid is defined a-priori as the actual center of the object; here the centroid c_k has no explicit meaning, but is automatically inferred as a useful but arbitrary reference point. Third, in traditional voting schemes voting integrates local information extracted from individual patches; here the receptive field size of $\Phi_u(\mathbf{x})$ may be enough to comprise the whole object, or more. The goal of eq. (2) and (3) is not to pool local information, but to solve a representational issue.

3.2 Learning additive semi-convolutional features

Learning the semi-convolutional features of eq. (2) can be formulated in many different ways. Here we adopt a simple direct formulation inspired by [12] and build a loss by considering, for each image $\mathbf{x}$ and instance $S \in \mathcal{S}$ in its segmentation, the distance between the embedding of each pixel $u \in S$ and the segment-wise mean of these embeddings:

$$\mathcal{L}(\Psi|\mathbf{x}, \mathcal{S}) = \sum_{S \in \mathcal{S}} \frac{1}{|S|} \sum_{u \in S} \left\| \Psi_u(\mathbf{x}) - \frac{1}{|S|} \sum_{u \in S} \Psi_u(\mathbf{x}) \right\|. \tag{5}$$

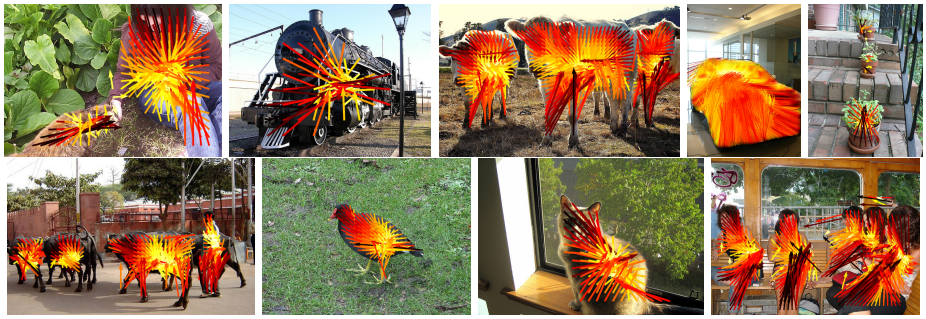


Fig. 2. Semi-convolutional embedding. The first two dimensions of the embedding $\Phi_u(\mathbf{x})$ are visualized as arrows starting from the corresponding pixel location u. Arrows from the same instance tend to point towards a *instance-specific* location c_k.

Note that while this quantity resembles the variance of the embedding values for each segment, it is not as the distance is not squared; this was found to be more robust.

Note also that this loss is simpler than the margin condition (1) and than the losses proposed in [12], which resemble (1) more closely. In particular, this loss only includes an "attractive" force which encourages embeddings for each segment to be all equal to a certain mean value, but does not explicitly encourage different segments to be assigned different embedding values. While this can be done too, empirically we found that minimizing eq. (5) is sufficient to learn good additive semi-convolutional embeddings.

3.3 Coloring instances using individuals' traits

In practice, very rarely an image contains exact replicas of a certain object. Instead, it is more typical for different occurrences to have some distinctive individual traits. For example, different people are generally dressed in different ways, including wearing different colors. In instance segmentation, one can use such cues to tell right away an instance from another. Furthermore, these cues can be extracted by conventional convolutional operators.

In order to incorporate such cues in our additive semi-convolutional formulation, we still consider the expression $\Psi_u(x) = \hat{u} + \Phi_u(\mathbf{x})$. However, we relax $\Phi_u(\mathbf{x}) \in \mathbb{R}^d$ to have more than two dimensions $d > 2$. Furthermore, we define $\hat{u}$ as the pixel coordinates of u, u_x and u_y, extended by zero padding:

$$\hat{u} = \begin{bmatrix} u_x & u_y & 0 & \dots & 0 \end{bmatrix}^\top \in \mathbb{R}^d. \tag{6}$$

In this manner, the last $d-2$ dimensions of the embedding work as conventional convolutional features and can extract instance-specific traits normally.

3.4 Steered bilateral kernels

The pixel embedding vectors $\Psi_u(\mathbf{x})$ must ultimately be decoded as a set of image regions. Again, there are several possible strategies, starting from simple K-means clustering, that can be used to do so. In this section, we consider transforming embeddings in an affinity matrix between two pixels, as the latter can be used in numerous algorithms.

In order to define the affinity between pixels $u, v \in \Omega$, consider first the Gaussian kernel

$$K(u,v) = \exp\left(-\frac{\|\Psi_u(\mathbf{x}) - \Psi_v(\mathbf{x})\|^2}{2}\right). \tag{7}$$

If the augmented embedding eq. (6) is used in the definition of $\Psi_u(\mathbf{x}) = \hat{u}+\Phi_u(\mathbf{x})$, we can split $\Phi_u(\mathbf{x})$ into a geometric part $\Phi^g_u(\mathbf{x}) \in \mathbb{R}^2$ and an appearance part $\Phi^a_u(\mathbf{x}) \in \mathbb{R}^{d-2}$ and expand this kernel as follows:

$$K(u,v) = \exp\left(-\frac{\|(u+\Phi^g_u(\mathbf{x})) - (v+\Phi^g_v(\mathbf{x}))\|^2}{2}\right) \exp\left(-\frac{\|\Phi^a_u(\mathbf{x}) - \Phi^a_v(\mathbf{x})\|^2}{2}\right). \tag{8}$$

It is interesting to compare this definition to the one of the *bilateral kernel*:[3]

$$K_{\text{bil}}(u,v) = \exp\left(-\frac{\|u-v\|^2}{2}\right) \exp\left(-\frac{\|\Phi^a_u(\mathbf{x}) - \Phi^a_v(\mathbf{x})\|^2}{2}\right). \tag{9}$$

The bilateral kernel is very popular in many applications, including image filtering and mean shift clustering. The idea of the bilateral kernel is to consider pixels to be similar if they are close in both space and appearance. Here we have shown that kernel (8) and hence kernel (7) can be interpreted as a generalization of this kernel where spatial locations are steered (distorted) by the network to move pixels that belong to the same underlying object instance closer together.

In a practical implementation of these kernels, vectors should be rescaled before being compared, for example in order to balance spatial and appearance components. In our case, since embeddings are trained end-to-end, the network can learn to perform this balancing automatically, but for the fact that (4) implicitly defines the scaling of the spatial component of the kernel. Hence, we modify eq. (7) in two ways: by introducing a learnable scalar parameter σ and by considering a Laplacian rather than a Gaussian kernel:

$$K_\sigma(u,v) = \exp\left(-\frac{\|\Psi_u(\mathbf{x}) - \Psi_v(\mathbf{x})\|}{\sigma}\right). \tag{10}$$

This kernel is more robust to outliers (as it uses the Euclidean distance rather than its square) and is still positive definite [17]. In the next section we show an example of how this kernel can be used to perform instance coloring.

[3] In the bilateral kernel, a common choice is to set $\Phi^a_u(\mathbf{x}) = \mathbf{x}_u \in \mathbb{R}^3$ as the RGB triplet for the appearance features.

3.5 Semi-convolutional Mask-RCNN

The semi-convolutional framework we proposed in sec. 3.1 is very generic and can be combined with many existing approaches. Here, we describe how it can be combined with the Mask-RCNN (MRCNN) framework [23], the current state-of-the-art in instance segmentation.

MRCNN is based on the RCNN *propose & verify* strategy and first produces a set of rectangular regions $\mathcal{R}$, where each rectangle $R \in \mathcal{R}$ tightly encloses an instance candidate. Then a fully convolutional network (FCN) produces foreground/background segmentation inside each region candidate. In practice, it labels every pixel u_i in R with a foreground score logit $s(u_i) \in \mathbb{R}$. However, this is not an optimal strategy for articulated objects or occluded scenes (as validated in sec. 4.2), as it is difficult for a standard FCN to perform individual foreground/background predictions. Hence we leverage our pixel-level translation sensitive embeddings in order to improve the quality of the predictions $s(u_i)$.

Extending MRCNN. Our approach is based on two intuitions: first, some points are easier to be recognized as foreground than others, and, second, once one such *seed point* has been determined, its affinity with other pixels can be used to cut out the foreground region.

In practice, we first identify a seed pixel u_s in each region R using the MRCNN foreground confidence score map $\mathbf{s} = [s(u_1), \ldots, s(u_{|R|})]$. We select the *most confident* seed point as $u_s = \text{argmax}_{1 \leq i \leq |R|} s(u_i)$, evaluate the steered bilateral kernel $K_\sigma(u_s, u)$ after extracting the embeddings Ψ_{u_s} for the seed and Ψ_{u_i} of each pixel u_i in the region, and then defining updated scores $\hat{s}(u_i)$ as $\hat{s}(u_i) = s(u_i) + \log K_\sigma(u_s, u_i)$. The combination of the scores and the kernel is performed in the log-space due to improved numerical stability. The final per-pixel foreground probabilities are obtained as in [23] with $\text{sigmoid}(\hat{s}(u_i))$.

The entire architecture —the region selection mechanism, the foreground prediction, and the pixel-level embedding —is trained end-to-end. For differentiability, this requires the following modifications: we replace the maximum operator with a soft maximum over the scores $\mathbf{p}_s = \text{softmax}(\mathbf{s})$ and we obtain the seed embedding Ψ_{u_s} as the expectation over the embeddings Ψ_u under the probability density $\mathbf{p}_s$. The network optimizer minimizes, together with the MRCNN losses, the image-level embedding loss $\mathcal{L}(\Psi|\mathbf{x}, \mathcal{S})$ and further attaches a secondary binary cross entropy loss that, similar to the MRCNN mask predictor, minimizes binary cross entropy between the kernel output $K_\sigma(u_s, u_i)$ and the ground truth instance masks.

The predictors of our semi-convolutional features Ψ_u were implemented as an output of a shallow subnetwork, shared between all the FPN layers. This subnet consists of a 256-channel 1×1 convolutional filter followed by ReLU and a final 3×3 convolutional filter producing $D = 8$ dimensional embedding Ψ_u. Due to an excessive sensitivity of the RPN component to perturbations of the underlying FPN representation, we downscale the gradients that are generated by the shallow subnetwork and received by the shared FPN tensors by a factor of 10.

3.6 The convolutional coloring dilemma

In this section, we prove some properties of convolutional operators in relation to solving instance segmentation problems. In order to do this, we need to start by formalizing the problem.

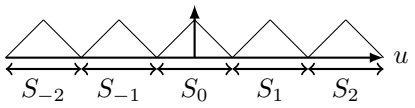
We consider signals (images) of the type $\mathbf{x} : \Omega \to \mathbb{R}$, where the domain Ω is either $\mathbb{Z}^m$ or $\mathbb{R}^m$.[4] In segmentation, we are given a family $\mathbf{x} \in \mathcal{X}$ of such signals, each of which is associated to a certain partition $\mathcal{S}_\mathbf{x} = \{S_1, \dots, S_{K_\mathbf{x}}\}$ of the domain Ω. The goal is to construct a *segmentation algorithm* $\mathcal{A} : \mathbf{x} \mapsto \mathcal{S}_\mathbf{x}$ that computes this function. We look in particular at algorithms that pre-process the signal by assigning a label $\Phi_u(\mathbf{x}) \in \mathcal{L}$ to each point $u \in \Omega$ of the domain. Furthermore, we assume that this labeling operator Φ is *local and translation invariant*[5] so as to be implementable with a convolutional neural network.

There are two families of algorithms that can be used to segment signals in this manner, discussed next.

Propose & verify. The first family of algorithms submits all possible regions $S_r \subset \Omega$, indexed for convenience by a variable r, to a labeling function $\Phi_r(\mathbf{x}) \in \{0, 1\}$ that *verifies* which ones belong to the segmentation $\mathcal{S}_\mathbf{x}$ (i.e. $\Phi_r(\mathbf{x}) = 1 \Leftrightarrow S_r \in \mathcal{S}_\mathbf{x}$). Since in practice it is not possible to test all possible subsets of Ω, such an algorithm must focus on a smaller set of proposal regions. A typical choice is to consider all translated squares (or rectangles) $S_u = [-H, H]^m + u$. Since the index variable $u \in \Omega$ is now a translation, the operator $\Phi_u(\mathbf{x})$ has the form discussed above, although it is not necessarily local or translation invariant.

Instance coloring. The second family of approaches directly colors (labels) pixels with the index of the corresponding region, i.e. $\Phi_u(\mathbf{x}) = k \Leftrightarrow u \in S_k$. Differently from P&V, this can efficiently represent arbitrary shapes. However, the map Φ needs implicitly to decide which number to assign to each region, which is a global operation. Several authors have sought to make this more amenable to convolutional networks. A popular approach [15,12] is to color pixels arbitrarily (for example using vector embeddings) so that similar colors are assigned to pixels in the same region and different colors are used between regions, as already detailed in eq. (1).

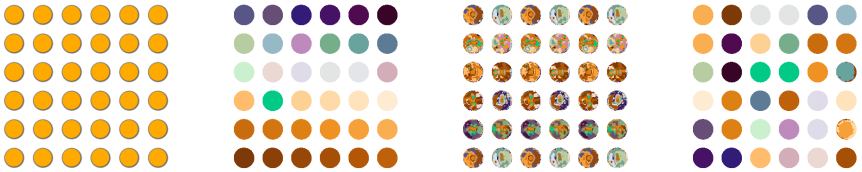
Convolutional coloring dilemma. Here we show that, even with the variants discussed above, IC cannot be approached with convolutional operators even for cases where these would work with P&V.



We do so by considering a simple 1D example. Let $\mathbf{x}$ be a signal of period 2 (i.e. $x_{u+2} = x_u$) where for $u \in [-1, 1]$ the signal is given by $x_u = \min(1-u, 1+u)$.

[4] We assume that the domain extends to infinity to avoid having to deal explicitly with boundary conditions.

[5] We say that Φ is translation invariant if $\Phi_u(\mathbf{x}(\cdot - \tau)) = \Phi_{u-\tau}(\mathbf{x})$ for all translations $\tau \in \Omega$. We say that it is also local if there exists a constant $M > 0$ such that $x_u = x_{u'}$ for all $|u - u'| < M$ implies that $\Phi_u(\mathbf{x}) = \Phi'_u(\mathbf{x})$.



(a) Training image (b) GT instance labels (c) Convolutional embedding (d) Semi-conv. embedding (ours)

Fig. 3. Experiment on synthetic data. An instance segmentation pixel embedding is trained for a synthetic training image consisting of a regular dot pattern (a). After training a model on that image, the produced embeddings are clustered using k-means, encoding the corresponding cluster assignments with consistent pixel colors. A standard convolutional embedding (c) cannot successfully embed each dot into a unique location due to its translational invariance. Our proposed semi-convolutional operator (d) naturally embeds dots with identical appearance but distinct location into distinct regions in the feature space and hence allows for successful clustering of the instances.

Suppose that the segmentation associated to $\mathbf{x}$ is $\mathcal{S} = \{[-1, 1] + 2k, k \in \mathbb{Z}\}$. If we assume that a necessary condition for a coloring-based algorithm is that at least some of the regions are assigned different colors, we see that this cannot be achieved by a convolutional operator. In fact, due to the periodicity of $\mathbf{x}$, any translation invariant function will assign exactly the same color to pixels $2k, k \in \mathbb{Z}$. Thus *all* regions have at least one point with the same color.

On the other hand, this problem can be solved by P&V using the proposal set $\{[-1, 1] + u, u \in \Omega\}$ and the local and translation invariant verification function $\Phi_u(\mathbf{x}) = [x_u = 1]$, which detects the center of each region.

The latter is an extreme example of a convolutional coloring dilemma: namely, a local and translation invariant operator will naturally assign the same color to identical copies of an object even if when they are distinct occurrences (c.f. interesting concurrent work that explores related convolutional dilemmas [37]).

Solving the dilemma. Solving the coloring dilemma can be achieved by using operators that are *not* translation invariant. In the counterexample above, this can be done by using the semi-convolutional function $\Phi_u(x) = u + (1 - x_u)\dot{x}_u$. It is easy to show that $\Phi_u(x) = 2k$ colors each pixel $u \in S_k = [-1, 1] + 2k$ with twice the index of the corresponding region by moving each point u to the center of the closest region. This works because such displacements can be computed by looking only locally, based on the shape of the signal.

4 Experiments

We first conduct experiments on synthetic data in order to clearly demonstrate inherent limitations of convolutional operators for the task of instance segmen-

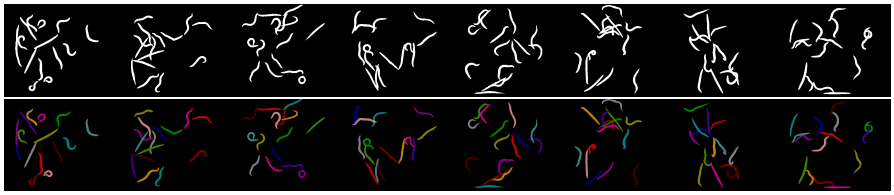


Fig. 4. Sample image crops (top) and corresponding ground-truth (bottom) from the *C. Elegans* dataset.

Table 1. Average precision (AP) for instance segmentation on *C. Elegans* reporting the standard COCO evaluation metrics [36]

AP	AP	$AP_{0.5}$	$AP_{0.75}$	AP_S	AP_M
Ours	**0.569**	**0.885**	**0.661**	**0.511**	**0.671**
MRCNN [24]	0.559	0.865	0.641	0.502	0.650

tation. In the ensuing parts we demonstrate benefits of the semi-convolutional operators on a challenging scenario with a high number of overlapping articulated instances and finally we compare to the competition on a standard instance segmentation benchmark.

4.1 Synthetic experiments

In sec. 3.1 and 3.6 we suggested that convolution operators are unsuitable for instance segmentation via coloring, but that semi-convolutional ones can do. These experiments illustrate this point by learning a deep neural network to segment a synthetic image x_S where object instances correspond to identical dots arranged in a regular grid (fig. 3 (a)).

We use a network consisting of a pretrained ResNet50 model truncated after the Res2c layer, followed by a set of 1×1 filters that, for each pixel u, produce 8-dimensional pixel embeddings $\Phi_u(x_S)$ or $\Psi_u(x_S)$. We optimize the network by minimizing the loss from *eq.* (5) with stochastic gradient descent. Then, the embeddings corresponding to the foreground regions are extracted and clustered with the k-means algorithm into K clusters, where K is the true number of dots present in the synthetic image.

Fig. 3 visualizes the results. Clustering the features consisting of the position invariant convolutional embedding $\Phi_u(x_S)$ results in nearly random clusters (fig. 3 (c)). On the contrary, the semi-convolutional embedding $\Psi_u(x_S) = \Phi_u(x_S) + u$ allows to separate the different instances almost perfectly when compared to the ground truth segmentation masks (fig. 3 (d)).

4.2 Parsing biological images

The second set of experiments considers the parsing of biological images. Organisms to be segmented present non-rigid pose variations, and frequently form

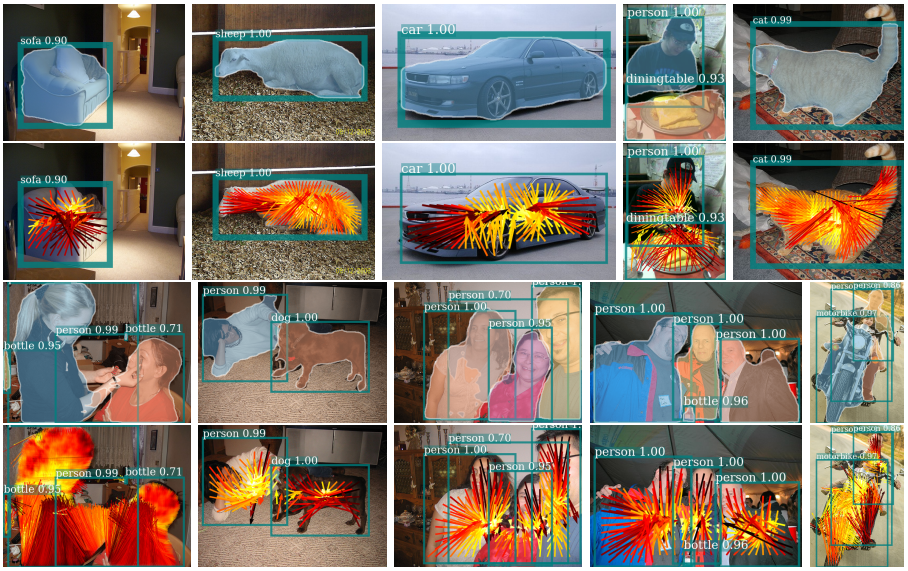


Fig. 5. Instance segmentation on Pascal VOC 2012. Each pair of rows visualizes instance segmentations produced with method, together with the corresponding semi-convolutional embeddings

clusters of overlapping instances, making the parsing of such images challenging. Yet, this scenario is of crucial importance for many biological studies.

Dataset and evaluation. We evaluate our approach on the C. Elegans dataset (illustrated fig. 4), a subset of the Broad Biomedical Benchmark collection [40]. The dataset consists of 100 bright-field microscopy images. Following standard practice [53,55], we operate on the binary segmentation of the microscopy images. However, since there is no publicly defined evaluation protocol for this dataset, a fair numerical comparison with previously published experiments is infeasible. We therefore compare our method against a very strong baseline (MRCNN) and adopt the methodology introduced by [55] in which the dataset is divided into 50 training and 50 test images. We evaluate the segmentation using average precision (AP) computed using the standard COCO evaluation criteria [36]. We compare our method against the MRCNN FPN-101 model from [23] which attains results on par with state of the art on the challenging COCO instance segmentation task.

Results. The results are given in table 1. We observe that the semi-convolutional embedding Ψ_u brings improvements in all considered instance segmentation metrics. The improvement is more significant at higher IoU thresholds which underlines the importance of utilizing position sensitive embedding in order to precisely delineate an instance within an MRCNN crop.

Table 2. Instance-level segmentation comparison using mean APr metric at 0.5 IoU on the PASCAL VOC 2012 validation set

SDS [20]	PFN [34]	DIN [3]	MNC [10]	FCIS [32]	R2-IOS [33]	DML [15]	R. Emb. [28]	BAIS [22]	MRCNN [24]	**Ours**
43.8	58.7	61.7	63.5	65.7	66.7	62.1	64.5	65.7	69.0	**69.9**

Table 3. Average precision (AP) for instance segmentation on PASCAL VOC 2012 reporting the standard COCO evaluation metrics [36]

AP	AP	$AP_{0.5}$	$AP_{0.75}$	AP_S	AP_M	AP_L
Ours	**0.412**	**0.699**	**0.424**	0.107	**0.317**	**0.538**
MRCNN [24]	0.401	0.690	0.412	**0.111**	0.313	0.525

4.3 Instance segmentation

The final experiment compares our method to competition on the instance segmentation task on a standard large scale dataset, PASCAL VOC 2012 [14].

As in the previous section, we base our method on the MRCNN FPN-101 model. Because we observed that the RPN component is extremely sensitive to changes in the base architecture, we employed a multistage training strategy. First, MRCNN FPN-101 model is trained until convergence and then our embeddings are attached and fine-tuned with the rest of the network . We follow [23] and learn using 24 SGD epochs, lowering the initial learning rate of 0.0025 tenfold after the first 12 epochs. Following other approaches, we train on the training set of VOC 2012 and test on the validation set.

Results. The results are given in table 2. Our method attains state of the art on PASCAL VOC 2012 which validates our approach. We further compare in detail against MRCNN in table 3 using the standard COCO instance segmentation metrics from [36]. Our method outperforms MRCNN on the considered metrics, confirming the contribution of the proposed semi-convolutional embedding.

5 Conclusions

In this paper, we have considered dense pixel embeddings for the task of instance-level segmentation. Departing from standard approaches that rely on translation invariant convolutional neural networks, we have proposed semi-convolutional operators which can be easily obtained with simple modifications of the convolutional ones. On top of their theoretical advantages, we have shown empirically that they are much more suited to distinguish several identical instances of the same object, and are complementary to the standard Mask-RCNN approach.

Acknowledgments. We gratefully acknowledge the support of Naver, EPSRC AIMS CDT, AWS ML Research Award, and ERC 677195-IDIU.

References

1. Adam W Harley, Konstantinos G. Derpanis, I.K.: Segmentation-aware convolutional networks using local attention masks. In: Proc. ICCV (2017)
2. Andriluka, M., Stewart, R., Ng, A.Y.: End-to-end people detection in crowded scenes. In: Proc. CVPR (2016)
3. Arnab, A., Torr, P.H.S.: Pixelwise instance segmentation with a dynamically instantiated network. In: Proc. CVPR (2017)
4. Bai, M., Urtasun, R.: Deep watershed transform for instance segmentation. In: Proc. CVPR (2017)
5. Ballard, D.H.: Readings in computer vision: Issues, problems, principles, and paradigms. chap. Generalizing the Hough Transform to Detect Arbitrary Shapes, pp. 714–725. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1987)
6. Chandra, S., Usunier, N., Kokkinos, I.: Dense and Low-Rank Gaussian CRFs Using Deep Embeddings. In: Proc. ICCV (2017)
7. Chen, Y.T., Liu, X., Yang, M.H.: Multi-instance object segmentation with occlusion handling. In: Proc. CVPR (2015)
8. Dai, J., He, K., Li, Y., Ren, S., Sun, J.: Instance-sensitive fully convolutional networks. In: Proc. ECCV (2016)
9. Dai, J., He, K., Sun, J.: Convolutional feature masking for joint object and stuff segmentation. In: Proc. CVPR (2015)
10. Dai, J., He, K., Sun, J.: Instance-aware semantic segmentation via multi-task network cascades. In: Proc. CVPR (2016)
11. Dai, J., Li, Y., He, K., Sun, J.: R-fcn: Object detection via region-based fully convolutional networks. In: Proc. NIPS, pp. 379–387 (2016)
12. De Brabandere, B., Neven, D., Van Gool, L.: Semantic instance segmentation with a discriminative loss function. arXiv preprint arXiv:1708.02551 (2017)
13. Duda, R.O., Hart, P.E.: Use of the hough transformation to detect lines and curves in pictures. Commun. ACM **15**(1), 11–15 (1972)
14. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html
15. Fathi, A., Wojna, Z., Rathod, V., Wang, P., Song, H.O., Guadarrama, S., Murphy, K.P.: Semantic instance segmentation via deep metric learning. CoRR **abs/1703.10277** (2017)
16. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. IJCV **59**(2), 167–181 (2004)
17. Feragen, A., Lauze, F., Hauberg, S.: Geodesic exponential kernels: When curvature and linearity conflict. In: Proc. CVPR (2015)
18. Girshick, R.: Fast r-cnn. In: Proc. ICCV (2015)
19. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proc. CVPR (2014)
20. Hariharan, B., Arbeláez, P., Girshick, R., Malik, J.: Simultaneous detection and segmentation. In: Proc. ECCV (2014)
21. Harley, A.W., Derpanis, K.G., Kokkinos, I.: Learning dense convolutional embeddings for semantic segmentation. In: Proc. ICLR (2016)
22. Hayder, Z., He, X., Salzmann, M.: Boundary-aware instance segmentation. In: Proc. CVPR (2017)
23. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Proc. ICCV (2017)

24. Hu, H., Lan, S., Jiang, Y., Cao, Z., Sha, F.: Fastmask: Segment multi-scale object candidates in one shot. In: Proc. CVPR (2017)
25. Kendall, A., Gal, Y., Cipolla, R.: Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. Proc. CVPR (2017)
26. Kirillov, A., Levinkov, E., Andres, B., Savchynskyy, B., Rother, C.: Instancecut: From edges to instances with multicut. In: Proc. CVPR (July 2017)
27. Kokkinos, I.: Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In: Proc. CVPR (2017)
28. Kong, S., Fowlkes, C.: Recurrent pixel embedding for instance grouping. In: Proc. CVPR (2018)
29. Kong, S., Fowlkes, C.: Recurrent pixel embedding for instance grouping. arXiv (2017)
30. Ladický, L., Sturgess, P., Alahari, K., Russell, C., Torr, P.H.S.: What, where and how many? combining object detectors and crfs. In: Proc. ECCV (2010)
31. Leibe, B., Schiele, B.: Interleaved object categorization and segmentation. In: Proc. BMVC (2003)
32. Li, Y., Qi, H., Dai, J., Ji, X., Wei, Y.: Fully convolutional instance-aware semantic segmentation. In: Proc. CVPR (2017)
33. Liang, X., Wei, Y., Shen, X., Jie, Z., Feng, J., Lin, L., Yan, S.: Reversible recursive instance-level object segmentation. In: Proc. CVPR (2016)
34. Liang, X., Wei, Y., Shen, X., Yang, J., Lin, L., Yan, S.: Proposal-free network for instance-level object segmentation. arXiv preprint arXiv:1509.02636 (2015)
35. Lin, T., Dollar, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: Proc. CVPR (2017)
36. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European conference on computer vision. pp. 740–755. Springer (2014)
37. Liu, R., Lehman, J., Molino, P., Such, F.P., Frank, E., Sergeev, A., Yosinski, J.: An intriguing failing of convolutional neural networks and the coordconv solution. arXiv preprint arXiv:1807.03247 (2018)
38. Liu, S., Jia, J., Fidler, S., Urtasun, R.: Sgn: Sequential grouping networks for instance segmentation. In: Proc. ICCV (2017)
39. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: Proc. ECCV (2016)
40. Ljosa, V., Sokolnicki, K.L., Carpenter, A.E.: Annotated high-throughput microscopy image sets for validation. Nat Methods **9**(7), 637 (2012)
41. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proc. CVPR (2015)
42. Newell, A., Huang, Z., Deng, J.: Associative embedding: End-to-end learning for joint detection and grouping. In: Proc. NIPS (2017)
43. Pinheiro, P.O., Collobert, R., Dollár, P.: Learning to segment object candidates. In: Proc. NIPS (2015)
44. Pinheiro, P.O., Lin, T., Collobert, R., Dollár, P.: Learning to refine object segments. In: Proc. ECCV (2016)
45. Redmon, J., Farhadi, A.: YOLO9000: better, faster, stronger. In: Proc. CVPR (2017)
46. Ren, M., Zemel, R.S.: End-to-end instance segmentation with recurrent attention. In: Proc. CVPR (2017)
47. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: Proc. NIPS (2015)

48. Romera-Paredes, B., Torr, P.H.S.: Recurrent instance segmentation. In: Proc. ECCV (2016)
49. Shi, J., Malik, J.: Normalized cuts and image segmentation. PAMI **22**(8), 888–905 (2000)
50. Silberman, N., Sontag, D., Fergus, R.: Instance segmentation of indoor scenes using a coverage loss. In: Proc. ECCV (2014)
51. Tighe, J., Niethammer, M., Lazebnik, S.: Scene parsing with object instances and occlusion ordering. In: Proc. CVPR (2014)
52. Uhrig, J., Cordts, M., Franke, U., Brox, T.: Pixel-level encoding and depth layering for instance-level semantic labeling. In: German Conference on Pattern Recognition (2016)
53. Wählby, C., Riklin-Raviv, T., Ljosa, V., Conery, A.L., Golland, P., Ausubel, F.M., Carpenter, A.E.: Resolving clustered worms via probabilistic shape models. In: Biomedical Imaging: From Nano to Macro, 2010 IEEE International Symposium on. pp. 552–555. IEEE (2010)
54. Wang, L., Lu, H., Ruan, X., Yang, M.H.: Deep networks for saliency detection via local estimation and global search. In: Proc. CVPR (June 2015)
55. Yurchenko, V., Lempitsky, V.: Parsing images of overlapping organisms with deep singling-out networks. arXiv preprint arXiv:1612.06017 p. 2 (2016)
56. Zhang, Z., Schwing, A.G., Fidler, S., Urtasun, R.: Monocular object instance segmentation and depth ordering with cnns. In: Proc. ICCV (2015)