Instance Segmentation of Dense and Overlapping Objects via Layering

Long Chen[1]
long.chen@lfb.rwth-aachen.de
Yuli Wu[1]
yuli.wu@lfb.rwth-aachen.de
Dorit Merhof[2,3]
dorit.merhof@ur.de

[1] Institute of Imaging & Computer Vision
RWTH Aachen University
Aachen, Germany

[2] Faculty of Informatics and Data Science,
University of Regensburg, Germany

[3] Fraunhofer Institute for Digital Medicine
MEVIS, Bremen, Germany

Abstract

Instance segmentation aims to delineate each individual object of interest in an image. State-of-the-art approaches achieve this goal by either partitioning semantic segmentations or refining coarse representations of detected objects. In this work, we propose a novel approach to solve the problem via object layering, i.e. by distributing crowded, even overlapping objects into different layers. By grouping spatially separated objects in the same layer, instances can be effortlessly isolated by extracting connected components in each layer. In comparison to previous methods, our approach is not affected by complex object shapes or object overlaps. With minimal post-processing, our method yields very competitive results on a diverse line of datasets: *C. elegans* (BBBC), Overlapping Cervical Cells (OCC) and cultured neuroblastoma cells (CCDB). The source code is publicly available [1].

1 Introduction

Different from semantic segmentation, which pays no attention to the individual objects, instance segmentation aims not only to associate every pixel of an image with a class label but also to delineate objects of the same class as individuals. This task becomes more challenging, when objects are densely located or even overlapping with each other.

A prevalent top-down class of instance segmentation methods is detection-based, which firstly localize an instance with a coarse shape representation and then refine the shape in an additional step. As a paradigm, Mask-RCNN [10] refines bounding boxes obtained from Region-based Convolutional Neural Networks (R-CNN) [8, 20]. Relying on non-maximum suppression (NMS) to remove duplicate predictions, detection-based methods become less competent in the cases of dense clusters and overlapping objects. A finer polygon representation approach was proposed by [22] to reduce false suppressions. However, NMS has a methodological flaw for objects that inherently overlap. Moreover, many objects, especially

[1] https://github.com/looooongChen/instSeg/

in the biomedical domain, cannot be well-approximated by tractable shape representations, such as bounding boxes and star-convex polygons.

Free from false suppression and coarse shape approximation, alternative bottom-up approaches obtain instances by grouping pixels [1, 2, 7]. DCAN [2] predicts the object boundary explicitly and groups connected pixels that are separated by these boundaries as instances. However, this approach is sensitive to broken boundaries. A few misclassified pixels can lead to erroneous merging of adjacent objects. Graph partition based on pixel-pair affinity [7] and the watershed transform [1] are more robust in terms of grouping, but the pipeline as a whole relies heavily on post-processing, with the learning model only optimized with intermediate results. In addition, grouping-based approaches are inherently incapable of handling overlapping objects, where one pixel may belong to more than one object.

Recent research [3, 5, 12, 23] introduces pixel embedding for grouping. In these approaches, a deep neural network is trained to map pixels into an embedding space, in which pixel embeddings from the same object are close, while those from different objects, especially adjacent objects, are apart. Then, pixels are grouped in the embedding space with low-level clustering algorithms, such as Mean Shift [4] and DBSCAN [6]. Although pixel embedding based grouping has proven to be more robust, it still can not avoid the shortcomings of grouping-based methods: the reliance on post-processing, the optimization of intermediate results, and the incapability to segment overlapping objects.

Our work is partially inspired by the pixel embedding approach. To alleviate the aforementioned limitations, we propose to train a more structured embedding space. Specifically, we restrict one object to "live" in one dimension. Correspondingly, embedding vectors will be one-hot in overlap-free areas and have more than one active digits at locations where an overlap of objects occurs (Figure 1). We figuratively call this process of objects being distributed to different dimensions "layering", and use the terms "dimension" and "layer" interchangeably in the following context. Since our training loss penalizes adjacent objects having the same embedding vector, only spatially separated objects can be assigned to the same layer, eliminate crowding and overlapping of objects. Therefore, instances can be effortlessly obtained by finding connected components in each layer.

The main contributions of our work are as follows: We propose (1) a novel approach to eliminate crowdedness and overlap of instances by layering objects into different output layers; (2) an approach of *spontaneous* object layering through deep model learning; (3) a concise and effective framework for segmentation of densely distributed objects without data-specific post-processing efforts.

To our best knowledge, our work is the first pixel embedding based approach that does not require explicit pixel clustering, and is capable of handling object overlap. Competing with several state-of-the-art approaches, our method yields comparable or better results on a diverse line of datasets: *C. elegans* [16] (BBBC), overlapping cervical cells [17, 18] (OCC) and cultured neuroblastoma cells [24] (CCDB).

2 Proposed Method

2.1 Overview

Our model consists of two output branches: the foreground branch and the layering branch (Figure 1). The foreground segmentation, as an auxiliary, excludes background pixels from

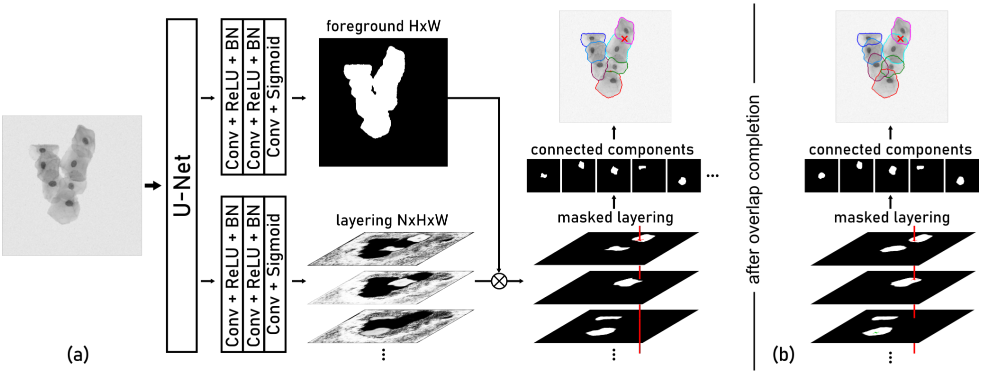

Figure 1: Our approach eliminates the object crowding and overlapping by distributing adjacent objects into different layers. (a) To achieve this, each foreground pixel is assigned by drawing pixels of the same object into the same layer and pushing pixels of adjacent objects into different layers (Section 2.2). Areas where overlap occurs are ignored in this training phase. (b) After object layering converges, overlapping areas are trained to complete the object (Section 2.3). At locations where overlap occurs (the red cross and line), more than one layer will give foreground prediction.

further processing, while the layering branch is devoted to separating foreground instances.

The foreground branch uses a 1-channel convolutional layer with *Sigmoid* activation for output. The output layer can also be set to multi-channel and *Softmax* activated in the case of more than one semantic category. The layering branch has N output channels with the *Sigmoid* activation, each of which is trained to only contain spatially separated objects. Since objects in the same layer exhibit no contact or overlap, they can be effortlessly isolated by extracting connected components.

We train the model in two phases: layering training and overlap completion. In the first phase, under the supervision of our proposed layering loss (Section 2.2), the model learns to assign foreground pixels to one of the layers, maintaining the restriction that neighboring objects should be located in different layers. We only consider image areas where no objects overlap in this phase, since the layering loss exclusively chooses one layer for one pixel. After the layering training converges, we generate N binary masks by ordering each object mask into one of the N layers, based on the layering results of the object's overlap-free part. It is worth mentioning that more than one layer will be positive at locations where an overlap of objects occurs. A *Dice*-like loss (Section 2.3) is then computed with the generated masks and further included in the second training phase. The overlapping area is explicitly trained in this phase, to complete the intact object.

In summary, the model is trained in two phases with the following loss:

$$L = [L_{foreground}]_{S}^{1,2} + [L_{layering}]_{S_{fn}}^{1,2} + [L_{overlap}]_{S_f}^{2}, \tag{1}$$

where the subscript of $[\cdot]$ denotes on which area a loss term is computed: S, S_f and S_{fn} represent three progressively smaller areas, namely, the whole image, the foreground and the foreground without object overlap. The superscript indicates in which training phase the loss term is included. The standard *Crossentropy* is used as foreground training loss $L_{foreground}$. Details of $L_{layering}$ and $L_{overlap}$ are depicted in the following two sections.

2.2 Layering Loss

The object layering is achieved by drawing pixels of the same object together into the same dimension with an attracting loss term L_{attr} and pushing away neighboring objects into different dimensions with a repelling loss term L_{rep}. Our loss is constructed aroud the cosine similarity $D(\mathbf{e_i}, \mathbf{e_j}) = \frac{\mathbf{e_i}^\top \mathbf{e_j}}{\|\mathbf{e_i}\|_2 \|\mathbf{e_j}\|_2}$ [3, 23], which is zero when two vectors are located in two othogonal spaces. The operation $\|\cdot\|_2$ computes the $\mathcal{L}^2$ norm.

We push adjacent objects into the orthogonal space of each other, while non-adjacent objects can stay in the same cluster. Assuming that there are C objects ($\{O_i | i = 1, 2, ..., C\}$) in an image, we represent the overlapping and overlap-free part of an object with O_i^o and O_i^n, respectively. The attracting and repelling terms can thus be formulated as:

$$L_{attr} = 1 - \frac{1}{\sum_{i=1}^{C} |O_i^n|} \sum_{i=1}^{C} \sum_{p \in O_i^n} D(\mathbf{e_p}, \mathbf{u_i})^2 \,, \tag{2}$$

$$L_{rep} = \frac{1}{C} \sum_{i=1}^{C} \frac{1}{|Adj(O_i)|} \sum_{j \in Adj(O_i)} D(\mathbf{u_i}, \mathbf{u_j})^2 \,, \tag{3}$$

where e_p indicates the embedding vector of pixel p and $\mathbf{u_i} = \frac{1}{|O_i^n|} \sum_{p \in O_i^n} \mathbf{e_p}$ is the mean embedding of overlap-free part of the i-th object. $Adj(O_i)$ represents the set of adjacent objects to object O_i, whose shortest distances to O_i are less than a threshold t ($t = 15$ pixels in this work). The operator $|\cdot|$ returns the element number of a set. For example, $|O_i^n|$ is the pixel number of area O_i^n and $|Adj(O_i)|$ is the number of adjacencies of object O_i.

Training with the attracting term L_{attr} and the repelling term L_{rep} specifies embedding vectors to locate in mutually orthogonal spaces, but it does not guarantee that the vectors only "live" in one layer (dimension), i.e. they are distributed on the standard axes (see Figure 2). Therefore, we introduce the sparse term L_{sparse}, which imposes preferences for the vector whose digits, except for a single one, are suppressed to 0:

$$L_{sparse} = 1 - \frac{1}{\sum_{i=1}^{C} |O_i^n|} \sum_{i=1}^{C} \sum_{p \in O_i^n} \max\left(\frac{\mathbf{e_p}}{||\mathbf{e_p}||_2}\right) \,, \tag{4}$$

where the operator $\max(\cdot)$ takes the value of the maximal digit.

The layering loss consists of the three terms above:

$$L_{layering} = L_{attr} + L_{rep} + \lambda L_{sparse} \,, \tag{5}$$

where λ is a weighting constant, and we use $\lambda = 0.1$ in this work. It is worth mentioning that only the overlap-free foreground part is involved in the calculation of $L_{layering}$.

2.3 Overlap Completion

After layering training, the overlap-free parts of objects are assigned to one of the layers, while the overlapping parts remain untrained. Generally, the untrained pixels from overlapping areas will randomly belong to one dimension to which one of the objects that overlap belongs (Figure 2b). Accordingly, a random boundary between objects will be formed, as illustrated in Figure 1a.

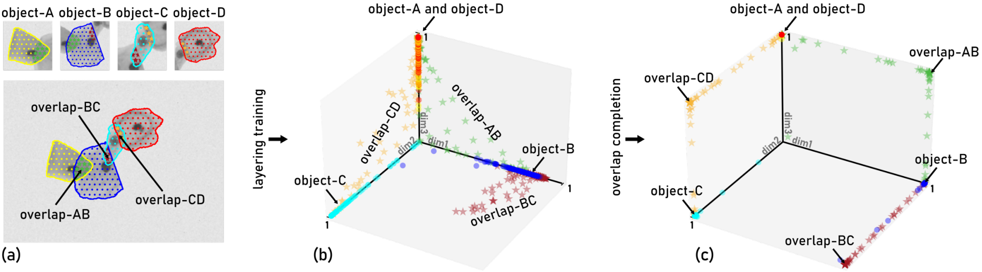

Figure 2: Visualization of trained embedding space. (a) Four objects with three overlapping areas are present in this example. (b) After layering training, the object parts that do not overlap are ideally distributed in different dimensions (layers). Spatially separated objects, such as A and D, can exist in the same dimension. Pixels of overlapping areas, untrained in this phase, live randomly in the plane composed of axes to which the overlapping objects belong. (c) After overlap completion, pixels of the overlapping area will congregate to the point $(1,1)$, indicating a foreground area in both layers. And, pixels of the non-overlap part are also more compactly distributed and closer to 1.

To train the region with overlap, we firstly generate N binary masks, denoted by $S \in \{0,1\}^{H\times W\times N}$, by placing the silhouette of each object into one of the N masks based on the layering results of the object's overlap-free part. Indexing the stack of N layers at different pixel positions p with $S_p \in \{0,1\}^N$, we define the following *Dice*-like [19] loss for overlap completion training:

$$L_{overlap} = 1 - 2\frac{\sum_{i=1}^{C}\sum_{p\in O_i}\mathbf{e_p}^\top\mathbf{S_p}}{\sum_{i=1}^{C}\sum_{p\in O_i}(\mathbf{1}^\top\mathbf{e_p}+\mathbf{1}^\top\mathbf{S_p})}\,. \tag{6}$$

The dot product between binary vector $\mathbf{e_p}$ and $\mathbf{S_p}$ on the numerator of $L_{overlap}$ is analogous to the intersection in *Dice* loss, whose value is maximized when $\mathbf{e_p}$ and $\mathbf{S_p}$ are exactly the same. The denominator sums up all digits of vector $\mathbf{e_p}$ and $\mathbf{S_p}$, which is equivalent to the sum in the *Dice* loss. The overlap completion loss $L_{overlap}$ is applied to the foreground, including the overlapping and non-overlapping areas. The vector $\mathbf{S_p}$ is one-hot in areas without overlap, while it has more than one non-zero digits at locations where overlap occurs.

2.4 Post-Processing

Since each of the N layers only contains spatially separated objects, high-quality instance segmentations can be obtained with minimal post-processing effort, only involving simple and computationally efficient operations. Our post-processing steps are also general for all instance segmentation tasks, without utilizing any prior knowledge specific to a certain dataset. Detailed steps are listed in Algorithm 1. It worth mentioning that line 7-8 can be ignored if there is no object overlap for certain tasks.

The post-processing requires two trivial parameters: a threshold τ and a minimal object size S_{min}. The value τ is used for foreground thresholding (lines 1 and 7 in Algorithm 1). The minimal object size S_{min} is responsible for eliminating small noisy objects (lines 2 and 11 in Algorithm 1). We use $\tau = 0.5$ and $S_{min} = 250$ for all experiments in this work.

Algorithm 1 Post-processing to obtain object segmentations

```
Require: (raw prediction) foreground $F_{raw} \in (0,1)^{H \times W}$, layering $L_{raw} \in (0,1)^{H \times W \times N}$
Require: (parameters) threshold $\tau$, minimal object size $S_{min}$
threshold $F_{raw}$ with value $\tau$ to get foreground $F$
remove connected components in $F$ that are smaller than $S_{min}$
initial $N$ empty binary layering masks $L \in \{0,1\}^{H \times W \times N}$
for each pixel location $i \in \{1,2,...,H\}, j \in \{1,2,...W\}$ and layer $k \in \{1,2,...,N\}$ do
    if $L_{raw}(i,j,k)$ is the largest in $L_{raw}(i,j,:)$ then
        set $L(i,j,k)$ to 1
    else if $L_{raw}(i,j,k)$ is larger than $\tau$ then        ▷ omit if no overlap exists
        set $L(i,j,k)$ to 1                                    ▷ omit if no overlap exists
    end if
end for
take all connected components larger than $S_{min}$ in each layer as objects
```

3 Experiments and Results

3.1 Implementation

While our approach is not tied to a particular network architecture, we perform experiments with UNet [21]. In our implementation, we apply batch normalization after each convolutional layer. In addition, we experiment with a UNet-S and a UNet-L, which have, respectively, one less and one more convolutional block on the contracting and expansive path than the original UNet (denoted as UNet-M), to investigate the effect of receptive field size.

The foreground branch and the layering branch share the same feature map from the UNet backbone. Two 3x3 covolutional layers with 64 features are added before the output layer to avoid "feature conflict". We use an N-channel convolutional layer ($N = 8$ in this work) and a 1-channel convolutional layer with the *sigmoid* activation for the final output.

The model was trained with *RMSprop* optimizer [11] with a learning rate of $1e^{-4}$ exponentially decayed to 0.9 every 10000 steps. The training dataset was randomly split into 90% and 10% for training and validation. The layering training phase lasted 1000 epoches. Then, the overlap completion training continued from the best model of layering training for another 500 epoches. The best model of overlap completion training was used as the final model for evaluation. The "best" models are chosen based on validation results.

3.2 Datasets and Evaluation Metrics

For the evaluation, we use three biomedical image datasets containing a population of objects with different shapes, different degrees of density and overlap. All datasets were augmented using random horizontal and vertical flips, random rotation, random gamma $\gamma \in (0.5, 2)$ correction transform and elastic deformation.

BBBC010 contains 100 bright-field microscopic images of live/dead *C. elegans* [16]. The *C. elegans* are slender, bilaterally symmetrical objects in curved or ring-shaped poses, which may cross over others in this data set. Subset D was left for evaluation and the rest for training. We crop the image and a 448x448 pixel area containing objects remains.

OCC2014 is an EDF (extended depth of field) image collection of overlapping cervical cells from Pap smears [17, 18], consisting of 16 real and 945 synthesized images. The cells

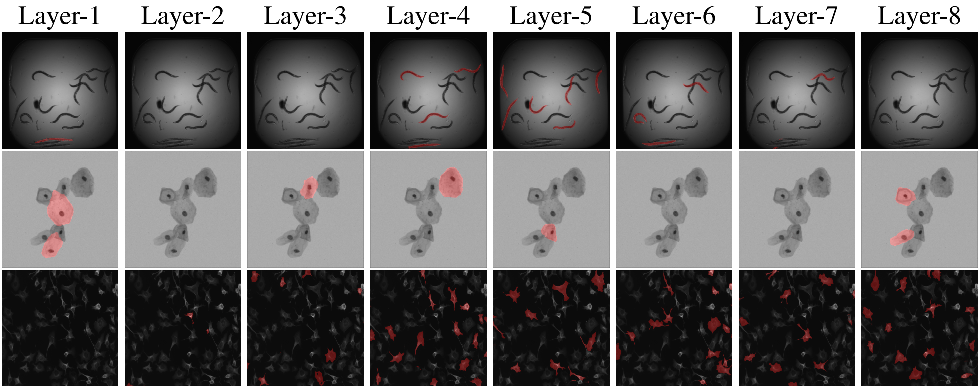

Figure 3: Activation map of each layer: results on the datasets BBBC, OCC and CCDB from top to bottom. Spatially adjacent objects are distributed to different layers, eliminating crowding and overlapping of objects. Meanwhile, not all layers contain objects, which suggests that, by keeping only adjacent objects separate, layers are fully utilized.

are roundish and densely clustered, and the overlap can be larger than 50% of the object size. In our experiments, the original training set and half of the test set served for training. Evaluation was conducted on the other half. The images are resized to 320x320 pixels.
CCDB6843 contains 100 wide field fluorescent images of cultured neuroblastoma cells collected by [24]. The cells are densely located and irregularly shaped. We randomly chose 24 images as the test set. All images are resized to 448x448 pixels.

In our evaluation, a predicted object I_{pred} is considered to be a successful match (true positive TP_t) if its intersection over union $IoU = \frac{I_{pred} \cap I_{gt}}{I_{pred} \cup I_{gt}}$ with a ground truth object is larger than a given threshold t, while unmatched predictions and ground truth objects are counted as false positive (FP_t) and false negative (FN_t). Using these values, a measure of detection accuracy can be calculated: $AP_t = \frac{TP_t}{TP_t + FP_t + FN_t}$. By passing from loose to strict thresholds, the segmentation accuracy is also reflected. To better quantify the segmentation performance, we also calculate the Aggregated Jaccard Index (AJI) [13].

3.3 Competing Methods

UNet-2/3: We train UNet [21] models as semantic segmentation tasks with a two labels (object and background) and a three labels (object, boundary and background) setup. In terms of network structure, UNet is the least different from our model: UNet output all objects with a single image plane, whereas our model has several containing layered objects.
Mask-RCNN: Mask-RCNN [10] localizes objects by proposal classification and non-max suppression (NMS). Then, segmentation is performed on each object bounding box. The NMS threshold was set to 0.7 for all experiments.
StarDist: Without an explicit segmentation step, StarDist [22] obtains object masks by combining distances from the center to the boundary along different directions. We used 32 radial directions in our experiments.

All methods except Mask-RCNN were trained from scratch, while the Mask-RCNN model was fine-tuned on a COCO pretrained model [15] with the ResNet-101 backbone [9]. For the 3-label UNet and StarDist, we used a pseudo-boundary as an approximated separa-

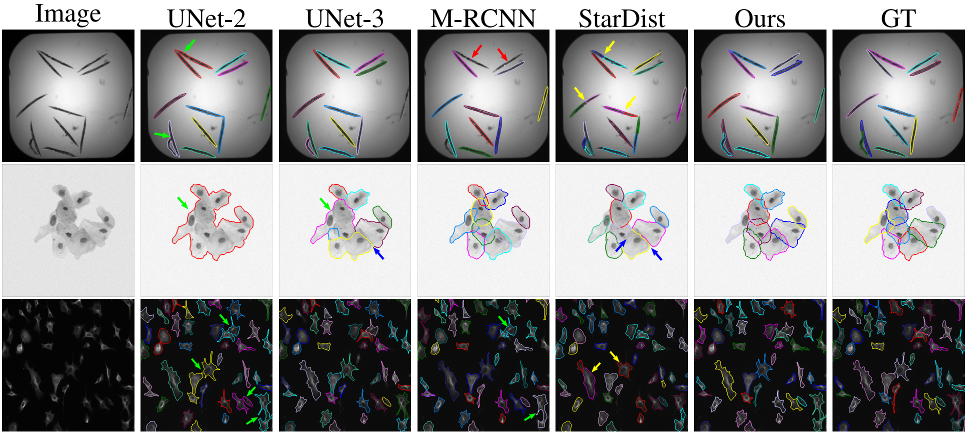

Figure 4: Qualitative segmentation results on the datasets BBBC, OCC and CCDB from top to bottom. A few typical errors are marked with arrows: false suppression (red), merged objects (green), inaccurate shape (yellow) and incapability to handle overlap (blue).

tion of overlapping regions, obtained by skeletonizing [14] the binary mask of overlapping regions together with boundaries.

3.4 Results and Discussion

For a careful interpretation, we discuss the methods from the following aspects: the handling of object shape, touching objects, object overlap and the effect of high object density.
Shape: Using radial directions, StarDist has difficulty in reconstructing slender shapes and boundaries with fine structure. This is reflected by the broken body of *C. elegans* and the roundish approximation of the neuroblastoma cells (Figure 4). Other methods, which perform pixel-wise segmentation, are less affected.
Touching objects: Since it does not take clustered objects explicitly into consideration, the 2-label UNet performs suboptimally on all datasets due to false fusion, while others are object-aware to different degrees under different conditions.
Object overlap: Since UNet and StarDist naturally assign one pixel to one object, an overlapping area can only be handled with approximated boundaries. In case of less severe overlap, such as on the BBBC and CCDB datasets, UNet (3-label) still achieves very good results, although its performances drops significantly when the overlap ratio increases on the dataset OCC.
Object density: Mask-RCNN suffers from the false suppression of NMS, when the bounding boxes of two objects overlap with a large ratio, as, for example, in the case of the two parallel and close *C. elegans* in Figure 4. We also find that Mask-RCNN has difficulty in distinguishing two closely located irregular-shaped objects, such as the neuroblastoma cells in Figure 4.

As described above, segmentation errors, including false suppression, merged objects, inaccurate shape and the incapability to handle overlap (Figure 4), occur on different approaches, depending on the data characteristics. By contrast, our method is more robust in all of these aspects, and, therefore, achieves the best and, evidently, better results on the dataset BBBC and CCDB (Table 1). On the dataset OCC, the performance of our approach

Table 1: Quantitative evaluation. Average precisions (AP_t) under different *IoU* thresholds, mean values of average precision *AP* over these thresholds, and the Aggregated Jaccard Index (*AJI*) are reported. The best two results are shown in bold, and the best is underlined.

Data and Methods		$AP_{0.5}$	$AP_{0.6}$	$AP_{0.7}$	$AP_{0.8}$	$AP_{0.9}$	*meanAP*	*AJI*
BBBC	UNet-2	.5455	.4645	.4497	.4212	.2355	.4233	.5327
	UNet-3	.8863	.8197	.7412	**.5795**	**.2717**	**.6597**	**.7786**
	StarDist	.3098	.1410	.0372	.0011	.000	.0978	.4499
	MRCNN	**.8953**	**.8629**	**.8111**	.5305	.0382	.6276	.7580
	Ours	**.9357**	**.9188**	**.8648**	**.7606**	**.2904**	**.7541**	**.8442**
OCC	UNet-2	.1548	.1303	.1129	.1055	.1048	.1217	.2058
	UNet-3	.7010	.6071	.5097	.3767	.1950	.4779	.5802
	StarDist	.6556	.5566	.4346	.2970	.1547	.4197	.6927
	MRCNN	**.9277**	**.9181**	**.8870**	**.8117**	**.5564**	**.8202**	**.8412**
	Ours	**.9230**	**.8768**	**.8007**	**.6788**	**.4349**	**.7429**	**.8353**
CCDB	UNet-2	.3698	.3360	.3049	.2763	.2228	.3020	.1185
	UNet-3	.7307	**.6774**	**.6210**	**.5153**	**.2838**	**.5656**	**.7148**
	StarDist	**.7428**	.6532	.4958	.2685	.0326	.4386	.6903
	MRCNN	.6248	.5691	4888	3476	.0763	.4213	.5842
	Ours	**.7968**	**.7467**	**.6767**	**.4889**	**.2230**	**.5864**	**.7601**

Table 2: Performance of models with different sizes with/without overlap completion (OC)

meanAP		UNet-S	UNet-M	UNet-L
BBBC	w/o OC	.4655	.6670	.7120
	w/ OC	.5406	.7162	**.7541**
OCC	w/o OC	.4359	.5273	.5441
	w/ OC	.4815	.6841	**.7429**
CCDB	w/o OC	.5229	.5569	.5743
	w/ OC	.5234	.5614	**.5864**

is only marginally worse than Mask-RCNN in terms of the Aggregated Jaccard Index (*AJI*).

By chance, through some examples from experiments, we found certain layers reveal obvious semantics. For example, common morphological features, such as body orientation, are observed in layers predicted by the BBBC models (Figure 5). Another feature commonly exploited by all models is the relative position. For example, the leftmost protruding objects are active in one common layer of OCC2014 predictions (Figure 5). To capture high-level morphological features and sophisticated relative positions, especially in very crowded cases, an adequately large receptive field (RF) is required. To verify our analysis, we train three UNet variants with RF of 108, 220 and 444 pixels (Section 3.1). The experiments shows very significant differences: 39.49%, 54.29% and 12.04% improvement from UNet-S to UNet-L on the datasets BBBC, OCC and CCDB in terms of *meanAP* (Table 2).

In addition, we compared the performance before and after overlap completion (Table 2). On the dataset BBBC and CCDB, the difference is relatively small: 5.91% and 2.11%, taking the best model UNet-L as an example, since the objects are only slightly overlapped. By contrast, the segmentation of severely overlapping cervical cells gains a 36.54% performance boost.

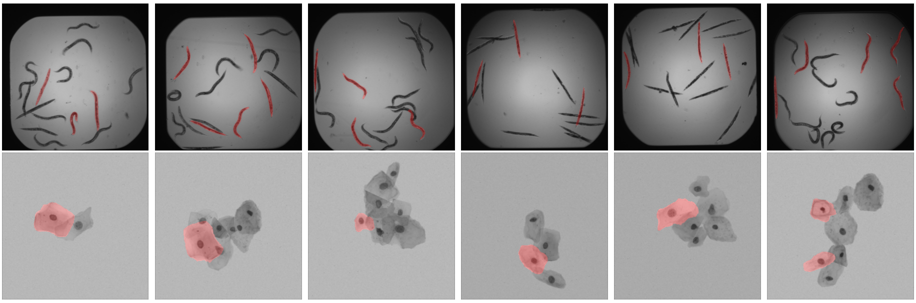

Figure 5: Example activation maps of certain layers. *C. elegans* with the vertical body orientation are active in the 4th layer of the BBBC data (first row). In the 8th layer of OCC data (second row), most leftmost protruding cells are active.

4 Conclusion and Outlook

Our proposed approach can successfully layer touching and overlapping objects into different image layers. By grouping spatially separated objects in the same layer, our method simplifies post-processing and improves the accuracy of instance segmentation, yielding very competitive results on a diverse line of data sets. Our future work will focus on understanding the layering mechanism.

References

[1] Min Bai and Raquel Urtasun. Deep watershed transform for instance segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5221–5229, 2017.

[2] Hao Chen, Xiaojuan Qi, Lequan Yu, and Pheng-Ann Heng. Dcan: deep contour-aware networks for accurate gland segmentation. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2487–2496, 2016.

[3] Long Chen, Martin Strauch, and Dorit Merhof. Instance segmentation of biomedical images with an object-aware embedding learned with local constraints. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 451–459. Springer, 2019.

[4] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 24(5):603–619, 2002.

[5] Bert De Brabandere, Davy Neven, and Luc Van Gool. Semantic instance segmentation for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 7–9, 2017.

[6] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.

[7] Naiyu Gao, Yanhu Shan, Yupei Wang, Xin Zhao, Yinan Yu, Ming Yang, and Kaiqi Huang. Ssap: Single-shot instance segmentation with affinity pyramid. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 642–651, 2019.

[8] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation, 2014.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[10] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn, 2018.

[11] Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. *Cited on*, 14(8):2, 2012.

[12] Shu Kong and Charless C Fowlkes. Recurrent pixel embedding for instance grouping. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9018–9028, 2018.

[13] Neeraj Kumar, Ruchika Verma, Sanuj Sharma, Surabhi Bhargava, Abhishek Vahadane, and Amit Sethi. A dataset and a technique for generalized nuclear segmentation for computational pathology. *IEEE transactions on medical imaging*, 36(7):1550–1560, 2017.

[14] Ta-Chih Lee, Rangasami L. Kashyap, and Chong-Nam Chu. Building skeleton models via 3-d medial surface/axis thinning algorithms. *Computer Vision, Graphics, and Image Processing*, 56(6):462–478, 1994.

[15] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[16] Vebjorn Ljosa, Katherine L Sokolnicki, and Anne E Carpenter. Annotated high-throughput microscopy image sets for validation. *Nature methods*, 9(7):637–637, 2012.

[17] Zhi Lu, Gustavo Carneiro, and Andrew P Bradley. An improved joint optimization of multiple level set functions for the segmentation of overlapping cervical cells. *IEEE Transactions on Image Processing*, 24(4):1261–1272, 2015.

[18] Zhi Lu, Gustavo Carneiro, Andrew P Bradley, Daniela Ushizima, Masoud S Nosrati, Andrea GC Bianchi, Claudia M Carneiro, and Ghassan Hamarneh. Evaluation of three algorithms for the segmentation of overlapping cervical cells. *IEEE journal of biomedical and health informatics*, 21(2):441–450, 2016.

[19] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 fourth international conference on 3D vision (3DV)*, pages 565–571. IEEE, 2016.

[20] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[21] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[22] Uwe Schmidt, Martin Weigert, Coleman Broaddus, and Gene Myers. Cell detection with star-convex polygons. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 265–273. Springer, 2018.

[23] Yuli Wu, Long Chen, and Dorit Merhof. Improving pixel embedding learning through intermediate distance regression supervision for instance segmentation. In *ECCV Workshops*, pages 213–227, 2020.

[24] Weimiao Yu, Hwee Kuan Lee, Srivats Hariharan, Wen Yu Bu, and Sohail Ahmed. Ccdb: 6843, mus musculus, neuroblastoma. *Cell Image Library.* URL https://doi.org/doi:10.7295/W9CCDB6843.