Superpixel-based Refinement for Object Proposal Generation

Christian Wilms
Computer Vision Group
University of Hamburg
Hamburg, Germany
Email: wilms@informatik.uni-hamburg.de

Simone Frintrop
Computer Vision Group
University of Hamburg
Hamburg, Germany
Email: frintrop@informatik.uni-hamburg.de

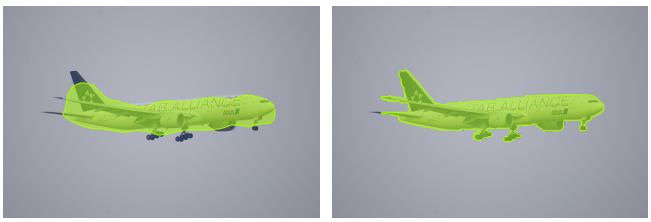
***Abstract*—Precise segmentation of objects is an important problem in tasks like class-agnostic object proposal generation or instance segmentation. Deep learning-based systems usually generate segmentations of objects based on coarse feature maps, due to the inherent downsampling in CNNs. This leads to segmentation boundaries not adhering well to the object boundaries in the image. To tackle this problem, we introduce a new superpixel-based refinement approach[1] on top of the state-of-the-art object proposal system AttentionMask. The refinement utilizes superpixel pooling for feature extraction and a novel superpixel classifier to determine if a high precision superpixel belongs to an object or not. Our experiments show an improvement of up to 26.0% in terms of average recall compared to original AttentionMask. Furthermore, qualitative and quantitative analyses of the segmentations reveal significant improvements in terms of boundary adherence for the proposed refinement compared to various deep learning-based state-of-the-art object proposal generation systems.**

I. Introduction

Object proposal generation, the class-agnostic generation of object candidates in images, is a vital part of many modern object detection or instance segmentation systems to reduce the search space [1]–[4]. Depending on the task, object candidates are either bounding boxes [5], [6] or pixel-precise segmentation masks [7]–[10]. In contrast to instance segmentation, object proposal generation is class-agnostic and thus not limited to the classes seen in training. This leads to a more general approach for discovering objects in images [11].

Since the advent of deep learning, object proposal generation made big strides [5]–[10]. However, deep learning systems easily miss fine details of objects. This is due to the nature of convolutional neural networks (CNNs) subsampling the input to generate semantically rich features. While usually not relevant in classification, the loss of fine details is very important for tasks like pixel-precise object proposal generation. Here, detailed contours like the airplane tail or the wheels in Fig. 1 are usually lost. This is especially a problem for larger objects as those suffer most from downsampling effects [9], [10].

To tackle this problem, multiple approaches were presented in semantic segmentation or salient object detection. Applying a conditional random field (CRF) as post-processing [12]–[14] is a simple solution. Similarly, an encoder-decoder architecture can upsample the coarse results utilizing guidance from earlier feature maps [13]–[16]. Those are well-established techniques in salient object detection [12], [15] and semantic segmentation [13], [14], [16]. However, object proposal generation is different as individual segmentations are created for the object proposals, which potentially overlap. A CRF would have to be applied to every proposal and an encoder-decoder architecture is only applicable by taking crops of the input image. Thus, both options are computationally demanding [8], [17].



(a) without our refinement (b) with our refinement

Fig. 1. Example of an object proposal without (a) and with (b) refined contours after applying our superpixel-based refinement. Note the precise segmentation of fine details after applying our refinement.

Other lines of work include dilated convolutions to generate semantically rich features without downsampling [16], [18]. [19], [20] propose iterative adaptation of initial segmentations. These techniques are mostly computationally demanding as well and thus not applicable to hundreds of object proposals generated by object proposal generation systems. Hence, a refinement system is necessary that effectively captures details of objects and shares computation between proposals to be efficient enough for application to hundreds of object proposals.

In classic computer vision, superpixels [21] are used to reduce the number of basic entities in images. Subsequently, this increases the efficiency, while retaining the relevant structures of images. Superpixels as a result of an oversegmentation represent coherent regions of an image. Due to the non-lattice structure of superpixel segmentations, it is difficult to integrate superpixels naturally into CNNs [22]. However, [23]–[25] already successfully applied superpixels in CNNs to aggregate information across parts of feature maps.

In this paper, we therefore propose an end-to-end learned superpixel-based refinement approach for object proposal gen-

[1]Code is available at: https://www.inf.uni-hamburg.de/spxrefinement

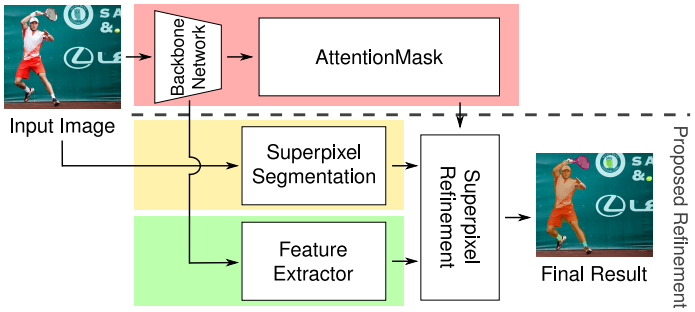


Fig. 2. Overview of the proposed superpixel-based refinement on top of AttentionMask [10]. Superpixel segmentations are generated and features are extracted from the backbone network. The superpixel refinement creates feature vectors per superpixels based on the segmentations and a feature extractor. The per-superpixel feature vectors are combined with the coarse results of AttentionMask and classified for final, more precise object proposals.

eration systems. As visualized in Fig. 2, we utilize the coarse proposal masks of the state-of-the-art object proposal generation system AttentionMask [10] and the highly accurate superpixels [26]. To combine both, we first apply superpixel pooling. Inspired by [23]–[25], we extract per-superpixel features from a backbone network that are shared between proposals. Second, we sample superpixels in and around the coarse object proposal masks for refinement. Finally, we propose a new superpixel classifier to distinguish between superpixels as part of an object or the background. The classifier decides based on the pooled features and the coarse object proposal masks roughly capturing objects. This leads to highly precise results utilizing deep features. Quantitative results on the LVIS dataset [27], which has more precise annotations compared to the MS COCO dataset [28], indicate the effectiveness of our superpixel-based refinement. Additionally, qualitative and quantitative results show a significantly improved boundary adherence compared to state-of-the-art systems.

Our main contributions are:

- Utilizing superpixel pooling in a shared and end-to-end learned manner to extract rich features while maintaining highly accurate segmentations capturing fine details.
- A superpixel classifier to distinguish foreground and background superpixels based on pooled features and a prior derived from coarse object proposals as guidance.
- Introducing a superpixel-based refinement approach that can be easily integrated into object proposal generation systems to increase the results by up to 26.0%.

II. Related Work

In this section, we give an overview of related work in the areas of object proposal generation and superpixels in CNNs.

1) Object Proposal Generation: Object proposal generation as a new task was first introduced by [29]. [30] give a survey of systems not using deep learning. In general, systems can be distinguished by the type of proposals they generate. The proposals are either bounding boxes [5], [6], [31], [32] or pixel-precise segmentation masks [7]–[10], [34]–[36]. Systems generating bounding boxes sample windows across the input image and calculate an objectness score per window. The score describes the likelihood of the window containing an object. Additionally, the bounding box is refined from the initial window. Both, hand-crafted [31], [32] and deep learning [5], [6] systems exist. In contrast to those, we are working on pixel-precise segmentation masks. This is more complex as not only the four parameters of the bounding box have to be regressed but a classification of each pixel or superpixel is necessary.

Pixel-precise object proposal generation systems mainly started from grouping superpixels using hand-crafted features [35], [36]. With the emergence of deep learning, superpixels got abandoned from the systems as their integration into the networks is non-trivial. Thus, most systems switched to a window scoring approach [7]–[10]. DeepMask [7] and SharpMask [8] process windows from an image pyramid and generate a pixel-precise segmentation as well as an objectness score per window. SharpMask in contrast to DeepMask uses an encoder-decoder architecture to refine the initial segmentation.

FastMask [9] reduces the redundancy by generating a feature pyramid within the network, thus extracting features only once per image. From the feature pyramid, windows are densely sampled and objectness scores as well as pixel-precise segmentations are generated. Further increasing the efficiency and improving the detection of small objects, AttentionMask [10] introduces a notion of attention. In AttentionMask, only promising windows are sampled from the feature pyramid to reduce computation and allow an additional pyramid level for small objects. AttentionMask is the base for our proposed superpixel-based refinement approach and is described in more detail in Sec. III. All four approaches, however, suffer from the downsampling in CNNs leading to coarse segmentations.

2) Superpixels in CNNs: Few approaches utilize superpixels for CNNs as the concepts are non-trivial to combine. This is mainly attributed to the missing lattice structure in superpixel segmentations. [22] circumvent the missing lattice structure and create a vector per superpixel describing the differences to all other superpixels in the image. Those vectors serve as input to a CNN. [37] propose a Gaussian bilateral filtering on superpixels to enforce proximate superpixels to have similar output. In salient object detection, [38] and [39] generate the saliency prediction per superpixel based on its surrounding. However, different to our approach, they use superpixels only to reduce the number of entities, not for pooling features.

More similar to us, [23]–[25], [40] utilize superpixel pooling to generate feature representations per superpixel. Using superpixel pooling, [23] aggregate information of similar superpixels along a sequence of frames, while [24] capture semantic units in weakly supervised semantic segmentation. [25] and [40] use features from superpixels for classification in semantic segmentation. However, our approach is different, since we use superpixel pooling to generate refined, more detailed segmentations in object proposal generation.

III. Base Framework

As mentioned in Sec. I, AttentionMask [10] is the base for our refinement approach and therefore briefly described

here. Similar to FastMask [9], in AttentionMask the backbone network processes an image only once yielding a feature pyramid within the network. AttentionMask uses a ResNet-50 [41] as backbone. The feature pyramid consists of 5 or 8 scales ($\mathcal{S}_n$), as depicted in the upper part of Fig. 3, color-coded in red. Here, n denotes the downscale factor with respect to the input image and ranges from 8 to 128. These scales correspond to different object sizes with $\mathcal{S}_8$ capturing small objects while $\mathcal{S}_{128}$ captures large objects. Per scale, features are extracted and a scale-specific objectness attention module (SOAM) is introduced. The SOAMs are learned components that highlight for each scale the locations of relevant objects within the respective feature map. Consequentially, false positives and the computational effort are reduced.

AttentionMask samples windows of fixed size from the feature pyramid at locations where the scale-specific objectness attention of the corresponding SOAM is high. By sampling fixed sized windows across feature maps of different scales, objects of different sizes are detected. For each sampled window the objectness is determined and the attentional head generates a first rough estimation of the object's location. The localization is refined to a coarse pixel-precise segmentation mask of size 40×40. Hence, the size of the mask is independent of the object size in the input image. This process is visualized in the upper right corner of Fig. 3. Note, that we refer here to the AttentionMask$^8_{128}$ model in [10]. However, all subsequent steps can be carried out in combination with the other models from [10] as well.

IV. Method

In this section, we introduce our novel superpixel-based refinement approach on top of AttentionMask (see Sec. III), visualized in Fig. 3. We start from the final proposals of AttentionMask, which are coarse segmentation masks in 40×40 windows and apply the refinement per window, i.e. per proposal. First, the scale $\mathcal{S}_n$ from which each window originates is determined and the according segmentation is chosen. Next, we crop the part of the superpixel segmentation spatially corresponding to the window. The cropped segmentation and the AttentionMask result are the input for our superpixel pooling module, described in Sec. IV-A. The superpixel pooling module calculates the average value of the coarse segmentation mask across every superpixel within the crop, coined mask prior. Furthermore, a feature extractor generates per-scale feature maps from the backbone network. Our superpixel pooling module uses these feature maps and creates a feature vector for every superpixel of each scale's segmentation. To combine these two streams per window, we propose a superpixel sampling module. The superpixel sampling module concatenates for each superpixel of a window the mask prior with the relevant superpixel features. Finally, we propose a novel superpixel classifier (see Sec. IV-B) to classify each superpixel based on this information as part of an object or background. The final highly accurate pixel-precise proposals are created using the superpixel segmentations and the classification results. Sec. IV-C outlines details of this integration into AttentionMask.

A. Superpixel Pooling and Sampling

The first major novelty of the proposed refinement approach is our superpixel pooling module. It enables us to have an output resolution that is largely independent of AttentionMask's final segmentation resolution. Inspired by [23]–[25], our superpixel pooling is a mixture of global average pooling and standard $n \times n$ average pooling known from CNNs. In superpixel pooling, the superpixels describe the areas for pooling an input feature map. Thus, we pool one feature vector per superpixel in a given segmentation. Similar to the standard pooling, backpropagation through the superpixel pooling module is possible. For this purpose, the gradients are propagated back to the locations of the input feature map corresponding to the superpixels.

As indicated in Fig. 3, we utilize our superpixel pooling module twice in our superpixel-based refinement. First, the module generates per-superpixel features based on feature maps extracted from the backbone net. This is applied once on each scale with a unique segmentation and feature map per scale as visualized in the bottom part of Fig. 3, color-coded in green. Second, visualized in the right part of Fig. 3, we apply superpixel pooling to each of the AttentionMask result windows using corresponding crops of the superpixel segmentations. Thus, the windows that are coarsely segmented on a 40×40 grid by AttentionMask are cropped from the segmentations of the respective scales. The central part of Fig. 3 depicts this process, color coded in yellow. Note that the crops of the segmentations are larger (here: 80×80 to 1280×1280) to generate more precise segmentations. The result of the second superpixel pooling step is a superpixelized version of the AttentionMask results denoted as mask prior. This prior helps to capture entire objects as [24] demonstrate.

Subsequently, our novel superpixel sampling module combines the two streams of superpixel pooling. Per window of the AttentionMask results, the superpixel sampling module concatenates the mask prior and the corresponding superpixel features for every superpixel within this window. The lower right part of Fig. 3 visualizes this concatenation of the mask prior (gray superpixels) and pooled superpixel features (bluish superpixels). Finally, we concatenate the per window results of the sampling across all windows to form one batch of superpixels for classification. Note, that the superpixel pooling of features has to be applied only once per scale and superpixel. However, the superpixel pooling of the mask prior has to be applied to each result of AttentionMask individually.

B. Superpixel Classifier

As another major novelty, we propose a superpixel classifier to distinguish between superpixels belonging to objects and background. The classifier, located in the bottom right section of Fig. 3, takes the result of the superpixel sampling module, the concatenation of the mask prior and the pooled feature vector per superpixel, as input. The architecture consists of

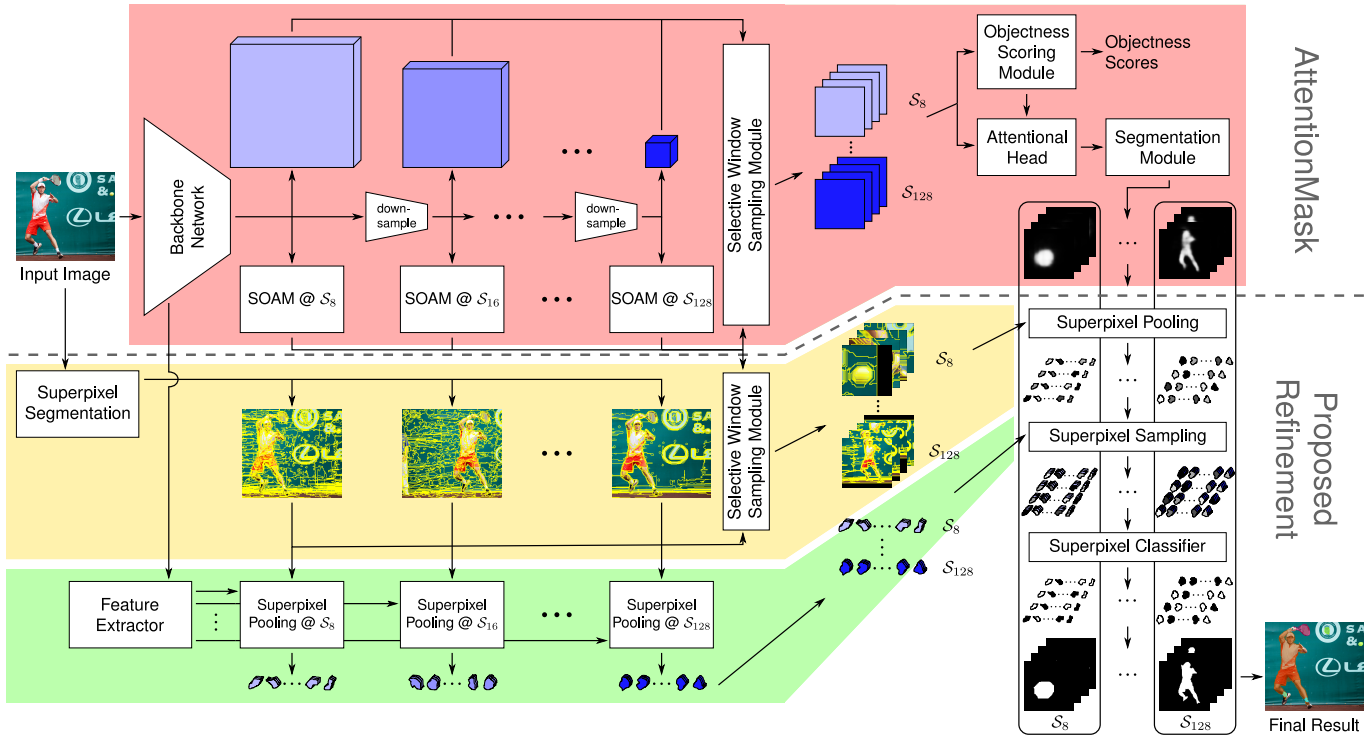


Fig. 3. Detailed view of our proposed superpixel-based refinement approach on top of AttentionMask [10]. We first apply AttentionMask to the input image, generating coarse pixel-precise masks on 40×40 grids (red). In parallel, a highly accurate superpixel segmentation is generated per scale, which captures fine details of objects (yellow). Based on the segmentations, we apply superpixel pooling to feature maps extracted from the backbone network (green). This leads to per-superpixel feature-vectors. Next, we crop the respective parts of the superpixel segmentations for each AttentionMask result window and apply superpixel pooling on the AttentionMask results. The superpixel sampling combines coarse results and superpixels by concatenating the per-superpixel average of AttentionMask results (gray superpixels) and the per-superpixel features (bluish superpixels). Finally, all superpixels are classified and recombined for high precision results. For clarity in this figure only four windows per scale are considered and the feature extractor generates only three features per superpixel.

TABLE I
ARCHITECTURES FOR THE PROPOSED SUPERPIXEL CLASSIFIER. THE FIRST COLUMN INDICATES THE AMOUNT OF NEURONS PER FULLY-CONNECTED LAYER. AR@n DENOTES THE AVERAGE RECALL FOR THE FIRST n PROPOSALS.

Architecture	AR@10	AR@100	AR@1000
512 − 512 − 512	**0.160**	**0.296**	**0.382**
1024 − 1024 − 1024	0.156	0.291	0.381
256 − 256 − 256	0.158	0.292	0.380
512 − 512 − 512 − 512	0.156	0.288	0.377
512 − 512	0.154	0.283	0.373

TABLE II
RESULTS OF THE PROPOSED REFINEMENT APPROACH USING FEATURES FROM DIFFERENT RESNET-BLOCKS IN FEATURE EXTRACTION. AR@n DENOTES THE AVERAGE RECALL FOR THE FIRST n PROPOSALS.

ResNet-block	AR@10	AR@100	AR@1000
res1	0.158	0.292	0.377
res2	**0.160**	**0.296**	**0.382**
res3	0.156	0.292	0.380
res4	0.153	0.287	0.372

three fully-connected layers with 512 neurons and ReLU activation each as well as a final fully-connected layer with sigmoid activation (see Fig. 4). Compared to other structures the proposed architecture is effective, as the results in Tab. I reveal. The classifier processes the superpixels individually without interaction between superpixels. All superpixels of a scale that do not overlap with a window and therefore are neither sampled nor classified are background for that window.

C. Integration into AttentionMask

To integrate our novel superpixel-based refinement into AttentionMask, we first substitute the backbone network from [10] for a ResNet-34 [41]. This is necessary as the original model utilizes almost all the GPU memory and thus does not allow for any refinement steps. Changing the backbone leads to only a small degradation as the results in Sec. VI-A indicate. Following [20], we utilize the output of the *res2*-block to extract features from the backbone. Results using features from other levels of the backbone (see Tab. II) justify this decision. The rest of AttentionMask stays unchanged with all added components learned end-to-end as detailed in Sec.V.

Outside the network we generate superpixel segmentations of the input image with the approach of [26] (FH). FH generates superior results compared to other segmentation approaches (see Sec. VI-B1). The segmentations differ in the number of superpixels between scales. As FH does not allow to explicitly set a number of superpixels, we optimize the free

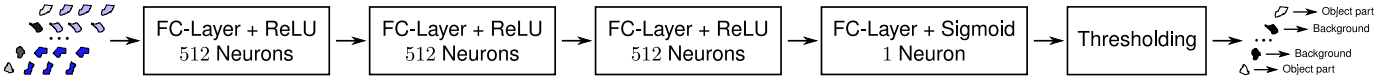


Fig. 4. Architecture of our proposed superpixel classifier with fully-connected layers denoted as FC-layer. The input is a batch of superpixels across different windows and scales (variations of blue). For each superpixel, it consists of one mask prior value (gray) from the AttentionMask result and 512 learned features (bluish) extracted from the *res2*-block of the backbone network. As a result the classifier determines if a superpixel is part of an object (white) or part of the background (black).

parameter in FH. For best performance, we generate roughly 8000 superpixels for $\mathcal{S}_8$ and 500 superpixels for $\mathcal{S}_{128}$ with regular intermediate steps for the intermediate scales.

After superpixel pooling, sampling and classification, we recombine the results per window to form high precision segmentation masks that capture even fine object details. These masks are further post-processed in three steps. First, bilateral filtering [42] is applied on the level of superpixels using color information. Second, we apply morphological opening and closing. Closing removes small holes within superpixels. Opening eliminates segmentation artifacts like thin protuberances extending from the superpixel boundaries, which are frequently occurring in FH segmentations. Finally, we apply non-maximum suppression to the set of proposals with a high intersection over union of 0.95. This leads to the removal of near duplicates, which are more likely to occur when using superpixels as the number of basic entities is smaller.

V. Implementation Details

This section presents technical details of our overall object proposal generation system. In terms of trainable modules, AttentionMask is mainly extended by the novel superpixel classifier (Sec. IV-B). Other components like superpixel pooling or sampling do not learn features as they only restructure the input. Another component, which learns features, is the feature extractor. The feature extractor comprises a 1×1 convolution on the feature maps extracted from the backbone.

To train the superpixel classifier and in turn the feature extractor as well, we use cross entropy loss as the loss function. We integrate this new superpixel classification loss $\mathcal{L}_{spx}$ into the overall loss function $\mathcal{L}$ of AttentionMask by adding it with a weight factor w_{spx} (here $w_{spx} = 1$):

$$\begin{aligned}\mathcal{L} =& w_{objn}\mathcal{L}_{objn} + w_{ah}\mathcal{L}_{ah} + w_{seg}\mathcal{L}_{seg} + \\ & w_{att}\sum_{n}\mathcal{L}_{att_n} + w_{spx}\mathcal{L}_{spx}.\end{aligned} \tag{1}$$

$\mathcal{L}_{objn}, \mathcal{L}_{ah}, \mathcal{L}_{seg}$, and $\mathcal{L}_{att_n}$ denote the losses from [10] for the objectness score, the attentional head, the pixel-precise segmentation and the SOAMs of the different scales $\mathcal{S}_n$ with weights respectively. Therefore, the entire system can be trained end-to-end in one step. For training, similar to [7]–[10], we use the training set of the MS COCO dataset [28]. The backbone network is initialized with ImageNet weights [41] while the rest of the system is learned from scratch. We use SGD as the optimizer and an initial learning rate of 0.0001.

Selecting the ground truth for the superpixel classifier is non-trivial since the annotations of the MS COCO dataset [28] used for training are pixel-precise. Therefore, some superpixels cover both object and non-object pixels. To solve this, we generate a set of superpixels leading to an optimal intersection over union (IoU) for every annotated object. For this purpose, we start with a superpixel completely contained in the annotation. If this does not hold true for any superpixel, we start from the superpixel with the highest IoU with the annotated object. Subsequently, we greedily add all superpixels that increase the IoU between the annotated object and the set of superpixels. The result is an optimal set of superpixels per annotation.

VI. Experiments

In this section, we present qualitative and quantitative results of our novel superpixel-based refinement approach. In contrast to [7]–[10], we carry out the evaluation on the very recent LVIS dataset [27]. The LVIS dataset contains images from the MS COCO dataset's test split [28] with more accurate annotations. More accurate annotations are necessary for evaluating our system, as it aims to recover fine details of objects. An evaluation on the MS COCO dataset itself is not conducted. However, [10] already showed the superior performance of AttentionMask compared to other state-of-the-art object proposal systems on the MS COCO dataset.

Following [8], [30], the results are reported in terms of average recall (AR) for 10, 100 and 1000 proposals as well as for small, medium and large objects. AR describes how many annotated objects are found and how well those objects are segmented. Note that all results and annotations are pixel-precise masks. We compare our system to original AttentionMask [10] based on two different ResNets as backbone as well as to DeepMask [7], SharpMask [8], and FastMask [9]. We also compare to MCG [34], a top-performing non deep learning-based system [30]. Since not relying on deep learning, MCG does not suffer from downsampling effects. Further comparison to instance segmentation systems is not conducted as those systems use class information for segmentation [3].

A. Results on the LVIS dataset

Tab. III shows the quantitative results on the LVIS dataset. The numbers reveal that our proposed system outperforms all other systems. Compared to AttentionMask using the same backbone network the AR@100 improves by 0.021 with improvements across all scales of objects as AR^S@100, AR^M@100 and AR^L@100 indicate. Compared to AttentionMask with the more powerful ResNet-50 backbone and FastMask, the improvements in AR@100 are 0.017 (AttentionMask) and 0.045 (FastMask). Hence, our system is good enough to overcome the less powerful backbone. DeepMask

TABLE III
RESULTS ON THE LVIS DATASET USING AVERAGE RECALL (AR). AR^S, AR^M AND AR^L DENOTE RESULTS ON SMALL, MEDIUM AND LARGE OBJECTS.

Method	Backbone	AR@10	AR@100	AR@1000	AR^S@100	AR^M@100	AR^L@100
MCG [34]	-	0.048	0.131	0.237	0.031	0.204	0.462
DeepMask [7]	ResNet-50	0.069	0.147	0.214	0.014	0.314	0.430
SharpMask [8]	ResNet-50	0.073	0.154	0.229	0.014	0.327	0.460
FastMask [9]	ResNet-50	0.069	0.161	0.256	0.055	0.296	0.386
AttentionMask [10]	ResNet-50	0.073	0.189	0.284	0.081	0.312	0.446
AttentionMask [10]	ResNet-34	0.076	0.185	0.271	0.083	0.305	0.423
Ours	ResNet-34	**0.092**	**0.206**	**0.290**	**0.092**	**0.340**	**0.473**

Fig. 5. Qualitative results of FastMask [9], AttentionMask [10] and our approach on the LVIS dataset. The filled colored contours denote found objects, while not filled red contours denote missed objects. The red arrows highlight locations showing the strength of our proposals adhering well to object boundaries even in complex scenarios.

and SharpMask both show very competitive results on large objects as indicated by the AR^L@100 results. This is mainly attributed to fewer downsampling steps in these systems that depend on an image pyramid. However, our proposed system still outperforms DeepMask and SharpMask on large objects and shows significant improvement in AR@100 across all scales of objects ($+0.059$ and $+0.052$). Interestingly, the non deep learning-based method MCG also produces very competitive results on AR^L@100 as no downsampling is applied. However, the results on medium and small object are much worse compared to our approach.

Qualitative results for the top-performing systems from Tab. III are shown in Fig. 5. It is clearly visible that across the different scenarios the results of our proposed system adhere very well to the object boundaries. For instance in the example in the first row, our result precisely captures fine details like the tail of the sheep, the visible front leg and the ears. In contrast, AttentionMask and FastMask generate rather blob like proposals containing a significant amount of background. The second row shows examples for very small objects with complex boundaries, which are still precisely segmented by our system. The results in the third and fourth row depict

TABLE IV
COMPARISON OF THE BEST PROPOSALS PER IMAGE AS SEGMENTATIONS USING SEGMENTATION MEASURES BOUNDARY RECALL (BR) AND UNDERSEGMENTATION ERROR (UE) ON THE LVIS DATASET.

Method	Backbone	BR↑	UE↓
MCG [34]	-	0.685	0.073
DeepMask [7]	ResNet-50	0.488	0.087
SharpMask [8]	ResNet-50	0.561	0.080
FastMask [9]	ResNet-50	0.510	0.084
AttentionMask [10]	ResNet-50	0.568	0.070
AttentionMask [10]	ResNet-34	0.547	0.075
Ours	ResNet-34	0.681	0.068

examples of humans in complex scenarios with spread legs and arms. Our proposals fit much better to the arms and legs capturing mostly the entire limbs. Note that the ground truth for the image in the forth row annotates the wet suits rather than the humans itself. The final row depicts an image with a highly cluttered environment where our system is still able to refine the proposals well especially around the corners of the objects. Despite the strong overall performance, this example also shows a minor failure case for our system as the overlapping proposals for the nearby monitors indicate. Other typical sources of errors are usually related to the segmentation and occur in areas with low contrast or high texture.

Further analyzing the accuracy of the segmentations, we use the typical segmentation metrics boundary recall (BR) and undersegmentation error (UE). To generate segmentations from annotations, we join per image the binary ground truth segmentations. For the results we join the binary segmentations of the best proposal per annotated object for each system. Thus, we create one proposal segmentation per image and system as well as one ground truth segmentation per image. The results on this segmentation-based evaluation are presented in Tab. IV. Our proposed approach outperforms all deep learning based systems in terms of BR and UE. The significant improvement in terms of BR (between 39.5% and 19.9%) indicates the stronger boundary adherence of our results. Therefore, our system segments more object boundaries significantly more precise. The results are also in line with the qualitative results in Fig. 5. Compared to the non deep learning-based approach MCG, our results are on a similar level in terms of BR and better in terms of UE. Thus, we are able to combine the highly accurate segmentations of the non deep learning systems with the strong overall performance of the deep learning systems.

B. Ablation Studies

In addition to the ablation studies in Sec. IV-B and Sec. IV-C, we show the influence of the number of superpixels, the segmentation method and the post-processing steps (Sec. IV-C) to further analyze our proposed refinement. Except for the post-processing, all ablation studies were conducted on a validation set of MS COCO dataset with a reduced model following [9], [10]. The reduced model contains only five scales and no post-processing to speed up training and testing.

TABLE V
RESULTS OF DIFFERENT SEGMENTATION METHODS AND NUMBERS OF SUPERPIXELS USED IN THE REFINEMENT APPROACH. NUMBERS OF SUPERPIXELS ARE APPROXIMATED ACROSS THE VALIDATION DATASET.

Segmentation	#Superpixels	AR@10	AR@100	AR@1000
FH [26]	$8000-500$	0.160	0.296	0.382
FH [26]	$4000-250$	0.151	0.282	0.370
FH [26] with GT	$8000-500$	0.182	0.345	0.462
FH [26] with GT	$4000-250$	0.198	0.385	0.527
ETPS [44]	$8000-500$	0.157	0.294	0.377
ERS [45]	$8000-500$	0.143	0.278	0.368
SEEDS [46]	$8000-500$	0.148	0.269	0.344
SLIC [43]	$8000-500$	0.137	0.267	0.356

TABLE VI
RESULTS OF THE PROPOSED REFINEMENT APPROACH USING DIFFERENT POST-PROCESSING STEPS ON THE LVIS DATASET.

Post-processing	AR@10	AR@100	AR@1000
none	0.076	0.184	0.276
bilateral filtering	0.079	0.191	0.288
+ opening	0.079	0.192	0.292
+ closing	0.079	0.193	0.293
+ near duplicate removal	0.092	0.206	0.290

1) Influence of Segmentation: First, we evaluate the number of superpixels per scale with the used segmentation method FH [26]. Compared to the setup presented in Sec. IV-C using approximately 8000 to 500 superpixels per scale, a reduction by the factor two does not improve the results (see Tab. V). However, if we add ground truth edges to the segmentations a coarser segmentation is beneficial. This indicates that a strong oversegmentation is not helpful. Further increasing the number of superpixels is impossible due to memory constrains.

Similar conclusions can be drawn from the results of different segmentation methods. We compare the FH segmentation to the popular SLIC [43] method as well as ETPS [44], ERS [45] and SEEDS [46]: the top 3 methods in the superpixel benchmark of [47]. All four methods show inferior results compared to FH (see Tab. V). This correlates with a higher oversegmentation error (up to $+17.6\%$) that those methods exhibit as they partition the image in rather regular superpixels of similar size. Thus, even highly uniform objects are artificially split into many pieces that have to be merged later on.

2) Post-processing of Superpixel Results: We further analyze the influence of the three step post-processing, described in Sec. IV-C. Tab. VI shows the results without post-processing and step-wise added post-processing on the LVIS dataset. Overall, the results in terms of AR are increased by 5.1% for AR@1000 and up to 21.1% for AR@10 with a positive effect of each post-processing step. Most important are the bilateral filtering and the duplicate removal. Especially the duplicate removal shows a strong increase in terms of AR@10 ($+16.5\%$) compared to only applying the other two post-processing steps. This indicates a rather large number of duplicates or near duplicates in the original results due to the use of superpixels.

VII. Conclusion

In this paper we address the problem of imprecise object segmentations due to downsampling in CNN-based object proposal generation systems. To tackle the problem, we presented a superpixel-based refinement approach on top of the state-of-the-art class-agnostic object proposal generation system AttentionMask. Our refinement utilizes highly accurate superpixel segmentations, superpixel pooling and a novel superpixel classifier to enhance the initial results. The experiments revealed improvements of up to 26.0% for our overall system on the LVIS dataset against original AttentionMask with further improvements compared to other state-of-the-art systems. We also showed the superiority of our proposals in terms of segmentation measures boundary recall (max. $+39.5\%$) and undersegmentation error (max. $+23.0\%$) compared to state-of-the-art systems. In combination with the qualitative results, this indicates the strength of our proposals adhering to object boundaries. Ablation studies on the segmentation methods and the number of superpixels reveal the need for a highly accurate segmentation with low oversegmentation error, thus not artificially dividing uniform image areas.

References

[1] R. Girshick, "Fast R-CNN," in *ICCV*, 2015, pp. 1440–1448.

[2] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *NIPS*, 2015, pp. 91–99.

[3] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *ICCV*, 2017, pp. 2961–2969.

[4] L.-C. Chen, A. Hermans, G. Papandreou, F. Schroff, P. Wang, and H. Adam, "MaskLab: Instance segmentation by refining object detection with semantic and direction features," in *CVPR*, 2018, pp. 4013–4022.

[5] S. Gidaris and N. Komodakis, "Attend refine repeat: Active box proposal generation via in-out localization," in *BMVC*, 2016, pp. 90.1–90.13.

[6] H. Li, Y. Liu, W. Ouyang, and X. Wang, "Zoom out-and-in network with recursive training for object proposal," *arXiv preprint arXiv:1702.05711*, 2017.

[7] P. O. Pinheiro, R. Collobert, and P. Dollár, "Learning to segment object candidates," in *NIPS*, 2015, pp. 1990–1998.

[8] P. O. Pinheiro, T.-Y. Lin, R. Collobert, and P. Dollár, "Learning to refine object segments," in *ECCV*, 2016, pp. 75–91.

[9] H. Hu, S. Lan, Y. Jiang, Z. Cao, and F. Sha, "FastMask: Segment multi-scale object candidates in one shot," in *CVPR*, 2017, pp. 991–999.

[10] C. Wilms and S. Frintrop, "AttentionMask: Attentive, efficient object proposal generation focusing on small objects," in *ACCV*, 2018, pp. 678–694.

[11] A. Ošep, P. Voigtlaender, M. Weber, J. Luiten, and B. Leibe, "4D generic video object proposals," in *ICRA*, 2020.

[12] Q. Hou, M.-M. Cheng, X. Hu, A. Borji, Z. Tu, and P. H. Torr, "Deeply supervised salient object detection with short connections," in *CVPR*, 2017, pp. 3203–3212.

[13] G. Lin, A. Milan, C. Shen, and I. Reid, "RefineNet: Multi-path refinement networks for high-resolution semantic segmentation," in *CVPR*, 2017, pp. 1925–1934.

[14] R. Li, K. Li, Y.-C. Kuo, M. Shu, X. Qi, X. Shen, and J. Jia, "Referring image segmentation via recurrent refinement networks," in *CVPR*, 2018, pp. 5745–5753.

[15] G. Li and Y. Yu, "Deep contrast learning for salient object detection," in *CVPR*, 2016, pp. 478–487.

[16] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *TPAMI*, vol. 40, no. 4, pp. 834–848, 2017.

[17] S. Chen, X. Tan, B. Wang, and X. Hu, "Reverse attention for salient object detection," in *ECCV*, 2018, pp. 234–250.

[18] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *ICLR*, 2016.

[19] C. Zhang, G. Lin, F. Liu, R. Yao, and C. Shen, "CANet: Class-agnostic segmentation networks with iterative refinement and attentive few-shot learning," in *CVPR*, 2019, pp. 5217–5226.

[20] A. Kirillov, Y. Wu, K. He, and R. Girshick, "PointRend: Image segmentation as rendering," 2020, pp. 9799–9808.

[21] X. Ren and J. Malik, "Learning a classification model for segmentation," in *ICCV*, 2003, pp. 10–17.

[22] S. He, R. W. Lau, W. Liu, Z. Huang, and Q. Yang, "SuperCNN: A superpixelwise convolutional neural network for salient object detection," *IJCV*, vol. 115, no. 3, pp. 330–344, 2015.

[23] Y. He, W.-C. Chiu, M. Keuper, and M. Fritz, "STD2P: RGBD semantic segmentation using spatio-temporal data-driven pooling," in *CVPR*, 2017, pp. 4837–4846.

[24] S. Kwak, S. Hong, and B. Han, "Weakly supervised semantic segmentation using superpixel pooling network," in *AAAI*, 2017, pp. 4111–4117.

[25] H. Park, J. Jeong, Y. J. Yoo, and N. Kwak, "Superpixel-based semantic segmentation trained by statistical process control," in *BMVC*, 2017, pp. 78.1–78.13.

[26] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *IJCV*, vol. 59, no. 2, pp. 167–181, 2004.

[27] A. Gupta, P. Dollar, and R. Girshick, "LVIS: A dataset for large vocabulary instance segmentation," in *CVPR*, 2019, pp. 5356–5364.

[28] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, and P. Dollár, "Microsoft COCO: Common objects in context," in *ECCV*, 2014, pp. 740–755.

[29] B. Alexe, T. Deselaers, and V. Ferrari, "What is an object?" in *CVPR*, 2010, pp. 73–80.

[30] J. Hosang, R. Benenson, P. Dollár, and B. Schiele, "What makes for effective detection proposals?" *TPAMI*, vol. 38, no. 4, pp. 814–830, 2015.

[31] B. Alexe, T. Deselaers, and V. Ferrari, "Measuring the Objectness of Image Windows," *TPAMI*, vol. 34, no. 11, pp. 2189–2202, 2012.

[32] L. Zitnick and P. Dollar, "Edge boxes: Locating object proposals from edges," in *ECCV*, 2014, pp. 391–405.

[33] W. Kuo, B. Hariharan, and J. Malik, "DeepBox: Learning objectness with convolutional networks," in *ICCV*, 2015, pp. 2479–2487.

[34] J. Pont-Tuset, P. Arbelaez, J. T. Barron, F. Marques, and J. Malik, "Multiscale combinatorial grouping for image segmentation and object proposal generation," *TPAMI*, vol. 39, no. 1, pp. 128–140, 2017.

[35] S. Manén, M. Guillaumin, and L. Van Gool, "Prime Object Proposals with Randomized Prim's Algorithm," in *ICCV*, 2013, pp. 2536–2543.

[36] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *IJCV*, vol. 104, no. 2, pp. 154–171, 2013.

[37] R. Gadde, V. Jampani, M. Kiefel, D. Kappler, and P. V. Gehler, "Superpixel convolutional networks using bilateral inceptions," in *ECCV*, 2016, pp. 597–613.

[38] R. Zhao, W. Ouyang, H. Li, and X. Wang, "Saliency detection by multi-context deep learning," in *CVPR*, 2015, pp. 1265–1274.

[39] Y. Tang and X. Wu, "Saliency detection via combining region-level and pixel-level predictions with cnns," in *ECCV*. Springer, 2016, pp. 809–825.

[40] M. Mostajabi, P. Yadollahpour, and G. Shakhnarovich, "Feedforward semantic segmentation with zoom-out features," in *CVPR*, 2015, pp. 3376–3385.

[41] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016, pp. 770–778.

[42] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *ICCV*, 1998, pp. 839–846.

[43] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *TPAMI*, vol. 34, no. 11, pp. 2274–2282, 2012.

[44] J. Yao, M. Boben, S. Fidler, and R. Urtasun, "Real-time coarse-to-fine topologically preserving segmentation," in *CVPR*, 2015, pp. 2947–2955.

[45] M.-Y. Liu, O. Tuzel, S. Ramalingam, and R. Chellappa, "Entropy rate superpixel segmentation," in *CVPR*, 2011, pp. 2097–2104.

[46] M. Van den Bergh, X. Boix, G. Roig, B. de Capitani, and L. Van Gool, "SEEDS: Superpixels extracted via energy-driven sampling," in *ECCV*, 2012, pp. 13–26.

[47] D. Stutz, A. Hermans, and B. Leibe, "Superpixels: An evaluation of the state-of-the-art," *Computer Vision and Image Understanding*, vol. 100, no. 166, pp. 1–27, 2018.