Vision Transformers Are Good Mask Auto-Labelers

Shiyi Lan\textsuperscript{1} Xitong Yang\textsuperscript{2} Zhiding Yu\textsuperscript{1} Zuxuan Wu\textsuperscript{3} Jose M. Alvarez\textsuperscript{1} Anima Anandkumar\textsuperscript{1,4}  
\textsuperscript{1}NVIDIA \textsuperscript{2}Meta AI, FAIR \textsuperscript{3}Fudan University \textsuperscript{4}Caltech

\url{https://github.com/NVlabs/mask-auto-labeler}

Figure 1. Examples of mask pseudo-labels generated by Mask Auto-Labeler on COCO. Only human-annotated bounding boxes are used as supervision during training to obtain these results.

Abstract

We propose Mask Auto-Labeler (MAL), a high-quality Transformer-based mask auto-labeling framework for instance segmentation using only box annotations. MAL takes box-cropped images as inputs and conditionally generates their mask pseudo-labels. We show that Vision Transformers are good mask auto-labelers. Our method significantly reduces the gap between auto-labeling and human annotation regarding mask quality. Instance segmentation models trained using the MAL-generated masks can nearly match the performance of their fully-supervised counterparts, retaining up to 97.4\% performance of fully supervised models. The best model achieves 44.1\% mAP on COCO instance segmentation (test-dev 2017), outperforming state-of-the-art box-supervised methods by significant margins. Qualitative results indicate that masks produced by MAL are, in some cases, even better than human annotations.

1. Introduction

Computer vision has seen significant progress over the last decade. Tasks such as instance segmentation have made it possible to localize and segment objects with pixel-level accuracy. However, these tasks rely heavily on expansive human mask annotations. For instance, when creating the COCO dataset, about 55k worker hours were spent on masks, which takes about 79\% of the total annotation time [1]. Moreover, humans also make mistakes. Human annotations are often misaligned with actual object boundaries. On complicated objects, human annotation quality tends to drop significantly if there is no quality control. Due to the expensive cost and difficulty of quality control, some other large-scale detection datasets such as Open Images [2] and Objects365 [3], only contain partial or even no instance segmentation labels.

In light of these limitations, there is an increasing interest in pursuing box-supervised instance segmentation, where the goal is to predict object masks from bounding box supervision directly. Recent box-supervised instance segmentation methods [4–8] have shown promising performance. The emergence of these methods challenges the long-held belief that mask annotations are needed to train instance segmentation models. However, there is still a non-negligible gap between state-of-the-art approaches and their fully-supervised oracles.

Our contributions: To address box-supervised instance segmentation, we introduce a two-phase framework consisting of a mask auto-labeling phase and an instance segmentation training phase (see Fig. 2). We propose a Transformer-based mask auto-labeling framework, Mask Auto-Labeler (MAL), that takes Region-of-interest (RoI) images as inputs.
Figure 2. An overview of the two-phase framework of box-supervised instance segmentation. For the first phase, we train Mask Auto-Labeler using box supervision and conditionally generate masks of the cropped regions in training images (top). We then train the instance segmentation models using the generated masks (bottom).

and conditionally generates high-quality masks (demonstrated in Fig. 1) within the box. Our contributions can be summarized as follows:

- Our two-phase framework presents a versatile design compatible with any instance segmentation architecture. Unlike existing methods, our framework is simple and agnostic to instance segmentation module designs.
- We show that Vision Transformers (ViTs) used as image encoders yield surprisingly strong auto-labeling results. We also demonstrate that some specific designs in MAL, such as our attention-based decoder, multiple-instance learning with box expansion, and class-agnostic training, crucial for strong auto-labeling performance. Thanks to these components, MAL sometimes even surpasses humans in annotation quality.
- Using MAL-generated masks for training, instance segmentation models achieve up to 97.4% of their fully supervised performance on COCO and LVIS. Our result significantly narrows down the gap between box-supervised and fully supervised approaches. We also demonstrate the outstanding open-vocabulary generalization of MAL by labeling novel categories not seen during training.

Our method outperforms all the existing state-of-the-art box-supervised instance segmentation methods by large margins. This might be attributed to good representations of ViTs and their emerging properties such as meaningful grouping [9], where we observe that the attention to objects might benefit our task significantly (demonstrated in Fig. 6). We also hypothesize that our class-agnostic training design enables MAL to focus on learning general grouping instead of focusing on category information. Our strong results pave the way to remove the need for expensive human annotation for instance segmentation in real-world settings.

2. Related work

2.1. Vision Transformers

Transformers were initially proposed in natural language processing [10]. Vision Transformers [11] (ViTs) later emerged as highly competitive visual recognition models that use multi-head self-attention (MHSA) instead of convolutions as the basic building block. These models are recently marked by their competitive performance in many visual recognition tasks [12]. We broadly categorize existing ViTs into two classes: plain ViTs, and hierarchical ViTs.

**Standard Vision Transformers.** Standard ViTs [11] are the first vision transformers. Standard ViTs have the simplest structures, which consist of a tokenization embedding followed by a sequence of MHSA layers. However, global MHSA layers can be heavy and usually face significant optimization issues. To improve their performance, many designs and training recipes are proposed to train ViTs in data-efficient manners [9, 13–19].

**Hierarchical Vision Transformers.** Hierarchical Vision Transformers [12, 20–22] are pyramid-shaped architectures that aim to benefit other tasks besides image classification with their multi-scale designs. On top of plain ViTs, these ViTs [20, 21] separate their multi-head self-attention layers into hierarchical stages. Between the stages, there are spatial reduction layers, such as max-pooling layers. These architectures are usually mixed with convolutional layers [23] and often adopt efficient self-attention designs to deal with long sequence lengths.

2.2. Instance segmentation

Instance segmentation is a visual recognition task that predicts the bounding boxes and masks of objects.

**Fully supervised instance segmentation.** In this setting, both bounding boxes and instance-level masks are provided as the supervision signals. Early works [24–27] follow a two-stage architecture that generates box proposals or segmentation proposals in the first stage and then produces the final segmentation and classification information in the second stage. Later, instance segmentation models are broadly divided into two categories: some continue the spirit of the two-stage design and extend it to multi-stage architectures [28, 29]. Others simplify the architecture and propose one-stage instance segmentation, e.g., YOLACT [30], SOLO [31, 32], CondInst [33], PolarMask [34, 35]. Recently, DETR and Deformable DETR [36, 37] show great potential of query-based approaches in object detection. Then, methods like MaxDeepLab [38], MaskFormer [39], PanopticSegFormer [40], Mask2Former [41] and Mask DINO [42] are introduced along this line and have pushed the boundary of instance segmentation. On the other hand, the instance segmentation also benefits from more power-
Weakly supervised instance segmentation. There are two main styles of weakly supervised instance segmentation: learning with image-level and box-level labels. The former uses image-level class information to perform instance segmentation [45–49], while the latter uses box-supervision. Hsu et al. [4] leverages the tight-box priors. Later, BoxInst [5] proposes to leverage color smoothness to improve accuracy. Besides that, DiscoBox [7] proposes to leverage both color smoothness and inter-image correspondence for the task. Other follow-ups [6,8] also leverage tight-box priors and color smoothness priors.

2.3. Deep learning interpretation

The interest in a deeper understanding of deep networks has inspired many works to study the interpretation of deep neural networks. For example, Class Activation Map (CAM) [50] and Grad-CAM [51] visualize the emerging localization during image classification training of convolutional neural networks (CNNs). This ability has also inspired much weakly-supervised localization and shows deep connections to general weakly-supervised learning, which partly motivates our decoder design in this paper. DINO [9] further shows that meaning visual grouping emerges during self-supervised learning with ViTs. In addition, FAN [52] shows that such emerging properties in ViTs are linked to their robustness.

3. Method

Our work differs from previous box-supervised instance segmentation frameworks [4–8] that simultaneously learns detection and instance segmentation. We leverage a two-phase framework as visualized in Fig. 2, which allows us to have a network focused on generating mask pseudo-labels in phase 1, and another network focused on learning instance segmentation [24, 28, 41, 43] in phase 2. Our proposed auto-labeling framework is used in phase 1 to generate high-quality mask pseudo-labels.

We propose this two-phase framework because it brings the following benefits:

- We can relax the learning constraints in phase 1 and focus only on mask pseudo-labels. Therefore, in this phase, we can take Region-of-interest (RoI) images instead of untrimmed images as inputs. This change allows us to use a higher resolution for small objects and a strong training technique mentioned in Sec. 3.1, which helps improve the mask quality.
- We can leverage different image encoders and mask decoders in phases 1 and 2 to achieve higher performance. We empirically found that phases 1 and 2 favor different architectures for the image encoders and mask decoders. See the ablation study in Tab. 3 and 4.
- We can use MAL-generated masks to directly train the most fully supervised instance segmentation models in phase 2. This makes our approach more flexible than previous architecture-specific box-supervised instance segmentation approaches [4–8].

As phase 2 follows the previous standard pipelines, which do not need to be re-introduced here, we focus on introducing phase 1 (MAL) in the following subsections.

3.1. RoI input generation

Most box-supervised instance segmentation approaches [4–7] are trained using the entire images.
However, we find that using RoI images might have more benefits in box-supervised instance segmentation. Moreover, we compare two intuitive sampling strategies of RoI images to obtain foreground and background pixels and explain the better strategy, box expansion, in detail.

**Benefits of using RoI inputs.** There are two advantages of using RoI images for inputs. First, using the RoI images as inputs is naturally good for handling small objects because no matter how small the objects are, the RoI images are enlarged to avoid the issues caused by low resolution. Secondly, having RoI inputs allows MAL to focus on learning segmentation and avoid being distracted from learning other complicated tasks, e.g., object detection.

**RoI sampling strategy.** The sampling strategy should ensure both positive and negative pixels are included. We present two straightforward sampling strategies:

- The first strategy is to use bounding boxes to crop the images for positive inputs. We crop the images using randomly generated boxes containing only background pixels for negative inputs. MAL does not generate good mask pseudo-labels with cropping strategy. We observe that the networks tend to learn the trivial solution (all pixels are predicted as either foreground or background).

- The second is to expand the bounding boxes randomly and include background pixels, where negative bags are chosen from the expanded rows and columns. We visualize how we define positive/negative bags in Fig. 3 and explain the detail in Sec. 3.3. **This detailed design is critical to make MAL work** as it prevents MAL from learning trivial solutions. Without this design, the generated masks tend to fill the entire bounding box.

**Box expansion specifics.** Given an untrimmed image \( I^a \in \mathbb{R}^{C \times H^a \times W^a} \) and the bounding box \( b = (x_0, y_0, x_1, y_1) \) indicating the x, y coordinates of the top-left corners and the bottom-right corners. To obtain background pixels, we randomly expand the bounding box \( b \) to \( b' = (x_c + \beta_x (x_0 - x_c), y_c + \beta_y (y_0 - y_c), x_c + \beta_y (x_1 - x_c), y_c + \beta_y (y_1 - y_c)) \), where \( x_c = (x_0 + x_1)/2, y_c = (y_0 + y_1)/2 \). To generate random values of \( \beta_x, \beta_y, \beta_{x'}, \beta_{y'} \), we randomly generate \( \theta_x, \theta_y \in [0, \theta] \) for x- and y-direction, where \( \theta \) is the upper bound of box expansion rate. Next, we randomly generate \( \beta_x \in [0, \theta_x] \) and \( \beta_y \in [0, \theta_y] \). In the end, we assign \( \beta_{x'} \) as \( \theta_x - \beta_x \) and \( \beta_{y'} \) as \( \theta_y - \beta_y \). Finally, we use \( b' \) to crop the image and obtain trimmed image \( I' \). We conduct the ablation study for \( \theta \) in Tab. 5. At last, we resize the trimmed image \( I' \) to the size of \( C \times H^c \times W^c \) as the input image \( I^c \).

### 3.2. MAL architecture

MAL can be divided into two symmetric networks: the task network and the teacher network. The task network consists of an image encoder denoted as \( E \), and a mask decoder denoted as \( D \), demonstrated in Fig. 3. The architecture of the teacher network is identical to the task network. We denote the segmentation output of the task network and the teacher network as \( m, m^t \in \{0, 1\}^N \), respectively.

**Image encoder.** We use Standard ViTs [11] as the image encoder and drop the classification head of Standard ViTs. We compare different image encoders in Sec. 4.4. We also try feature pyramid networks on top of Standard ViTs, e.g., FPN [53], but it causes a performance drop. Similar conclusions were also found in ViTDet [43].

**Mask decoder.** For the mask decoder \( D \), we use a simple attention-based network inspired by YOLACT [30], which includes an instance-aware head \( K \) and a pixel-wise head \( V \), where \( D(E(I)) = K(E(I)) \cdot V(E(I)) \), and “·” represents the inner-product operator.

For the instance-aware head \( K \), we use a max-pooling layer followed by a fully connected layer. The input channel dimension of \( K \) is equivalent to the output channel dimension of \( E \). The output channel dimension of \( K \) is 256. For the pixel-wise head \( V \), we use four sequential convolutional layers. Each is followed by a ReLU layer. Between the second and the third convolutional layer, we insert a bilinear interpolation layer to increase the feature resolution by 2. The input channel dimension is equivalent to the output channel dimension of \( E \). We use 256 dimensions for hidden channels and output channels. We also compare different design choices of mask decoders in Sec. 4.5.

**Exponential moving average (EMA) teacher.** Instead of training the teacher network directly, we leverage exponential moving averages (EMA) to update the parameters in the teacher network using the parameters in the task network similar to MOCO [54]. The goal of using EMA Teacher is to eliminate the loss-explosion issues in training since optimizing Standard Vision Transformers is usually non-trivial [13, 14, 16]. We do not observe any significant performance drop or improvement on DeiT-small-based MAL after removing the teacher network. However, it makes the training more stable when we use larger-scale image encoders in MAL, e.g. ViT-MAE-Base [13].

### 3.3. Losses

We use Multiple Instance Learning Loss \( L_{mil} \) and Conditional Random Field Loss \( L_{crf} \) as the box-supervised loss:

\[
L = \alpha_{mil} L_{mil} + \alpha_{crf} L_{crf}
\]  

(1)
Multiple Instance Learning Loss. The motivation of the Multiple Instance Segmentation is to exploit the priors of tight-bounding box annotations.

After the student network produces the output $m$, we apply the Multiple Instance Learning (MIL) Loss on the output mask $m$. We demonstrate the process in Fig. 3.

We denote $m_{i,j}$ as the mask score at the location $i,j$ in the image $I^c$. We define each pixel as an instance in the MIL loss. Inspired by BBTP [4], we treat each row or column of pixels as a bag. We determine whether a bag is positive or negative based on whether it passes a ground-truth box. We define the bags as $B$, and each bag $B_i$ contains a row or column of pixels. Additionally, we define the label for each bag $g$, and each label $g_i$ corresponds to a bag $B_i$.

Therefore, we use the max pooling as the reduction function and dice loss [55]:

$$
L_{mil} = 1 - \frac{2 \sum_i g_i \cdot \max\{B_i\}^2}{\sum_i \max\{B_i\}^2 + \sum_i g_i}
$$

(2)

Conditional Random Field Loss. The goal of CRF loss is to refine the mask prediction by imposing the smoothness priors via energy minimization. Then, we leverage this refined mask as pseudo-labels to self-train the mask prediction in an online-teacher manner. We use the average mask prediction $m^a = \frac{1}{2}(m + m^i)$ as the mask prediction to be refined for more stable training.

Next, we define a random field $X = \{X_1, \ldots, X_N\}$, where $N = H^c \times W^c$ is the size of cropped image and each $X_i$ represents the label that corresponds to a pixel in $I^c$, therefore we have $X \in \{0,1\}^N$, meaning the background or the foreground. We use $l \in \{0,1\}^N$ to represent a labeling of $X$ minimizing the following CRF energy:

$$
E(l|m^a, X) = \mu(X|m^a, I^c) + \psi(X|I^c),
$$

(3)

where $\mu(X|m^a, I^c)$ represents the unary potentials, which is used to align $X_i$ and $m^a_i$ since we assume that most of the mask predictions are correct. Meanwhile, $\psi(X|I^c)$ represents the pairwise potential, which sharpens the refined mask. Specifically, we define the pairwise potentials as:

$$
\psi(X|I^c) = \sum_{i \in N(i)} \omega \exp\left(-\frac{|I^c_i - I^c_j|^2}{2\zeta^2}\right)[X_i \neq X_j],
$$

(4)

where $N(i)$ represents the set of 8 immediate neighbors to $X_i$, as shown in Fig. 3. Then, we use the MeanField algorithm [7, 56] to efficiently approximate the optimal solution, denoted as $l = \text{MeanField}(I^c, m^a)$. We attach the derivation and PyTorch code in the supplementary. At last, we apply Dice Loss to leverage the refined masks $l$ to self-train the models as:

$$
L_{self} = 1 - \frac{2 \sum_i l_i m_i}{\sum_i l_i^2 + m_i}
$$

(5)

4. Experiments

We evaluate MAL on COCO dataset [1], and LVIS [57]. The main results on COCO and LVIS are shown in Tab. 1 and 2. The qualitative results are shown in Fig. 1 and Fig. 5.

4.1. Datasets

COCO dataset. contains 80 semantic categories. We follow the standard partition, which includes train2017 (115K images), val2017 (5K images), and test-dev (20k images).

LVIS dataset. contains 1200+ categories and 164K images. We follow the standard partition of training and validation.

4.2. Implementation Details

We use 8 NVIDIA Tesla V100s to run the experiments.

Phase 1 (mask auto-labeling). We use AdamW [58] as the network optimizer and set the two momentums as 0.9, 0.9. We use the cosine and annealing scheduler to adjust the learning rate, which is set to $1.5 \cdot 10^{-4}$ per image. The MIL loss weight $\alpha_{mil}$, CRF loss weight $\alpha_{crf}$, $\zeta$, and $\omega$ in CRF pairwise potentials are set to 0.4, 0.5, 0.5, 2, respectively. We analyze the sensitivity of the loss weights and CRF hyperparameters in Fig. 8. We use the input resolution of 512 × 512, and a batch size of 32 (4 per GPU). For EMA, we use a momentum of 0.996. For the task and teacher network, we apply random flip data augmentation. On top of that, we apply extra random color jittering, random grey-scale conversion, and random Gaussian blur for the task network. We train MAL for 10 epochs. It takes around 23 hours and 35 hours to train MAL with Standard ViT-Base [11] on the COCO and LVIS datasets, respectively.

Phase 2 (Training instance segmentation models). We select a couple of high-performance fully supervised instance segmentation models, which are ConvNeXts [44] with Cascade R-CNN [28], Swin Transformers [12] with Mask2Former [41], ResNets [59] and ResNeXts [60] with SOLOv2 [31]. MAL works extremely well with these architectures, which demonstrates the great power of Mask Auto-Labeler from the perspective of accuracy and generalization. We leverage the codebase in MMDetection [61] for phase 2. Again, we only replace the GT masks with MAL-generated mask pseudo-labels to adjust all these fully supervised models to box-supervised learning.

4.3. Instance segmentation results

Retention Rate. We argue that the sole mAP of instance segmentation is not fair enough to evaluate box-supervised instance segmentation since the performance gain can be achieved by improving box quality unrelated to segmentation quality. However, the retention rate can better reflect the real mask quality because the fully supervised counterparts also get boosted by the better box results.

Results on COCO. In table 1, we show that various modern instance segmentation models can achieve up to 94.5%
Table 1. Main results on COCO. Ret means the retention rate of box-supervised mask AP. MAL with SOLOv2/ResNeXt-101 outperforms DiscoBox with SOLOv2/ResNeXt-101 by 1.6% on val2017 and 1.3% on test-dev. Our best model (Mask2former/Swin-Small) achieves 43.3% AP on val and 44.1% AP on test-dev.

4.4. Image encoder variation

To support our claim that Vision Transformers are good auto-labelers, we compare three popular networks as the image encoders of MAL: Standard Vision Transformers [11, 13, 16], Swin Transformer [12], ConvNeXts [44] in Tab. 4. First, we compare the fully supervised pretrained weights of these three models. We choose the official fully supervised pre-trained weights of ConvNeXts and Swin Transformers. For Standard Vision Transformers, we adopt a popular fully supervised approach, DeiT [16]. We observe that fully supervised Standard Vision Transformers (DeiT) as image encoders of Mask Auto-Labeler are better than Swin Transformers and ConvNeXts even though the imagenet-1k performance of Swin Transformers and ConvNeXts is higher than that of DeiT. We argue that the success of Standard Vision Transformers might be owed to the self-emerging properties of Standard ViTs [9, 11] (visualized in Fig. 6), and the larger-receptive field brought by global multi-head self-attention layers.

4.5. Mask decoder variation

We compare four different modern designs of mask decoders: the fully connected Decoder [62], the fully convolutional decoder [24,63], the attention-based decoder [30,31], and the query-based decoder [41] in Tab. 3. For the fully connected Decoder, we use two fully connected layers with a hidden dimension of 2048 and then output a confidence map for each pixel. We reshape this output vector as the 2D confidence map. For the fully convolutional Decoder, we adopt the pixel-wise head $V$ in the attention-based Decoder. For the query-based decoder, we follow the design in Mask2Former [41]. We spend much effort exploring the query-based Decoder on MAL since it performs extremely well on fully supervised instance segmentation. However, the results are surprisingly unsatisfactory. We suspect the slightly heavier layers might cause optimization issues un-
Table 2. Main results on LVIS v1. Training data means the dataset we use for training MAL. We also finetune it on COCO and then generate pseudo-labels of LVIS v1. Compared with trained on LVIS v1 directly, MAL finetuned on COCO only caused around 0.35% mAP drop on the final results, which indicates the great potential of the open-set ability of MAL. Ret means the retention rate of box-supervised mask AP.

Table 3. Ablation study of box expansion. We use Standard ViT-MAE-Base as the image encoder of MAL in phase 1 and Cascade RCNN with ConvNext-Small as the instance segmentation models in phase 2. The numbers are reported in % Mask mAP. Among different designs, the attention-based decoder performs the best. We can not obtain reasonable results with Query-based Decoder.

Table 4. Ablation study of different backbones. All models are pre-trained on ImageNet-1k. ConvNeXt and Swin Transformer outperform DeiT on image classification, but standard ViT-DeiT-Small [16] (ViT-DeiT-Small) outperforms ConvNeXt-base and Swin-Base on mask Auto-labeling. Standard ViT-Base (ViT-MAE-Base) and Standard ViT-Large (ViT-MAE-Large) pretrained via MAE achieve the best performance on mask Auto-labeling.

4.6. Clustering analysis

As the results are shown in Tab. 4, we wonder why the Standard ViTs outperform other modern image encoders in auto-labeling. As the comparison of classification ability does not seem to reflect the actual ability of auto-labeling, we try to use the ability clustering to evaluate the image encoders because foreground(FG)/background(BG) segmentation is very similar to the binary clustering problem.

Specifically, we extract the feature map output by the last layers of Swin Transformers [12], ConvNeXts [44], Standard ViTs [11]. Then, we use the GT mask to divide the feature vectors into the FG and BG feature sets. By evaluating the average distance from the FG/BG feature vectors to their clustering centers, we can reveal the ability of the networks to distinguish FG and BG pixels empirically.

Formally, we define the feature vector of token $i$ generated by backbone $E$ as $f^E_i$. We define the FG/BG clustering centers $f^F_1$, $f^B_0$ as the mean of the FG/BG feature vectors. Then, we use the following metric as the clustering score:

$$S = \frac{1}{N} \sum_{i}^{N} \left( \frac{f^E_i}{|f^E_i|} - \frac{f^F_1}{|f^F_1|} \right)^2,$$

where if pixel $i$ is FG, $\gamma(i) = 1$, otherwise $\gamma(i) = 0$.

We show the clustering evaluation on the COCO val 2017 in Tab. 6. The results align our conclusion that Standard Vision Transformers are better at mask auto-labeling.
Figure 6. Attention visualization of two RoI images produced by MAL. In each image group, the left-most image is the original image. We visualize the attention map output by the 4th, 8th, 12th MHSA layers of the Standard ViTs in MAL.

Figure 7. The lateral comparison between MAL-generated pseudo-labels (top) and GT masks (bottom) on COCO val2017. On the left, we observe that MAL-generated pseudo-labels are sharper and more boundary-sticky than GT masks in some cases. On the right, we observe that in highly occluded situations, human-annotated masks are still better.

Figure 8. Sensitivity analysis of loss weights and CRF hyper-parameters. We use ViT-Base [11] pretrained via MAE [13] as the image encoder for the first phase and SOLOv2 (ResNet-50) for the second phase. The x-axis and y-axis indicate the hyper-parameter values and the (%)mask AP, respectively.

4.7. MAL masks vs. GT masks

We show the apple-to-apple qualitative comparison in Fig. 7. We find out two prominent patterns in this qualitative comparison. First, MAL-generated mask pseudo-labels are sharper and boundary-sticky than the human-annotated masks since many mask annotations in COCO are poorly annotated. Second, occlusion is also a challenging problem in MAL-generated mask pseudo-labels.

5. Conclusions

In this work, we propose a novel two-phase framework for box-supervised instance segmentation and a novel Transformer-based architecture, Mask Auto-Labeler (MAL), to generate high-quality mask pseudo-labels in phase 1. We reveal that Standard Vision Transformers are good mask auto-labelers. Moreover, we find that random using box-expansion RoI inputs, the attention-based Decoder, and class-agnostic training are crucial to the strong mask auto-labeling performance. Moreover, thanks to the two-phase framework design and MAL, we can adjust almost all kinds of fully supervised instance segmentation models to box-supervised learning with little performance drop, which shows the great generalization of MAL.

Limitations. Although great improvement has been made by our approaches in mask auto-labeling, we still observe many failure cases in the occlusion situation, where human annotations are much better than MAL-generated masks. Additionally, we meet saturation problems when scaling the model from Standard ViT-Base to Standard ViT-Large. We leave those problems in the future work.

Broader impacts. Our proposed Transformer-based mask auto-labeler and the two-phase architecture serve as a standard paradigm for high-quality box-supervised instance segmentation. If follow-up work can find and fix the issues under our proposed paradigm, there is great potential that expansive human-annotated masks are no longer needed for instance segmentation in the future.
References


[13] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 16000–16009, 2022. 2, 4, 6, 7, 8


[22] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al.


