DISKMASK: FOCUSING OBJECT FEATURES FOR ACCURATE INSTANCE SEGMENTATION OF ELONGATED OR OVERLAPPING OBJECTS

Anton Böhm 1  Nikolaus Mayer 1  Thomas Brox 1

1 Department of Computer Science and Signalling Research Centres BIOSS and CIBSS, University of Freiburg, Germany

ABSTRACT

Deep learning has enabled automated segmentation in a large variety of cases. Instance segmentation of touching and overlapping objects remains an open challenge. We present an end-to-end approach that focuses object detections and features to local regions in an encoder stage and derives accurate instance masks in a decoder. We avoid heavy pre- or post-processing, such as lifting or non-maximum suppression. The approach compares favorably to the current state-of-the-art on three challenging biological datasets.

Index Terms— deep learning, detection, classification, instance segmentation, touching objects, overlapping objects.

1. INTRODUCTION

Instance segmentation is an essential tool for image analysis. In contrast to semantic segmentation which can be covered well by a simple and elegant encoder-decoder approach [1], instance segmentation is less mature. Literature contains a variety of comparatively complex processing pipelines which typically contain critical hyper-parameters.

The currently most popular approach is Mask R-CNN [2] which first solves a bounding-box detection problem to later compute an instance mask for each bounding box. The intermediate bounding box representation is an artificial construct and suboptimal for elongated objects. Rather than to a set of bounding boxes, we decode the image into a feature representation on a regular grid, which focuses the large-scale information of detected objects to local disks (“gateways”). From these gateways, together with their locations, a U-Net [1] decodes detailed segmentation masks using skip-connections which contain high-resolution information.

The object representation via spatially isolated gateways in the image domain allows us to derive an object ordering from their connected components. This resolves touching and overlapping objects without a need for lifting into higher-dimensional space, as in ISOO(v2) [3,4] and TensorMask [5]. The instance ordering via connected components also avoids the use of recurrent networks to focus attention on one object, as in [6]. Feed-forward networks typically train better than recurrent networks which gives our approach an advantage.

2. METHOD

An overview of our method is shown in Fig. 2. We use two stacked networks to process an image $I : \Omega \subset \mathbb{N}^2 \rightarrow \mathbb{R}$. EncoNet encodes an object $O_k$ and its location in $I$ into an object feature pool $f_k$ which we call local pool. DecoNet decodes $f_k$ to the corresponding instance mask $\hat{m}_k : \Omega \rightarrow \{0, 1\}$. Skip-connections transport image features from EncoNet to DecoNet. We denote predicted variables with a hat ($\hat{\cdot}$).

2.1. Reference points and gateways

Reference points

An object’s reference point $p_k$ [4] is a keypoint that can be placed at any reasonable position in the image with reference to this object. We assume that there...
is exactly one reference point per object and that its position stays unchanged (relative to the object) over training. Deviations from this assumption (e.g. noisy data annotation) are treated as variation of the input. As in [4], we couple every object reference point with a disk of a certain radius $r_k$. These disks, denoted as $\Theta$ in Fig. 2, are predicted in the form of logits by EncoNet. The ground truth disk radius $r_k$ is adaptive: it shrinks in the immediate vicinity of another reference point. We use adaptive disks and not just points, because this mitigates the class imbalance problem: it is harder for the network to predict sparse positions than a dense set. Moreover, the disk defines a local region on which object information from the encoder can be focused. Focusing to a single point would be more difficult. The adaptive size still keeps the disks spatially isolated which is important for data with closely located reference points (e.g. overlapping or small objects).

**Gateways** Similar to RoIAlign [2], we extract features from a feature map $F$ and forward them to later convolution layers. RoIAlign pools the features from differently sized objects’ RoIs to a small unified feature map (a pool). In our case, EncoNet creates the pool from its entire input image. Discarding RoIs from the pipeline has advantages. First, the $f_k$ receptive field is not restricted by a RoI: $f_k$ sees a much larger area, which comprises not only the corresponding object $O_k$ but also its surroundings. Second, we do not need RoI-related operations like non-maximum suppression which can fail in case of densely overlapping objects of similar size.

EncoNet is trained to focus $O_k$’s local pool into a compact disk with coordinates $D_k := \{x \in \Omega : \|x - p_k\|^2 < r_k^2\}$ centered at $p_k$. We call a set of 3D coordinates $G_k := D_k \times \{1, \ldots, C_F\}$ on $D_k$ a gateway for $O_k$. Gateways are bottlenecks which control information flow between the networks. Opening a gateway $G_k$ forwards $O_k$’s local pool $f_k$ to DecoNet where it is decoded into a segmentation mask for $O_k$. Closing $G_k$ stops this information flow and only zeroes are forwarded to DecoNet. We use gateways to control the output of DecoNet, forcing it to predict only one instance mask at a time during inference. The gateways allow us to separate touching and overlapping objects. Leaving both gateways of two overlapping objects open produces a combined (binary) mask for those objects, whereas opening only a single object’s gateway produces an individual instance mask. Since the position of $G_k$ stays unchanged relative to the object position, the composite network can unambiguously map from the source $I$ to the target domain $m_k$.

**2.2. Network architecture**

**DiskMask** is a compositional network architecture consisting of two fully convolutional encoder-decoder networks with unpadded convolutions (Fig. 2). EncoNet predicts gateway locations and encodes object-specific information. DecoNet derives a segmentation mask with attention to a specific object instance. The networks communicate via the gateways and inter-net skip connections. A gateway’s content and position provide object-specific information. Skip-connections between EncoNet and DecoNet propagate high-resolution information to DecoNet’s convolutional layers, helping them recover high-resolution details after going through a bottleneck (gateway).

**DecoNet** is a modified U-Net [1] with half the number of channels (e.g. $C=512$ at the lowest feature map resolution). We replaced the first block that consists of two $3 \times 3$ convolution operations (with ReLU activations) and a max-pooling operation by a single $5 \times 5$ convolution operation with stride 2 and a $C=64$ output feature map. Moreover, we do not use drop-out and use a pixel-wise sigmoid cross entropy loss for training after applying a 1x1 reduction layer with one channel output. The network takes a feature map $F$ of dimension $324 \times 324 \times 64$ with activated object features and yields a segmentation mask with dimensions $132 \times 132 \times 1$.

**EncoNet** takes an image crop of size $508 \times 508 \times 3$ as input and yields two tensors of sizes $324 \times 324 \times 64$ and $324 \times 324 \times 2$ as output. The architecture is as in DecoNet, but we remove the intra-net skip connections, forcing the network to pass the signal through all layers. We increase the number of channels in the expanding path, setting all convolution outputs to $C=256$,
except for the last block where we set \(C=64\). We replicate the last block, resulting in a network with two heads. The first head ends in a \(1 \times 1\) convolution layer that defines the dimensionality of a feature map \(F\) (we use \(C_F = 64\)). The second head ends in a \(1 \times 1\) convolution layer mapping the output to disk logits \(\hat{\Theta}\) with the number of channels corresponding to the maximum number of object classes in a dataset (here, \(C=2\)). The network is trained with weighted soft-max cross entropy loss \(\mathcal{L}\) on the second head’s output.

**Skip connections between EncoNet and DecoNet** pull the feature maps from every block’s end in the EncoNet contraction path to the DecoNet block’s beginning. As in U-Net \([1]\), we crop the EncoNet feature maps before their concatenation in DecoNet. Cropping is necessary since we use unpadded convolutions to reduce computational overhead. We process images as overlapping tiles if the image size exceeds \(132 \times 132\).

**During inference** (see Fig. 2 for step numbers), (1) a forward pass with EncoNet yields a feature map \(F\) with object features and object disk logits \(\hat{\Theta}\). After binarizing the logits and applying a \(3 \times 3\) median filter, we find disk positions \(D_{\{1, \ldots, \hat{N}\}}\) and the number of objects \(\hat{N}\) by extracting connected components in \(\hat{\Theta}\). (2) \(\hat{\Theta}\) is replicated along its channel dimension \(C_F\) at times. This results in 3D segmentation masks \(\Theta_{3D}\) within gateway positions \(G_{\{1, \ldots, \hat{N}\}}\). (3) A single gateway \(G_i\) in \(\Theta_{3D}\), with \(i \in \{1, \ldots, \hat{N}\}\) at a time is activated while zeroing the others. (4) Multiplying the feature map \(F\) and \(\Theta_{3D}\) with the active gateway yields the object local pool \(f_i\). (5) DecoNet synthesizes the object segmentation mask \(\hat{m}_i\) from \(f_i\) and the inter-net skip connections. Steps 3–5 are repeated for all gateways.

Practically, we use an overlap-tile strategy \([1]\). An object’s gateway is activated in multiple tiles if it is located in the tiles’ overlap. In this case, the object’s mask may need to be stitched together from multiple tile outputs, so gateways must be assigned unique IDs. To achieve this, we first run EncoNet (corresponding to step (1)) on tiles on the entire image, extract disk positions and assign IDs via connected components, then reprocess the image with the composite DiskMask network.

### 2.3. Training

We train the composite network end-to-end for 600k iterations with batch size 1 and the ADAM solver \([7]\) (\(\beta_1 = 0.9, \beta_2 = 0.999\)). We start with a learning rate of \(1e^{-4}\) and reduce to \(1e^{-5}\) after 300k iterations.

EncoNet is trained on a segmentation map \(\Theta\) which contains disk segments of radius \(r_k\). Similar to \([1]\), we weight the space between disks pixelwisely to ensure their spatial isolation. To minimize the effect of disk shrinking on the loss, we additionally weight the disk area by \(w_k := r_{\text{max}} - r_k + 1\).

The maximal disk radius \(r_{\text{max}}\) is a hyperparameter which we set to \(r_{\text{max}} = 9\) px in our experiments.

After the gateways are set up, we train DiskMask end-to-end, passing the signal through them. Compared to inference, we open not one but a subset \(A \subset \{1, \ldots, \hat{N}\}\) of gateways where each has a 50% chance of being selected. We exclude the gateways from \(A\) if they lay outside or on the DecoNet’s input borders. Activating multiple gateways accelerates the training process and simultaneously solves the gateway-to-instance assignment problem. The gateways are derived from the ground truth object disks. To prevent DecoNet from overfitting to a certain gateway size, we randomly shrink the radii by \([0, r_{\text{max}} - 2]\) in each iteration. We train DecoNet on the ground truth segmentation masks \(m_{2D}\) of projected object silhouettes. We project them by taking the maximum value of the active object masks \(m_{2D}(x) = \max\{\{m_i(x)\}\}\), with \(i \in A\). This results in a binary segmentation map (“active instance masks” and “background”). Objects which do not fit into the EncoNet’s input do not contribute to the training loss, since their reference points might reside outside the input.

### 3. EXPERIMENTS

**3.1. Datasets, baselines and metrics**

We evaluate our method on three public datasets:

1. **OSC-ISBI** from “The Second Overlapping Cervical Cytology Image Segmentation Challenge - ISBI 2015” \([8]\) (Fig. 2 contains densely overlapping objects of similar size. We use the same data preprocessing, augmentation, and metrics as in \([4]\). We place reference points at the centers of cell nuclei. We report the performance on the test set (with 9 images). Our method improves the state-of-the-art in detection (measured by object-based false negative rate \(\text{FNo}\)) while being on par in terms of segmentation (dice coefficient \(\text{DC}\), pixel-based true positive \(\text{TPp}\) and false positive \(\text{FPp}\) rates). Compared to the second-best approach ISOOv2 \([4]\), we
thresholds (T): how accurate the predictions are, we also report single IoU-precision (AP) on 10 IoU-thresholds from 0.5 to 0.95. To show ground truth masks of original size. For comparison, we retraction the mean intensity value and train the network on the lar to our method, we train it from scratch, we do not sub-

lengths of square anchors at all 5 resolution levels. Simi-

tions are important for detailed structures.

- ISOOV2 [4] was trained with a fixed shear angle $\gamma = 23^\circ$ using the same training protocol as in our work. We omit the low-IoU fallback to reference point backprojection [3] as the bounding boxes do not describe the worms’ shapes well.

- We take a TensorFlow [16] implementation of Mask R-CNN FPN-50 [17] and adapt some hyper-parameters to the dataset. We set the nms-threshold to 0.5 and double the lengths of square anchors at all 5 resolution levels. Similar to our method, we train it from scratch, we do not sub-
tract the mean intensity value and train the network on the ground truth masks of original size. For comparison, we report COCO evaluation results [18] including average precision (AP) on 10 IoU-thresholds from 0.5 to 0.95. To show how accurate the predictions are, we also report single IoU-thresholds (T): $AP_{50}$, $AP_{75}$ and $AP_{90}$ standing for $T=0.5$, $T=0.75$ and $T=0.90$ respectively. We also report AP for small $AP_S$ and middle-sized objects $AP_M$. In all three methods, we evaluate the final performance on a test set with 25 images, taking the best performing iteration on the validation set (with the remaining 25 images). Tab.2 summarizes the results: All methods show good detection performance ($T=0.5$). Mask R-CNN drops significantly after $T=0.8$, resulting in a poor overall AP score. Fig.3 shows a qualitative comparison.

We also report the inference times for this dataset, which we compute using a NVIDIA GTX1080Ti GPU and the Caffe framework [19] with a MATLAB wrapper. We need ca. 1.1s to process an image of size $520 \times 696 \times 3$ without objects, and on average ca. 1.9s for an image with 14 objects.

3. LSC. To rank our method among others, we partici-

pate in the CVPPP’17 Leaf Segmentation Challenge (LSC) [20–22]. We use the A1 subset which contains 128 images with non-overlapping instance masks. We normalize the data to the range [0, 1] and split it into training and validation sets with 102 and 26 images, respectively. We train the network from scratch with data augmentation (flipping, cropping, rotation and scaling in the range [1,1.5]). After training, we estimate the best performing training iteration using the validation set. We report our performance (evaluated by [23]) on 33 test images using the challenge metrics: absolute difference in count (|DiC|) and symmetric best dice (SBD). Our method outperforms [13] (see Tab.1 “Ward et al. (synth)”), where the authors train Mask R-CNN not only on real but also synthetic data, which gives them a performance boost of 3 SBD-points. The qualitative results are depicted in Fig.1.

3.2. Ablation experiments

We perform three experiments on C.Elegans (see Tab.3): 1. “w/o inter-net skip”: we remove inter-net skip connections. Without them, all information for an object mask must be passed through the gateway bottleneck. This does not affect detection performance ($T=0.5$) much, but for higher IoU thresholds performance drops substantially: inter-net connections are important for detailed structures.

2. “single rand pnt”: we create $f_k$ from a single point which is randomly sampled from within the predicted disk. This is the extreme test for robustness against gateway size and location. DecoNet is able to make a good segmentation mask even when the location and size of the gateway strongly deviates from the optimal one.

3. “shallow DecoNet”: we extremely reduce DecoNet’s capacity, dividing its overall number of channels by 16 (remaining e.g. with $C=32$ at the lowest resolution): performance drops but is still ahead of Mask-RCNN.

3.3. Conclusion

We presented a method for instance segmentation that combines all processing in a unified encoder-decoder approach. It achieves strong performance on all three datasets without pre-training, domain adaptation, or modelling of synthetic data.
4. REFERENCES


