Appendix

A. Additional Ablations

A.1. Fully supervised performance

We report fully supervised performance of Deep-MAC and Deep-MARC in Table 9.

Model	Backbone	AP	AP_S	AP_M	AP_L
ShapeMask	RF101	37.4	16.1	40.1	53.8
ShapeMask	RNF101	40.0	18.3	43.0	57.1
CPMask	RF101	39.2	22.2	41.8	50.1
Deep-MAC	HG104	39.4	20.5	41.9	54.0
Deep-MARC	SN143	42.8	24.3	46.0	60.5

Table 9: Fully supervised instance segmentation performance on COCO `test-dev2017`. Backbones include RF=ResNet-FPN, RNF=ResNet-NAS-FPN, HG=Hourglass, SN=SpineNet. Deep-MAC is trained at 1024×1024 resolution with an HG-100 mask-head and Deep-MARC is trained at 1280×1280 resolution with HG-52 mask-head. Mask heads are explored in detail in Section 6. We report mAP of `coco-testdev2017`.

A.2. Deep-MAC

A.2.1 Effect of instance and coordinate embedding

Table 10 shows the effects of coordinate embedding and instance embedding on ResNet and Hourglass mask heads. We notice that the additional embeddings do not make a significant difference to the Hourglass model, but coordinate embedding is required for the ResNet based mask heads to converge. For uniformity, we have thus used both components in all Deep-MAC variants.

Mask Head	C	I	Mask mAP		
			Overall	VOC	Non-VOC
ResNet-20			–	–	–
		✓	–	–	–
	✓		30.9	39.1	28.2
	✓	✓	31.4	39.1	28.8
HG-20			34.1	39.8	32.2
		✓	34.5	39.9	32.7
	✓		33.6	39.9	31.5
	✓	✓	34.3	39.8	32.5

Table 10: Effect of Coordinate Embedding (C) and Instance Embedding (I) on the generalization ability of Deep-MAC on unseen classes. A '–' indicates that the model failed to converge. All models are trained with masks only from VOC classes at an input image resolution of 512×512. Performance is reported with the `VOC-Masks-Only` setup.

FCN layers	Mask mAP		
	Overall	VOC	Non-VOC
2	29.1	38.4	26.0
4	30.5	37.5	28.2

Table 11: Effect of using fully connected layers as mask-heads on Deep-MAC . Performance is reported with the `VOC-Masks-Only` setup. For easy reference, the VOC/non-VOC mask mAP values for Resnet-4 and HG-52 mask-heads are 39.7/26.6 and 39.8/32.5 respectively.

Variant	Mask mAP
Class-specific (Proposals + GT)	37.2
Class-agnostic (Proposals + GT)	36.7
Class-agnostic (GT only)	36.4

Table 12: Fully supervised mask mAP of Mask-RCNN variants with a ResNet-50-FPN backbone.

A.2.2 Effect of using fully connected layers

See Table 11 for experiments with fully connected layers. We used Glorot normal initialization [13] the mask-head weights. Based on these results, we see that the fully connected mask-head models, which have full receptive field with respect to the input tensor, do not offer competitive performance compared to the HG-based mask-heads. However, early large receptive fields may still be beneficial to some extent as these fully connected mask-heads do outperform our shallowest convolution-only mask-heads (e.g. Resnet-4).

A.3. Mask R-CNN

Table 12 shows the impact of using groundtruth boxes (instead using proposals, which is the standard approach) for training the mask-head of a fully supervised Mask R-CNN model on COCO. First we see that using a class-agnostic mask head results in a slightly lower mask mAP compared to the standard class-specific mask-head. Training with groundtruth boxes instead of proposals does not further impact the performance of the class agnostic mask head significantly.

B. Using Deep-MAC just for its masks for two-stage training

In this section we show that it is detection quality rather than mask quality which is now the bottleneck to achieving even better performance on the partially supervised COCO task, at least with respect to the mAP metric. With this insight, we use Deep-MAC to label instance masks on classes where they do not already exist and train a model with better detection performance on the resulting data.

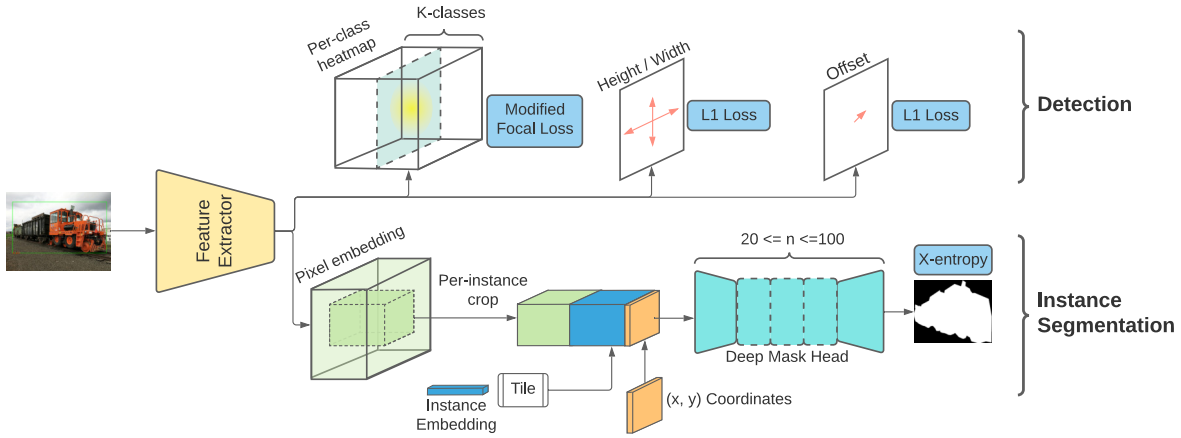


Figure 4: Schematic of the Deep-MAC architecture. The top-half is kept identical to CenterNet [55] and the bottom-half uses an RoI crop followed by a deep mask head. In our experiments, it was crucial to train the mask head with only groundtruth boxes.

Model	B.B.	Mask mAP		
		Overall	VOC	non-VOC
Deep-MAC [R4]	HG104	37.8	42.2	36.3
Mask R-CNN	RF50	36.1	40.2	34.7
Mask R-CNN	SN143	41.9	46.4	**40.4**

Table 13: Using Deep-MAC generated pseudo labels to train other models. Deep-MAC is trained as described in Table 9 on pseudo labels and evaluated on the `coco-val2017` set. Other models are trained with their default settings. Backbones(B.B.) include HG=Hourglass, RF=ResNet-FPN, SN=SpineNet. R4=ResNet-4 mask-heads. For reference, the "teacher" Deep-MAC model achieves a non-VOC mAP of 35.5% (c.f. Table 8).

B.1. Limited headroom on COCO mask quality.

If, as with the detector-free model from Section 6, we run the mIOU evaluation on the Deep-MAC model cropping to groundtruth boxes, we obtain 81.4% which is slightly better than the detection-free model. To put this number in perspective, [14] showed that COCO groundtruth masks achieve 83%-87% mIOU when compared to expert labels. Thus our finding suggests that remaining headroom on improving segmentation quality is quite limited (we are likely close to a saturation point). Note that this is not to say that our models have reached human level performance, since COCO annotation quality is known to be lower that some more recent datasets (e.g., LVIS [14]). However, our finding does suggest that future improvements on the partially supervised task on COCO as measured by mean AP will be much easier to come by via improvements to detection quality as opposed to segmentation quality.

B.2. Two stage (self-distillation style) training for improved mAP or cheaper models.

To illustrate, we use Deep-MAC just for its masks (and not its boxes), first segmenting unseen categories and then training a stronger detection model (Mask R-CNN with SpineNet [6], which reaches 48.6% box AP compared to Deep-MAC which reaches 44.1% box AP) in fully supervised mode on these pseudo labels. Table 13 (last row) shows the result of this experiment — specifically, Mask R-CNN with SpineNet is able to leverage the pseudo labels to get to a 40.4% non-VOC mask mAP, which is significantly higher than the original model that generated the pseudo labels. Thus improving box detection quality leads to a significantly increased final non-VOC mAP which is not upper bounded by the non-VOC mAP of Deep-MAC itself. This is also the highest performance ever reported on the partially supervised task by a margin of 6.4% (but only by virtue of better detection and without improving generalization to novel classes).

Our recommendation, consequently, is that the community should focus on harder tasks either by training with even fewer mask annotations, or evaluate partially supervised performance on LVIS [14] which has more classes and higher quality masks. As an initial step, we train Deep-MAC on COCO masks from all 80 categories and evaluate mIOU on LVIS masks (from the `v1-val` set) cropping to LVIS groundtruth boxes. Here our models using ResNet-4 and HG-100 mask-heads achieve 70.3% and 79.9% mIOU respectively, showing that architecture continues to matter for strong mask generalization even on LVIS. Comparing to [14] who report 90-92% mIOU dataset-to-expert agreement, we also see that there is still a gap between Deep-MAC and human performance (but this is likely at least in part due to COCO's lower quality masks).

Another application of two stage training is to train a cheaper instance segmentation model on masks produced by Deep-MAC. The first two rows of Table 13 demonstrate results using a cheaper Mask R-CNN model or Deep-MAC model with a shallower (4 layer) mask-head. This experiment is particularly interesting in the case of the student

Deep-MAC model with the shallow head since in this two stage setting, the student trains as if it were being fully supervised. According to our findings in Section 6, we should therefore expect it to achieve the same performance as Deep-MAC with the heavier mask-head (which it does, and even exceeds). Thus for COCO categories we are able to leverage the strong mask generalization properties of the heavier mask-head while retaining the computational benefits of the cheaper mask-head. When running at 1024×1024 resolution on a V100 GPU, Deep-MAC with an HG-100 mask-head takes 232 ms per image, whereas the cheaper student model with a ResNet-4 mask-head is faster (201 ms per image). Notably, this cheaper student model is on par with ShapeMask [27] in terms of speed (204 ms) while achieving a 2.1 % improvement on non-VOC mAP.[2]

C. Generalizing from a single class

In the majority of our experiments, we assumed the standard setup of "train-on-VOC, test-on-non-VOC". In this section, we restrict further, training on a single "source" class at a time, in order to better understand when Deep-MAC can be expected to strongly generalize to a novel class. In Figure 5a we plot results from this experiment, training on each of the VOC categories with 512×512 resolution inputs and an Hourglass-52 mask head. We observe that while some classes lead to strong performance, there is high variance depending on the source category (ranging from 12.5% mAP to 27.8% mAP). Notably, a single class can achieve strong results — as one datapoint, training only on the chair category with higher resolution 1024×1024 inputs yields a non-VOC mask mAP of 31.5, which is competitive with previous high-performance methods (e.g., ShapeMask [27]) when trained on all VOC categories.

In some cases it is easy to guess why a class might be a poor source — on the worst classes, we see that the quality of groundtruth masks is uneven in COCO. For example, labelers were not consistent about excluding objects that were on but not part of a dining table (see Section E).

For more detailed insight, we ask how training on a specific source class might generalize to a specific new target class. For source-target pairs (i, j), Figure 5b visualizes this relationship via the ratio between mAP on target class j if we were to train on just the source class i to mAP on target class j if we were to train on all classes. Here we cluster the rows and columns by similarity and truncate ratios to be at most 1.0 for visualization purposes.[3]

Figure 5b illustrates that some classes (e.g., apple [52], umbrella [45], stop sign [42]) are universally easy transfer targets likely due to being visually salient, having consistent appearance and not typically co-occurring with other examples of their own class. We also see that co-occurrence of source and target classes does not always lead to improved ratios (i.e. close to 1). For example, training on car does not yield strong performance on stop signs [42] or parking meters [43] and training on person does not yield strong performance on bench [01] or baseball bat [09]. On the other hand, categories that are similar semantically seem to function similarly as source categories, and with a few exceptions, the source categories cluster naturally into two broad groups: man-made and natural objects.

It remains an open question why a class might excel as a source class in general. Intuitively one might think that person, car or chair categories might be the best because they have the most annotations and are visually diverse, but perhaps surprisingly, using the bottle category is the best. This may be due to the fact that bottles tend to look alike and appear in groups, forcing the model to make non-local decisions about mask boundaries. We leave exploration of this hypothesis for future work.

D. Example images on unknown classes

See Figure 6 for example outputs of Deep-MAC . We look at the output of our model with user-specified boxes around object categories that are not in the COCO dataset. We observe that Deep-MAC generalizes to multiple different domains like biological and camera trap images and does well even in cluttered settings. For this experiment, we used a model trained with all COCO classes in fully supervised mode.

E. Looking at annotation quality

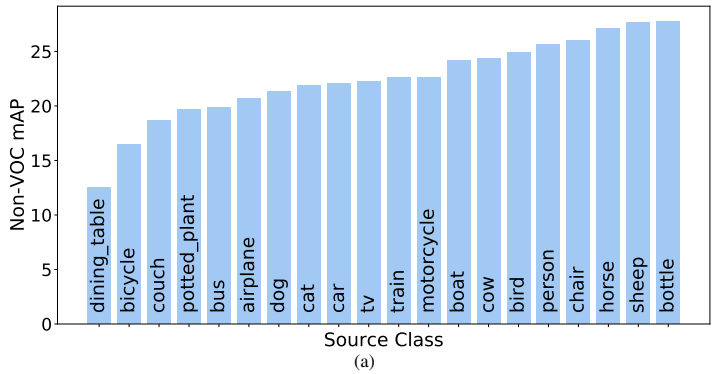
In Figure 7 we show examples of COCO groundtruth annotations from the dining table, bicycle and potted plant categories, the worst three categories to use as source training categories. The examples illustrate the inconsistencies/inaccuracies in mask annotations for these categories — for example, annotators were inconsistent about including or excluding objects on the dining tables.

F. Mask head architecture details

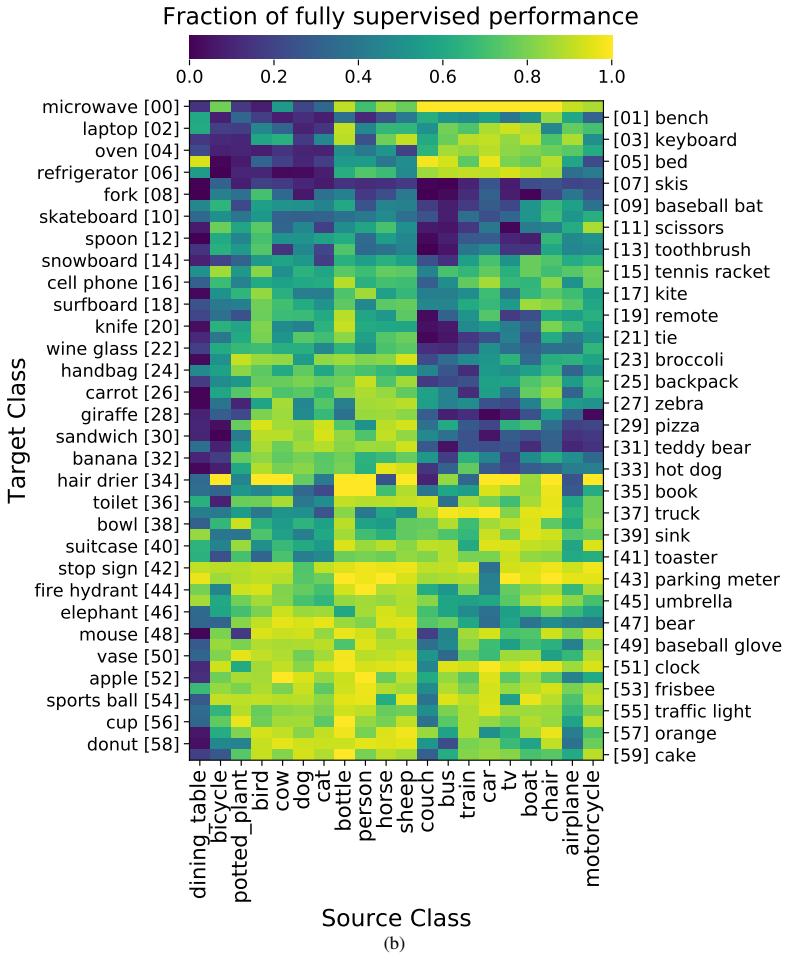
Details of mask head architectures can be found in Table 14, 15, 16 and 17. Figure 8 illustrates the computation graph of an hourglass mask head.

[2] Inference speed is not reported in CPMask [10]

[3] It is worth noting in several cases (most notably, hair drier [34]) that it is better to train on other source classes than it is to include the target class annotations during training.

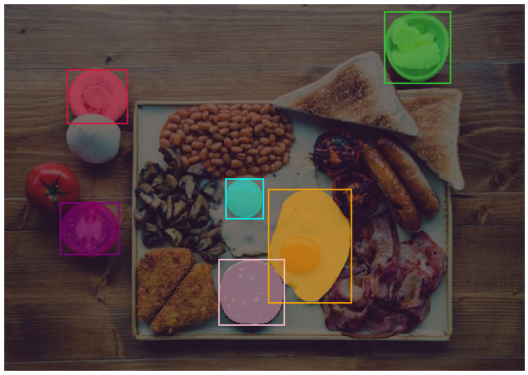


(a)

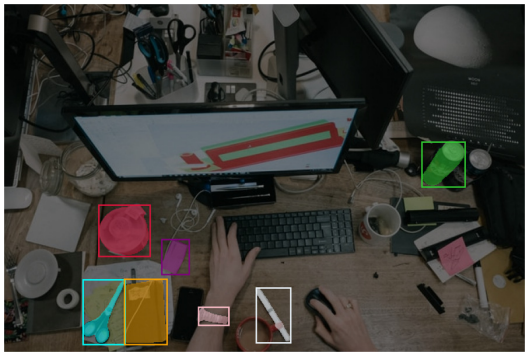


(b)

Figure 5: (a) Mask mAP on Non-VOC classes when training with masks from only a single source class from the VOC set; (b) Performance on specific (Non-VOC) target classes when training with masks from only a single class. We visualize performance relative to full supervision.

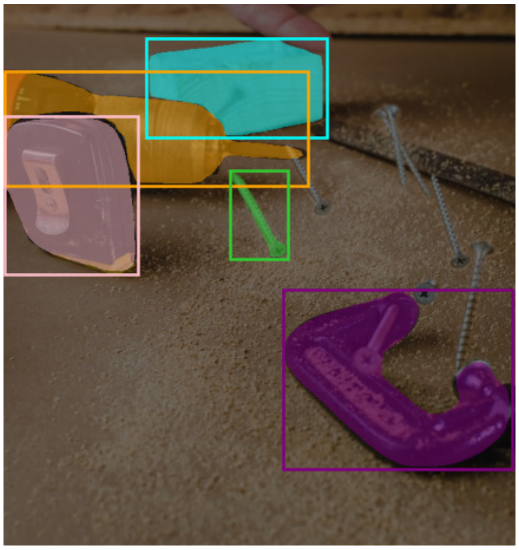


(a) Photo by Jonathan Farber on Unsplash.

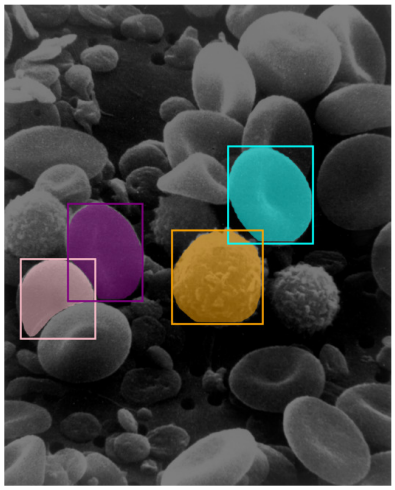


(b) Photo by Robert Bye on Unsplash.

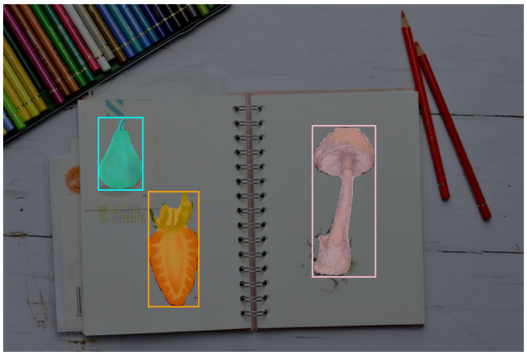
(c) Sample from the Snapshot Serengeti dataset.



(d) Photo by Chris Briggs on Unsplash.

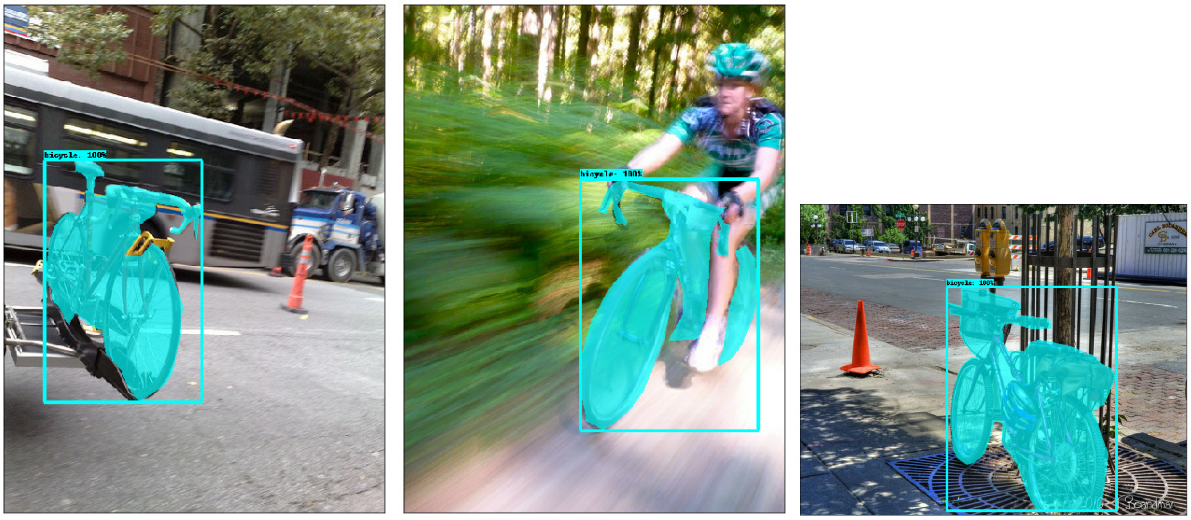


(e) SEM blood cells image from wikipedia.



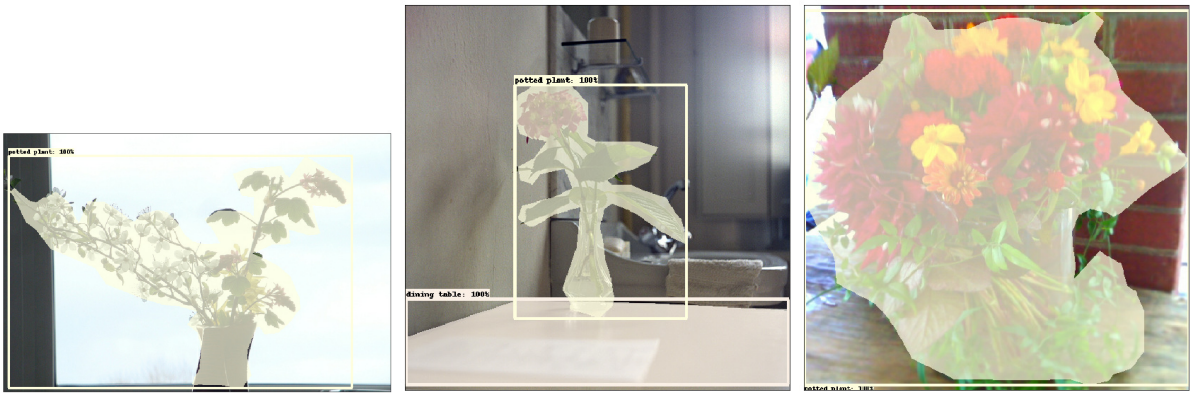
(f) Photo by Maggie Jaszowska on Unsplash.

Figure 6: Example outputs of Deep-MAC with hand-drawn boxes on unknown classes.



(a) Bicycle: The annotated masks don't capture the shape correctly, and quite often label parts of the background interspersed with the bicycle frame as foreground.

(b) Dining Table: Inconsistencies in annotated parts of dining tables. Left: Plate and cup with carrots is excluded whereas plate with empty glass is included in the mask. Center: Some glasses on the dining table are included as part of it whereas some classes aren't. Right: Chairs are excluded from the dining table mask in the dining tables near the bottom, whereas they are included in the dining table masks near the top.



(c) Potted plant: Areas of background are included in the foreground masks of potted plants, especially near the leaves.

Figure 7: Example annotations of the 3 worst source classes to train on.

Type	Depth	# of Blocks	Conv Block	
			Size	Channels
ResNet	4	1	32×32	64
		2	32×32	128
			32×32	128
	8	1	32×32	64
		4	32×32	128
			32×32	128
	12	1	32×32	64
		6	32×32	128
			32×32	128
	16	1	32×32	64
		8	32×32	128
			32×32	128
	20	1	32×32	64
		8	32×32	128
			32×32	128
		2	32×32	128
			32×32	128
ResNet Bottleneck	6	1	32×32	64
		2	32×32	128
			32×32	512
			32×32	128
	9	1	32×32	64
		3	32×32	128
			32×32	512
			32×32	128
	12	1	32×32	64
		4	32×32	128
			32×32	512
			32×32	128
	15	1	32×32	64
		5	32×32	128
			32×32	512
			32×32	128
	21	1	32×32	64
		6	32×32	128
			32×32	512
			32×32	128
		1	32×32	192
			32×32	384
			32×32	192

Table 14: Architecture details of ResNet and ResNet bottleneck mask heads.

Type	Depth	# of Blocks	Conv Block	
			Size	Channels
ResNet Bottleneck [1/4$^{\text{th}}$]	6	1	32×32	16
		2	32×32	32
			32×32	128
			32×32	32
	12	1	32×32	16
		4	32×32	32
			32×32	128
			32×32	32
	21	1	32×32	16
		6	32×32	32
			32×32	128
			32×32	32
		1	32×32	48
			32×32	192
			32×32	48
	30	1	32×32	16
		5	32×32	32
			32×32	128
			32×32	32
		5	32×32	48
			32×32	192
			32×32	48
	51	1	32×32	16
		6	32×32	32
			32×32	128
			32×32	32
		8	32×32	48
			32×32	192
			32×32	48
		3	32×32	64
			32×32	256
			32×32	64

Table 15: Architecture details of ResNet bottleneck [1/4$^{\text{th}}$] mask head.

Type	Depth	# of Blocks	Conv Block	
			Size	Channels
Hourglass	10	1	32×32	64
		3	32×32	128
			32×32	128
		1	16×16	128
			16×16	128
		1	32×32	128
	20	1	32×32	64
		3	32×32	128
			32×32	128
		4	16×16	128
			16×16	128
		2	8×8	192
			8×8	192
		1	32×32	128
	32	1	32×32	64
		5	32×32	128
			32×32	128
		4	16×16	128
			16×16	128
		4	8×8	192
			8×8	192
		2	4×4	192
			4×4	192
		1	32×32	128
	52	1	32×32	64
		5	32×32	128
			32×32	128
		4	16×16	128
			16×16	128
		4	8×8	192
			8×8	192
		4	4×4	192
			4×4	192
		4	2×2	192
			2×2	192
		4	1×1	256
			1×1	256
		1	32×32	128

Table 16: Architecture details of Hourglass mask head (Part 1 of 2).

Type	Depth	# of Blocks	Conv Block	
			Size	Channels
Hourglass	100	1	32×32	64
		9	32×32	128
			32×32	128
		8	16×16	128
			16×16	128
		8	8×8	192
			8×8	192
		8	4×4	192
			4×4	192
		8	2×2	192
			2×2	192
		8	1×1	256
			1×1	256
		1	32×32	128

Table 17: Architecture details of Hourglass mask head (Part 2 of 2).

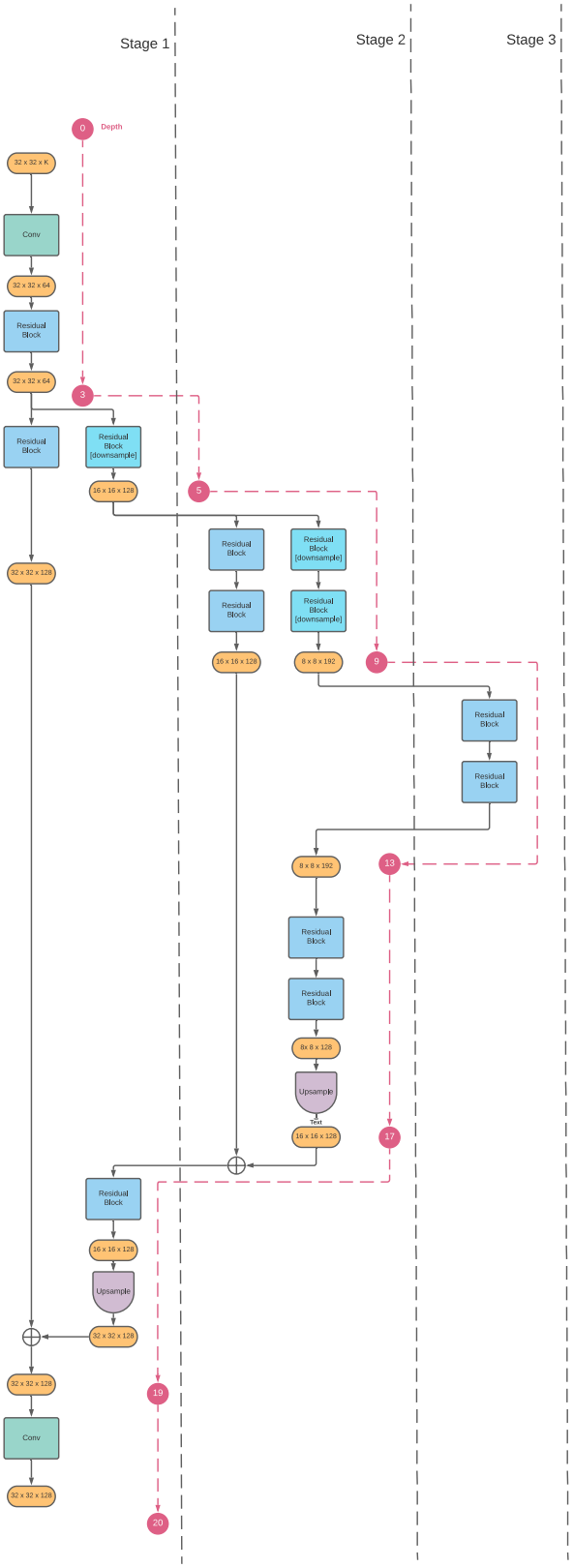


Figure 8: Illustration of the Hourglass 20 mask head computation graph.