Real-Time Panoptic Segmentation from Dense Detections

Rui Hou[*,1,2] Jie Li[*,1] Arjun Bhargava[1] Allan Raventos[1] Vitor Guizilini[1] Chao Fang[1]
Jerome Lynch[2] Adrien Gaidon[1]

[1]Toyota Research Institute [2]University of Michigan, Ann Arbor
[1]{firstname.lastname}@tri.global [2]{rayhou, jerlynch}@umich.edu

Abstract

Panoptic segmentation is a complex full scene parsing task requiring simultaneous instance and semantic segmentation at high resolution. Current state-of-the-art approaches cannot run in real-time, and simplifying these architectures to improve efficiency severely degrades their accuracy. In this paper, we propose a new single-shot panoptic segmentation network that leverages dense detections and a global self-attention mechanism to operate in real-time with performance approaching the state of the art. We introduce a novel parameter-free mask construction method that substantially reduces computational complexity by efficiently reusing information from the object detection and semantic segmentation sub-tasks. The resulting network has a simple data flow that requires no feature map re-sampling, enabling significant hardware acceleration. Our experiments on the Cityscapes and COCO benchmarks show that our network works at 30 FPS on 1024×2048 *resolution, trading a* 3% *relative performance degradation from the current state of the art for up to* 440% *faster inference.*

1. Introduction

Scene understanding is the basis of many real-life applications, including autonomous driving, robotics, and image editing. Panoptic segmentation, proposed by Kirillov et al. [14], aims to provide a complete 2D description of a scene. This task requires each pixel in an input image to be assigned to a semantic class (as in semantic segmentation) and each object instance to be identified and segmented (as in instance segmentation). Facilitated by the availability of several open-source datasets (e.g. Cityscapes [5], COCO [20], Mapillary Vistas [23]), this topic has drawn a lot of attention since it was first introduced [15, 13, 36, 28].

In panoptic segmentation, pixels are categorized in two high level classes: *stuff* representing amorphous and uncountable regions (such as sky and road), and *things* covering countable objects (such as persons and cars). These two categories naturally split the panoptic segmentation task into two sub-tasks, namely semantic segmentation and instance segmentation. Most recent approaches use a single backbone for feature extraction and add various branches on top of the shared representations to perform each downstream task separately, generating the final panoptic prediction with fusion heuristics [13, 36, 28].

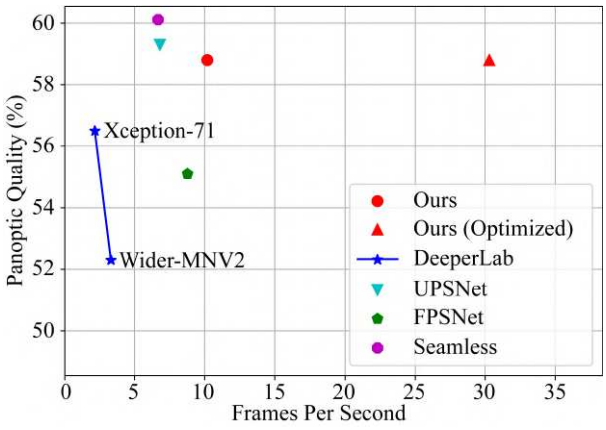

Figure 1: **Inference times and panoptic quality** (PQ) for the state of the art and our method on the Cityscapes validation set at 1024×2048 resolution with a ResNet-50-FPN backbone (except for DeeperLab). Our method runs in real-time at a competitive accuracy.

To date, most studies on panoptic segmentation focus on improving model accuracy, either by integrating more advanced semantic and instance segmentation methods [28, 16] or by introducing novel information flow and loss functions[15, 30]. None of these methods are suitable for real-time applications due to prohibitively slow inference speeds. A small subset of recent works is making progress towards faster panoptic segmentation algorithms [7, 38] but at a significant cost in terms of accuracy.

To achieve *high quality panoptic segmentation under*

[*] Equal contribution.
This work was done when Rui Hou was an intern at Toyota Research Institute (TRI).

real-time constraints, we identify two key opportunities for streamlining existing frameworks. Our first observation is that most accurate instance segmentation methods follow a "detect-then-segment" philosophy, but a significant amount of information is discarded during the "detect" phase. Specifically, "dense" object detection algorithms such as YOLO [29], RetinaNet [19] and FCOS [31] first generate a super-set of bounding box proposals (at least one per location), wherein multiple proposals may correspond to a single target object. Then, Non-Maximum Suppression (NMS) or an equivalent filtering process picks out predictions with the highest confidence and ignores the rest. This selection strategy discards lower ranking proposals generated by the network, even though they might have significant overlap with the ground truth. We instead propose to **reuse dense bounding box proposals discarded by NMS to recover instance masks directly**, i.e. without re-sampling features [36] or clustering post-processing [38].

Second, we observe that semantic segmentation captures much of the same information as detection, especially in existing panoptic segmentation frameworks. For example, in [36, 13, 15, 28], class predictions for object detections are a subset of those for semantic segmentation, and are produced from identical representations. Hence, **sharing computations across semantic segmentation and detection streams can significantly reduce the overall complexity**.

Given these insights, we explore how to maximally reuse information in a single-shot, fully-convolutional panoptic segmentation framework that achieves real-time inference speeds while obtaining performance comparable with the state of the art. Our **main contributions** are threefold: (i) we introduce a novel panoptic segmentation method extending dense object detection and semantic segmentation by reusing discarded object detection outputs via parameter-free global self-attention; (ii) we propose a single-shot framework for real-time panoptic segmentation that achieves comparable performance with the current state of the art as depicted in Figure 1, but with up to 4x faster inference; (iii) we provide a natural extension to our proposed method that works in a weakly supervised scenario.

2. Related Work

2.1. Instance Segmentation

Instance segmentation requires distinct object instances in images to be localized and segmented. Recent works can be categorized into two types: two-stage and single-stage methods. Represented by Mask R-CNN [10] and its variations [22, 4, 12], two-stage algorithms currently claim the state of the art in accuracy. The first stage proposes a set of regions of interest (RoIs) and the second predicts instance masks from features extracted using RoIAlign [10]. This feature re-pooling and re-sampling operation results in large computational costs that significantly decrease efficiency, rendering two-stage models challenging to deploy in real-time systems. Single-stage methods, on the other hand, predict instance location and shape simultaneously. Some single-stage methods follow the detect-then-segment approach, with additional convolutional heads attached to single-stage object detectors to predict mask shapes [37, 32, 3, 35]. Others learn representations for each foreground pixel and perform pixel clustering to assemble instance masks during post-processing [24, 8, 6, 25, 17]. The final representation can be either explicit instance-aware features [24, 17], implicitly learned embeddings [6, 25], or affinity maps with surrounding locations at each pixel [8].

Following the detect-then-segment philosophy, our work tackles instance segmentation solely based on object detection predictions. In this sense, it is similar to works which densely predict location-related information to separate different instances [24, 38, 6]. However, even though these methods claim real-time performance, their location proposal and pixel clustering processes are formulated in an iterative pixel-based manner, which makes inference time dramatically increase with the number of instances. Our framework regresses bounding boxes in such a way that proposals can be selected directly through NMS.

Our framework is also similar to the Proposal-Free Network (PFN) [17], as we use instance bounding box coordinates as features to group same-instance pixels. However, PFN relies on predicting the number of instances and on a clustering algorithm which is sensitive to parameter tuning. We use bounding boxes as both proposals and embeddings, and apply a straightforward correlation matrix-based approach for mask recovery. Casting instance segmentation as an object detection problem dramatically decreases the number of model parameters and hyper-parameters involved in training.

2.2. Panoptic Segmentation

Originally proposed in [14], panoptic segmentation has been widely accepted by the computer vision community as a major task for dense visual scene understanding. Each pixel in an image needs to be assigned a semantic label as well as a unique identifier (ID) if it is an object instance. Pixels with the same label belong to the same category, and pixels with the same ID (if present) furthermore belong to the same object instance.

Current state-of-the-art panoptic segmentation work uses a multi-decoder network to perform prediction with redundant information, as well as more sophisticated instance or semantic segmentation heads for better performance [36, 13, 15, 21, 28, 16].

While obtaining state-of-the-art accuracy on panoptic segmentation, these methods are far from real-time applications mainly due to two reasons: 1) the instance segmen-

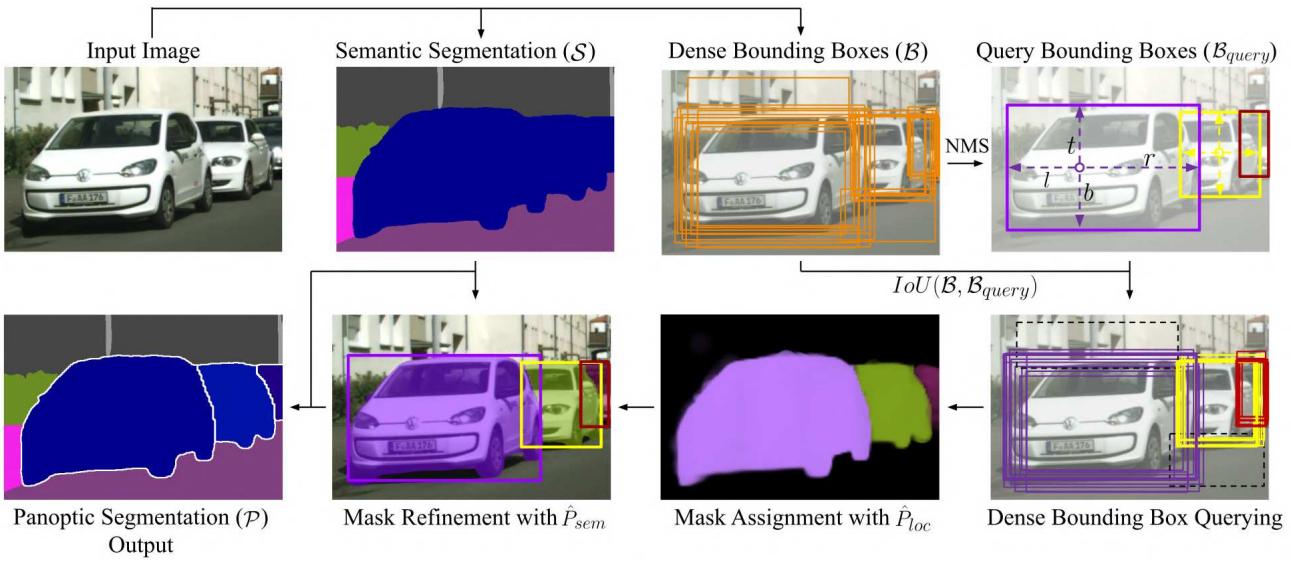

Figure 2: **Panoptic segmentation from dense detections.** We obtain a panoptic segmentation $\mathcal{P}$ from a semantic segmentation $\mathcal{S}$ and dense bounding box predictions $\mathcal{B}$. We first select the highest-confidence "query" bounding boxes through NMS; then we estimate a location based mask probability $\hat{P}_{loc}$ through self-attention of the query boxes and the dense box predictions. Finally, we refine the instance mask with semantic probability maps $\hat{P}_{sem}$ and merge them to produce our panoptic output.

tation branch contains two stages, which makes it difficult to be accelerated by inference engines (e.g. TensorRT [2]); and 2) similar information (i.e., semantics) is processed by both branches, demanding redundant kernel operations that slow down the entire network.

Recently, DeeperLab [38] proposed to solve panoptic segmentation with a one-stage instance parser without using the same design as Mask R-CNN [10]. However, this approach to instance segmentation involves prediction of key points and instance center offsets with a complicated post-processing step, which prevents it from running in real-time despite its single-shot design. Some other methods [7, 33] focus on pushing inference time to real-time by using simpler instance separation methods, however their accuracy is not comparable with the state of the art.

In this work, we resolve the deployment difficulties associated with multi-stage models by grafting a single-stage panoptic head directly onto the backbone, which simultaneously predicts instance as well as semantic segmentation information. Our method is more compatible with inference engine deployment, requiring less memory copy and resampling. Finally, our method also makes efficient usage of all network kernels, as no redundant information is predicted from different sequential branches. As a result, we can achieve real-time inference with accuracy comparable with the current state of the art, thus setting a new baseline for real-time panoptic segmentation methods.

3. Panoptic from Dense Detections

3.1. Problem Formulation

In this section, we describe our approach to address the panoptic segmentation task ($\mathcal{P}$) by solving a semantic segmentation task ($\mathcal{S}$) and a dense bounding box detection task ($\mathcal{B}$) as depicted in Figure 2. The objective of **panoptic segmentation** is to predict semantic and instance IDs for each pixel (x, y) in the input image:

$$\mathcal{P}(x, y) = (c, k), \quad c \in \{1, ..., N\}, k \in \mathbb{N}, \tag{1}$$

where c is the semantic class ID, k is the instance ID (with 0 for all *stuff* classes) and N is the total number of classes, including *stuff* (N_{stuff}) and *things* (N_{things}) classes. In the **semantic segmentation** sub-task, we predict a distribution over semantic classes for each pixel (x, y), $\mathcal{S}(x, y)$. We denote $\hat{P}_{sem}(x, y, c)$ as the predicted probability at pixel (x, y) for semantic class c, given by:

$$\hat{P}_{sem}(x, y, c) = \mathcal{S}(x, y)[c], \quad \mathcal{S}(x, y) \in \mathbb{R}^N \tag{2}$$

In the **dense bounding box detection** sub-task, we predict at least one bounding box at each image pixel:

$$\begin{aligned} \mathcal{B}(x, y) = \mathbf{B}, \quad \mathbf{B} = (\mathbf{b}, c), \\ \mathbf{b} = (x_1, x_2, y_1, y_2) \in \mathbb{R}^4, \quad c \in \{1, ..., N_{\text{things}}\}, \end{aligned} \tag{3}$$

where (x_1, y_1) and (x_2, y_2) are the coordinates of the top-left and bottom-right corners of bounding box $\mathbf{B}$ that pixel (x, y) belongs to; c is the predicted class ID for the corresponding bounding box. We note that because $\mathcal{S}$ and $\mathcal{B}$ are of fixed dimensions, they can be directly learned and predicted by fully convolutional networks.

3.2. Parameter-Free Mask Construction

Given $\mathcal{S}$ and $\mathcal{B}$, we introduce a parameter-free mask reconstruction algorithm to produce instance masks based on a global self-attention mechanism. Note that the following operations are embarrassingly parallel. We first obtain a reduced set of bounding box proposals from $\mathcal{B}$ through NMS:

$$\mathcal{B}_{query} = \{\mathbf{B}_j\}, \quad \mathbf{B}_j = (\mathbf{b}_j, c_j). \tag{4}$$

We denote the proposal set as $\mathcal{B}_{query}$ because we will use them to "search" for instance masks. For each query box $\mathbf{B}_j$, we construct a global mask probability map given by:

$$\mathcal{M}(x, y, j) = \hat{P}_{loc}(x, y, j) \cdot \hat{P}_{sem}(x, y, c_j), \tag{5}$$

where $\hat{P}_{loc}(x, y, j)$ is an estimated probability that pixel (x, y) is inside object j's bounding box. We estimate this probability by self-attention between the global set of boxes $\mathcal{B}$ and the query box $\mathbf{B}_j$ with Intersection over Union (IoU):

$$\hat{P}_{loc}(x, y, j) = \text{IoU}\left(\mathcal{B}(x, y), \mathbf{B}_j\right), \tag{6}$$

where $\text{IoU}(\mathbf{B}_i, \mathbf{B}_j) = \text{intersection}(\mathbf{b}_i, \mathbf{b}_j)/\text{union}(\mathbf{b}_i, \mathbf{b}_j)$, and $\hat{P}_{sem}(x, y, c_j)$ is the predicted probability that pixel (x, y) shares the same semantic class c_j given by Eq. 2.

To construct the final instance masks $\{\mathbf{M}_j\}$, we apply a simple threshold σ to the global mask probability map:

$$\mathbf{M}_j(x, y) = \mathcal{M}(x, y, j) > \sigma. \tag{7}$$

To produce a final panoptic segmentation, we follow the conventional fusion strategy in [14]. A graphical illustration of the complete method is shown in Figure 2. We demonstrate the efficacy of our proposed method in a novel real-time single-stage architecture in the following section. We note, however, that this method can be generalized to any architecture that provides predictions of $\mathcal{B}$ and $\mathcal{S}$.

4. Real-Time Panoptic Segmentation Network

We propose a single-stage panoptic segmentation network capable of real-time inference, as shown in Figure 3. As in other recent works on panoptic segmentation [28, 13], our architecture is built on a ResNet-style [11] encoder with a Feature Pyramid Network (FPN) [18]. Our FPN module consists of 5 levels corresponding to strides of 128, 64, 32, 16 and 8 with respect to the input image.

Our panoptic segmentation architecture is inspired by FCOS [31], a fast and accurate fully convolutional, anchorless object detector. However, we use a finer-grained target assignment scheme and additional single-convolution layers for semantic segmentation and a novel "levelness" prediction, which will be described later in Section 4.1. Our framework also leverages a fully tensorizable mask construction algorithm described in Section 3, and we propose an explicit instance mask loss which further improves the quality of the final panoptic segmentation.

4.1. Target Assignment

We formulate object detection as a per-pixel prediction problem. Specifically, let $\mathbf{F}_i \in \mathbb{R}^{h_i \times w_i \times C}$ be the feature map at layer i of the FPN, with stride z. We assign a target to each location (x, y) on $\mathbf{F}_i$ for all i. Since each (x, y) is the center of a receptive field, we can recover the pixel location in the original image using the relation: $(x_o, y_o) = (\lfloor \frac{z}{2} \rfloor + xz, \lfloor \frac{z}{2} \rfloor + yz)$. If a pixel location (x_o, y_o) falls within one of the ground truth instance masks $\mathcal{M}^t : \{\mathbf{M}_j^t\}$, then we consider it a foreground sample. Since instance masks are non-overlapping, location (x_o, y_o) is associated with *only one* mask $\mathbf{M}_j^t$ and its corresponding bounding box $\mathbf{B}_j^t$. This avoids the "ambiguous pixel" issue described in [31]. If location (x_o, y_o) is associated with ground truth $\mathbf{B}_j^t$, then we assign the following regression offsets $\mathbf{t}_{xy}^t = (l, t, r, b)$ to it:

$$l = x_o - x_1,\ t = y_o - y_1,\ r = x_2 - x_o,\ b = y_2 - y_o, \tag{8}$$

where $\mathbf{b}_j^t = (x_1, y_1, x_2, y_2)$ are the bounding box coordinates as defined in Eq 3. These 4-directional offsets are also showcased in Figure 2. We define $\mathcal{T}_i$ to be the set of regression targets $\{\mathbf{t}_{xy}^t\}$ assigned to feature map F_i.

Since it is possible that locations on multiple FPN levels $\mathbf{F}_i$ resolve to the same (x_o, y_o), we disambiguate them by removing offset $\mathbf{t}_{xy}^t$ from level i if it does not satisfy the following policy:

$$\mathbf{t}_{xy}^t \in \mathcal{T}_i, \text{iff} \quad m_{i-1} <= \max(l, t, r, b) <= m_i, \tag{9}$$

where m_i is the heuristic value of maximum object size for target assignment in $\mathbf{F}_i$. At each location, we also predict centerness, o_{xy}^t [31]:

$$o_{xy}^t = \sqrt{\frac{\min(l, r)}{\max(l, r)} \times \frac{\min(t, b)}{\max(t, b)}}. \tag{10}$$

During NMS, we multiply the bounding box confidence with predicted centerness to down-weight bounding boxes predicted near object boundaries. For each location, we also predict the object class, c_{xy}^t, of the assigned bounding box $\mathbf{B}_j^t$. Thus, we have a 6-dimensional label, $(\mathbf{t}_{xy}^t, c_{xy}^t, o_{xy}^t)$, at each foreground location (x, y) on each $\mathbf{F}_i$.

4.2. Unified Panoptic Head

We design a unified panoptic head predicting both semantic segmentation and dense bounding boxes from the multi-scale features maps.

Per Level Predictions At each location (x, y) of FPN level $\mathbf{F}_i$, we predict dense bounding boxes using two feature towers, one for localization and the other for semantics, as shown in Figure 3. Each tower contains 4 sequential convolutional blocks (Conv + GroupNorm + ReLU). The towers are shared across different FPN levels. We directly predict bounding box offsets $\hat{\mathbf{t}}_{xy}$, centerness $\hat{o}_{xy}$, and bounding

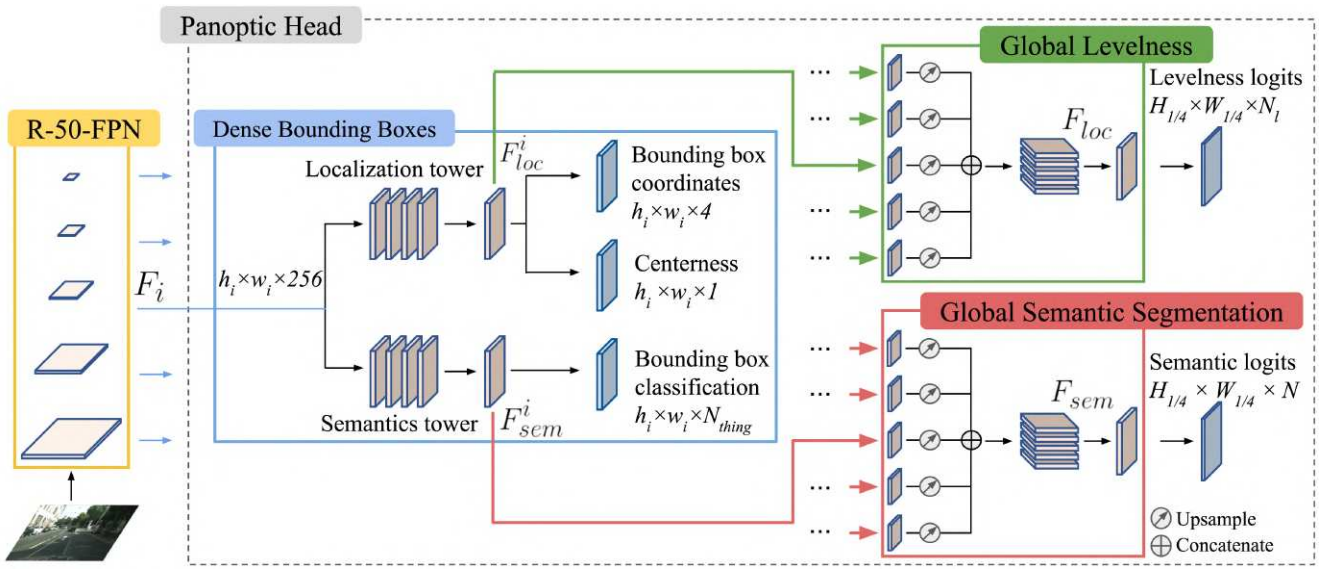

Figure 3: **Real-time panoptic segmentation.** Our model uses ResNet-50-FPN as backbone. The multi-scale feature maps are fed into a *unified* panoptic head. We predict dense bounding boxes at each FPN Level. We also upsample and concatenate intermediate feature maps across all levels to predict global levelness and semantic logits.

box class probabilities $\hat{\mathbf{c}}_{xy}$. The bounding box offset is predicted from the localization tower and the box class probability distribution is predicted from the semantics tower. We adopt an IoU Loss ([39]) on bounding box regression:

$$\mathcal{L}_{box\_reg} = \frac{1}{N_{\mathbf{fg}}} \sum_{xy} \mathrm{L}_{\mathrm{IoU}}(\hat{\mathbf{b}}_{xy}, \mathbf{b}^t_{xy}) \mathbb{1}_{\mathbf{fg}}(x, y), \tag{11}$$

where $N_{\mathbf{fg}}$ is the number of foreground (i.e. things) pixels according to the ground truth mask, $\hat{\mathbf{b}}_{xy}$ and $\mathbf{b}^t_{xy}$ are the absolute bounding box coordinates computed from Eq. 8 using $\hat{\mathbf{t}}_{xy}$ and $\mathbf{t}^t_{xy}$, and $\mathbb{1}_{\mathbf{fg}}(x, y)$ is an indicator function yielding 1 when (x, y) corresponds to a foreground.

We compute a loss on predicted centerness at the same locations (x, y) of each FPN level $\mathbf{F}_i$ using a Binary Cross Entropy (BCE):

$$\mathcal{L}_{center} = \frac{1}{N_{\mathbf{fg}}} \sum_{xy} \mathrm{L}_{\mathrm{BCE}}(\hat{o}_{xy}, o^t_{xy}) \mathbb{1}_{\mathbf{fg}}(x, y). \tag{12}$$

Finally, we predict a probability distribution over object classes $\hat{\mathbf{c}}_{xy} \in \mathbb{R}^{N_{\text{things}}}$ for all feature locations (x, y) including background pixels. For our box classification loss $\mathcal{L}_{box\_cls}$, we use a sigmoid focal loss as in [19, 31], averaged over the total number of locations across all FPN levels.

Global Predictions. In addition to the per level predictions, we leverage the intermediate features from the two towers (F^i_{loc} and F^i_{sem}) to globally predict:

1. Levelness $\mathcal{I}$: the FPN level that the bounding box at each location (x, y) belongs to (N_l= the number of FPN levels +1 logits for each location with 0 reserved for background pixels).
2. Semantic segmentation $\mathcal{S}$: the semantic class probability distribution over N classes.

As depicted in Figure 3, we upsample each $\mathbf{F}^i_{loc}$ and $\mathbf{F}^i_{sem}$ to an intermediate size of $(H/4, W/4)$ and concatenate them into a global $\mathbf{F}_{loc}$ and $\mathbf{F}_{sem}$. The levelness is predicted from $\mathbf{F}_{loc}$ through a single convolutional layer and is supervised by the FPN level assignment policy defined in (9). The levelness is trained using a multi-class cross-entropy loss:

$$\mathcal{L}_{levelness} = \mathrm{L}_{\mathrm{CE}}(\mathcal{I}, \mathcal{I}^t). \tag{13}$$

At inference time, for every (x, y) we have one bounding box prediction $\hat{\mathbf{b}}^{(i)}_{xy}$ coming from each FPN level $\mathbf{F}_i$. Levelness tells us which $\hat{\mathbf{b}}^{(i)}_{xy}$ to include in our global set of dense bounding box predictions $\mathcal{B}$:

$$\mathcal{B}(x, y) = \hat{\mathbf{b}}^{(\operatorname{argmax} \mathcal{I}(x,y))}_{xy}. \tag{14}$$

Instead of using a separate branch for semantic segmentation [15, 36, 13], we reuse the same features as bounding box classification. Doing so dramatically reduces the number of parameters and inference time of the network. We predict the full class semantic logits from F_{sem}, which we supervise using a cross-entropy loss:

$$\mathcal{L}_{semantics} = \mathrm{L}_{\mathrm{CE}}(\mathcal{S}, \mathcal{S}^t), \tag{15}$$

where $\mathcal{S}^t$ denotes semantic labels. We bootstrap this loss to only penalize the worst 30% of predictions as in [34, 27].

4.3. Explicit Mask Loss

As discussed in [31], the quality of bounding box prediction tends to drop with distance from the object center. This hurts the performance of our mask construction near boundaries. In order to refine instance masks, we introduce a loss that aims to reduce False Positive (FP) and False Negative (FN) pixel counts in predicted masks:

$$\mathcal{L}_{mask} = \frac{1}{|\mathcal{B}_{query}|} \sum_{j}^{|\mathcal{B}_{query}|} \frac{\beta_j}{N_j}(E_{\text{FP}_j} + E_{\text{FN}_j}), \quad (16)$$

where β_j is the IoU between proposal $\mathbf{b}_j$ and its associated target box $\mathbf{b}_j^t$, N_j is the count of foreground pixels in the ground truth mask for $\mathbf{b}_j^t$, and E_{FP_j} & E_{FN_j} are proxy measures for the counts of FP and FN pixels in our predicted mask for box $\mathbf{b}_j$:

$$E_{\text{FP}_j} = \sum_{xy} \text{IoU}(\mathcal{B}(x,y), \mathbf{b}_j)\mathbb{1}_{(x,y)\notin \mathbf{M}_j^t}, \quad (17)$$

$$E_{\text{FN}_j} = \sum_{xy} (1 - \text{IoU}(\mathcal{B}(x,y), \mathbf{b}_j))\mathbb{1}_{(x,y)\in \mathbf{M}_j^t}. \quad (18)$$

$\mathbb{1}_{(x,y)\notin \mathbf{M}_j^t}$ and $\mathbb{1}_{(x,y)\in \mathbf{M}_j^t}$ are indicator functions representing whether (x, y) belongs to a ground-truth instance mask. By penalizing FP and FN's, the mask loss helps to improve the final panoptic segmentation result as shown in ablative analysis (Table 3).

Our final loss function is:

$$\begin{aligned}\mathcal{L}_{total} = \mathcal{L}_{box\_reg} + \mathcal{L}_{center} + \mathcal{L}_{levelness} \\ + \mathcal{L}_{box\_cls} + \lambda\mathcal{L}_{semantics} + \mathcal{L}_{mask}.\end{aligned} \quad (19)$$

For Cityscapes experiments, we set λ to 1 for simplicity. For COCO experiments, we drop λ to 0.4 to account for the increased magnitude of the cross-entropy loss that results from the larger ontology.

5. Experiments

In this section, we evaluate our method on standard panoptic segmentation benchmarks. We compare our performance to the state of the art in both accuracy and efficiency. We also provide an extensive ablative analysis.

5.1. Datasets

The **Cityscapes** panoptic segmentation benchmark [5] consists of urban driving scenes with 19 classes, 8 *thing* classes, containing instance level labels, and 11 *stuff* classes. In our experiments, we only use the images annotated with fine-grained labels: 2975 for training, 500 for validation. All the images have a resolution of 1024×2048.

COCO [20] is a large-scale object detection and segmentation dataset. Following the standard protocol, we use the 2017 edition with $118k$ training and $5k$ validation images. The labels consist of 133 classes with 80 *thing* classes containing instance level annotations.

5.2. Metrics

We use the Panoptic Quality (PQ) metric proposed by [14] as summary metric:

$$PQ = \frac{\Sigma_{(p,g)\in TP} \text{IoU}_{(p,g)} \mathbb{1}_{\text{IoU}_{(p,g)}>0.5}}{|TP| + \frac{1}{2}|FP| + \frac{1}{2}|FN|}, \quad (20)$$

where *p* and *g* are matched predicted and ground-truth segments, and TP, FP, FN denote true positives, false positives, and false negatives, respectively. A positive detection is defined by $\text{IoU}_{(p,g)} > 0.5$. We also provide standard metrics on sub-tasks, including Mean IoU for semantic segmentation and average over AP^r [9] for instance segmentation.

5.3. Implementation details

All the models in our experiments are implemented in PyTorch and trained using 8 Tesla V100 GPUs. Inference timing is done on Tesla V100 GPU with batch size 1.

For Cityscapes experiments, our models are trained using a batch size of 1 per GPU, weight decay of $1e^{-4}$, learning rate of 0.013 for 48k total steps, decreasing by a factor of 0.1 at step 36k and 44k. We apply a random crop of 1800×900 and re-scale the crops randomly between $(0.7, 1.3)$. For COCO experiments, our models are trained using a batch size of 2 per GPU, weight decay of $1e^{-4}$, learning rate of 0.01 for 180k steps with learning rate with steps at 120k and 160k. For data augmentation, we randomly resize the input image to a shortest side length in $(640, 720, 800)$. No cropping is applied.

All our models use a ResNet-50 Backbone with ImageNet pretrained weights provided by Pytorch [26]. We freeze BatchNorm layers in the backbone for simplicity, similar to [36, 15].

5.4. Comparison to State of the Art

We compare the accuracy and efficiency of our proposed model to the state-of-the-art two-stage and single-stage panoptic segmentation algorithms that use a ResNet-50 or lighter backbone. We only report and compare to single-model prediction results to avoid ambiguity of the inference time during test-time augmentation. For inference time, we report the average inference time plus NMS processing time over the whole validation set. For Cityscapes, our model takes the full resolution as input size. For COCO, we resize all images to a longer side of 1333px as input. Our quantitative results are reported in Table 1 and Table 2.

Our model outperforms all the single-stage methods by a significant margin in both accuracy and inference speed. We are also closing the gap between the single-stage and slow but state-of-the-art two-stage methods, even outperforming some of them. To better highlight the potential of our method towards deployment in real-world systems and its parallelization benefits, we conduct a simple optimization by compiling the convolutional layers in our model using TensorRT [2]. It enables our model to operate in real-time (30.3 FPS on a V100 GPU) at full resolution on Cityscapes videos vs. 10.1 FPS without optimization (cf. Figure 1).

We also provide some qualitative examples comparing to one of the best two-stage methods, UPSNet [36], in Figure 4. Our method presents little degradation compared to UPSNet. In fact, our models provide better mask estimation on rare-shape instances (cf. the car with a roof-box,

Method	Backbone	PQ	PQ^{th}	PQ^{st}	mIoU	AP	GPU	Inference Time
Two-Stage								
TASCNet [15]	ResNet-50-FPN	55.9	50.5	59.8	-	-	V100	160ms
AUNet[16]	ResNet-50-FPN	56.4	52.7	59.0	73.6	33.6	-	-
Panoptic-FPN [13]	ResNet-50-FPN	57.7	51.6	62.2	75.0	32.0	-	-
AdaptIS† [30]	ResNet-50	59.0	55.8	61.3	75.3	32.3	-	-
UPSNet [36]	ResNet-50-FPN	59.3	54.6	62.7	75.2	33.3	V100	140ms*
Seamless Panoptic [28]	ResNet-50-FPN	60.2	55.6	63.6	74.9	33.3	V100	150ms*
Single-Stage								
DeeperLab [38]	Wider MNV2	52.3	-	-	-	-	V100	251ms
FPSNet [7]	ResNet-50-FPN	55.1	48.3	60.1	-	-	TITAN RTX	114ms
SSAP [8]	ResNet-50	56.6	49.2	-	-	**31.5**	1080Ti	>260ms
DeeperLab [38]	Xception-71	56.5	-	-	-	-	V100	312ms
Ours	ResNet-50-FPN	**58.8**	**52.1**	**63.7**	**77.0**	29.8	V100	**99**ms

Table 1: **Performance on Cityscapes validation set**. We bold the best number across single-stage methods and underline the best number across the two categories. †: method includes multiple-forward passes. ∗: Our replicated result from official sources using the same evaluation environment as our model.

Method	Backbone	PQ	PQ^{th}	PQ^{st}	Inf. Time
Two-Stage					
Panoptic-FPN [13]	ResNet-50-FPN	33.3	45.9	28.7	-
AdaptIS† [30]	ResNet-50	35.9	40.3	29.3	-
AUNet [16]	ResNet-50-FPN	39.6	49.1	25.2	-
UPSNet [36]	ResNet-50-FPN	42.5	48.5	33.4	110ms*
Single-Stage					
DeeperLab [38]	Xcep-71	33.8	-	-	94ms
SSAP [8]	ResNet-50	36.5	-	-	-
Ours	ResNet-50-FPN	**37.1**	**41.0**	**31.3**	**63**ms

Table 2: **Performance on COCO-validation**. We bold the best single-stage methods and underline the best across the two categories.†: methods including multiple-forward passes. ∗: Our replicated result from official sources using the same evaluation environment as our model.

or the small child) thanks to the self-attention in mask association. We observe particularly good results on unusual shapes, because unlike Mask-RCNN style methods[10], our mask construction process can associate pixels with a detection even if they fall outside its bounding box.

5.5. Ablative Analysis

We provide an ablative analysis on the key modules of our method in Table 3.

The first row presents a simplified baseline with *one* feature tower in the panoptic head that is used for both localization and semantics. We note that this baseline already leads to strong performance that is better than some two-stage methods reported in Table 1. This architecture can be used to achieve even greater speedups. In the second row, we can see how using two separate feature towers, one for localization and one for semantics, improves model performance slightly.

Two towers	Levelness	Mask loss	PQ	PQ^{th}	PQ^{st}
Fully Supervised					
			56.8	48.1	63.1
✓			57.1	47.8	**63.8**
✓	✓		58.1	50.4	63.7
✓	✓	✓	**58.8**	**52.1**	63.7
Weakly Supervised (No mask label)					
✓	✓		55.7	45.2	63.3

Table 3: **Ablative analysis**. We compare the impact of different key modules/designs in our proposed network. We also present a weakly supervised model trained without using instance masks.

Then, we introduce levelness, which leads to almost a 1 point boost in PQ. Without levelness, it is still possible to compute $\hat{P}_{loc}$ as in Eq. 6. We can compute the IoU between each query box and the predicted bounding boxes from every FPN level, and then take a max along the FPN levels. However, this operation suffers from ambiguity between object boundaries and background.

Finally, in the fourth row, we introduce our explicit mask loss from Section 4.3 which further refines mask association and construction for *thing* classes resulting in a higher PQ^{th} and corresponding bump in PQ.

5.6. Weakly supervised extension

We provide a simple yet promising extension of our model to the weakly supervised setting in which we have ground truth bounding boxes and semantic segmentation but no instance masks. A few straightforward changes are required: (1) regarding bounding box target assignment described in section 4.1, we now consider a pixel foreground as long as it is inside a ground truth bounding box, (2)

we update the foreground indicator function $\mathbb{1}_{\mathbf{fg}}(x, y)$ in Eq. 11 and Eq. 12 from 'in-mask' to 'in-box', and (3) we train without using the Mask Loss. Our weakly supervised Cityscapes model achieves promising accuracy and even outperforms some fully supervised methods, as shown in the last row of Table 3.

6. Conclusion

In this work, we propose a single-stage panoptic segmentation framework that achieves real-time inference with a performance competitive with the current state of the art. We first introduce a novel parameter-free mask construction operation that reuses predictions from dense object detection via a global self-attention mechanism. Our architecture dramatically decreases computational complexity associated with instance segmentation in conventional panoptic segmentation algorithms. Additionally, we develop an explicit mask loss which improves panoptic segmentation quality. Finally, we evaluate the potential of our method in weakly supervised settings, showing that it can outperform some recent fully supervised methods.

Acknowledgments

We would like to thank the TRI ML team, especially Dennis Park and Wolfram Burgard for their support and insightful comments during the development of this work.

Figure 4: **Panoptic segmentation results** on CityScapes and COCO comparing our predictions and UPSNet [36]. We leave it to the reader to guess which results are ours (the answer is in [1]).

References

[1] Ours results are in the middle for row 2,5 and on the right for row 1,3,4.

[2] TensorRT python library. `https://developer.nvidia.com/tensorrt`.

[3] Daniel Bolya, Chong Zhou, Fanyi Xiao, and Yong Jae Lee. YOLACT: real-time instance segmentation. *CoRR*, abs/1904.02689, 2019.

[4] Kai Chen, Jiangmiao Pang, Jiaqi Wang, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jianping Shi, Wanli Ouyang, et al. Hybrid task cascade for instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4974–4983, 2019.

[5] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.

[6] Bert De Brabandere, Davy Neven, and Luc Van Gool. Semantic instance segmentation with a discriminative loss function. *arXiv preprint arXiv:1708.02551*, 2017.

[7] Daan de Geus, Panagiotis Meletis, and Gijs Dubbelman. Fast panoptic segmentation network. *arXiv preprint arXiv:1910.03892*, 2019.

[8] Naiyu Gao, Yanhu Shan, Yupei Wang, Xin Zhao, Yinan Yu, Ming Yang, and Kaiqi Huang. Ssap: Single-shot instance segmentation with affinity pyramid. *arXiv preprint arXiv:1909.01616*, 2019.

[9] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Simultaneous detection and segmentation. In *European Conference on Computer Vision*, pages 297–312. Springer, 2014.

[10] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[11] 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.

[12] Zhaojin Huang, Lichao Huang, Yongchao Gong, Chang Huang, and Xinggang Wang. Mask scoring r-cnn. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6409–6418, 2019.

[13] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6399–6408, 2019.

[14] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9404–9413, 2019.

[15] Jie Li, Allan Raventos, Arjun Bhargava, Takaaki Tagawa, and Adrien Gaidon. Learning to fuse things and stuff. *arXiv preprint arXiv:1812.01192*, 2018.

[16] Yanwei Li, Xinze Chen, Zheng Zhu, Lingxi Xie, Guan Huang, Dalong Du, and Xingang Wang. Attention-guided unified network for panoptic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7026–7035, 2019.

[17] Xiaodan Liang, Liang Lin, Yunchao Wei, Xiaohui Shen, Jianchao Yang, and Shuicheng Yan. Proposal-free network for instance-level object segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2978–2991, 2017.

[18] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[19] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[20] 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.

[21] Huanyu Liu, Chao Peng, Changqian Yu, Jingbo Wang, Xu Liu, Gang Yu, and Wei Jiang. An end-to-end network for panoptic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6172–6181, 2019.

[22] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8759–8768, 2018.

[23] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulo, and Peter Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4990–4999, 2017.

[24] Davy Neven, Bert De Brabandere, Marc Proesmans, and Luc Van Gool. Instance segmentation by jointly optimizing spatial embeddings and clustering bandwidth. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8837–8845, 2019.

[25] Alejandro Newell, Zhiao Huang, and Jia Deng. Associative embedding: End-to-end learning for joint detection and grouping. In *Advances in Neural Information Processing Systems*, pages 2277–2287, 2017.

[26] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[27] Tobias Pohlen, Alexander Hermans, Markus Mathias, and Bastian Leibe. Full-resolution residual networks for semantic segmentation in street scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4151–4160, 2017.

[28] Lorenzo Porzi, Samuel Rota Bulo, Aleksander Colovic, and Peter Kontschieder. Seamless scene segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8277–8286, 2019.

[29] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[30] Konstantin Sofiiuk, Olga Barinova, and Anton Konushin. Adaptis: Adaptive instance selection network. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7355–7363, 2019.

[31] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. *arXiv preprint arXiv:1904.01355*, 2019.

[32] Jonas Uhrig, Eike Rehder, Björn Fröhlich, Uwe Franke, and Thomas Brox. Box2pix: Single-shot instance segmentation by assigning pixels to object boxes. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 292–299. IEEE, 2018.

[33] Mark Weber, Jonathon Luiten, and Bastian Leibe. Single-shot panoptic segmentation, 2019.

[34] Zifeng Wu, Chunhua Shen, and Anton van den Hengel. Bridging category-level and instance-level semantic image segmentation. *arXiv preprint arXiv:1605.06885*, 2016.

[35] Enze Xie, Peize Sun, Xiaoge Song, Wenhai Wang, Xuebo Liu, Ding Liang, Chunhua Shen, and Ping Luo. Polarmask: Single shot instance segmentation with polar representation. *arXiv preprint arXiv:1909.13226*, 2019.

[36] Yuwen Xiong, Renjie Liao, Hengshuang Zhao, Rui Hu, Min Bai, Ersin Yumer, and Raquel Urtasun. Upsnet: A unified panoptic segmentation network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8818–8826, 2019.

[37] Wenqiang Xu, Haiyang Wang, Fubo Qi, and Cewu Lu. Explicit shape encoding for real-time instance segmentation. *arXiv preprint arXiv:1908.04067*, 2019.

[38] Tien-Ju Yang, Maxwell D. Collins, Yukun Zhu, Jyh-Jing Hwang, Ting Liu, Xiao Zhang, Vivienne Sze, George Papandreou, and Liang-Chieh Chen. Deeperlab: Single-shot image parser. *CoRR*, abs/1902.05093, 2019.

[39] Jiahui Yu, Yuning Jiang, Zhangyang Wang, Zhimin Cao, and Thomas Huang. Unitbox: An advanced object detection network. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 516–520. ACM, 2016.