Multi-Exit Semantic Segmentation Networks

Samsung Research

Alexandros Kouris[1], Stylianos I. Venieris[*,1], Stefanos Laskaridis[*,1], Nicholas D. Lane[1,2]

[1] Samsung AI Center - Cambridge, UK [2] University of Cambridge *Indicates Equal Contribution

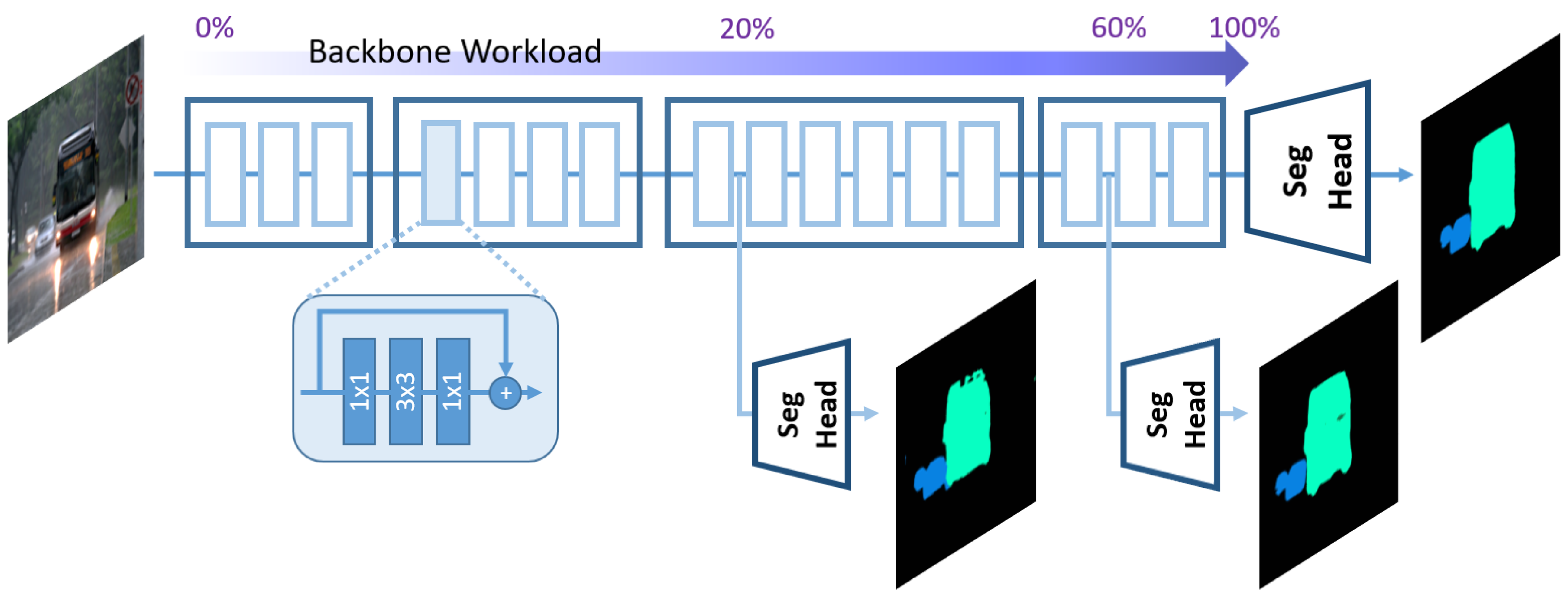
Overview

MESS is a framework that converts segmentation CNNs to multi-exit models:

- Specially trained models that employ parametrised exits along their depth.
- This allows to dynamically save computation during inference on easier samples.
- MESS jointly optimises the number, placement and architecture of early-exits.
- Saves training and maintenance cost by allowing post-training adaptation.

To enable this we also introduce:

- A novel two-stage training scheme for MESS networks, combining exit-aware backbone pre-training, with a selective self-distillation loss for the exits.
- An input-dependent inference pipeline for MESS networks, introducing a novel exit-policy, tailored to dense-output models, such as semantic segmentation.



Key findings

Compared to their respective backbones MESS networks are able to achieve:

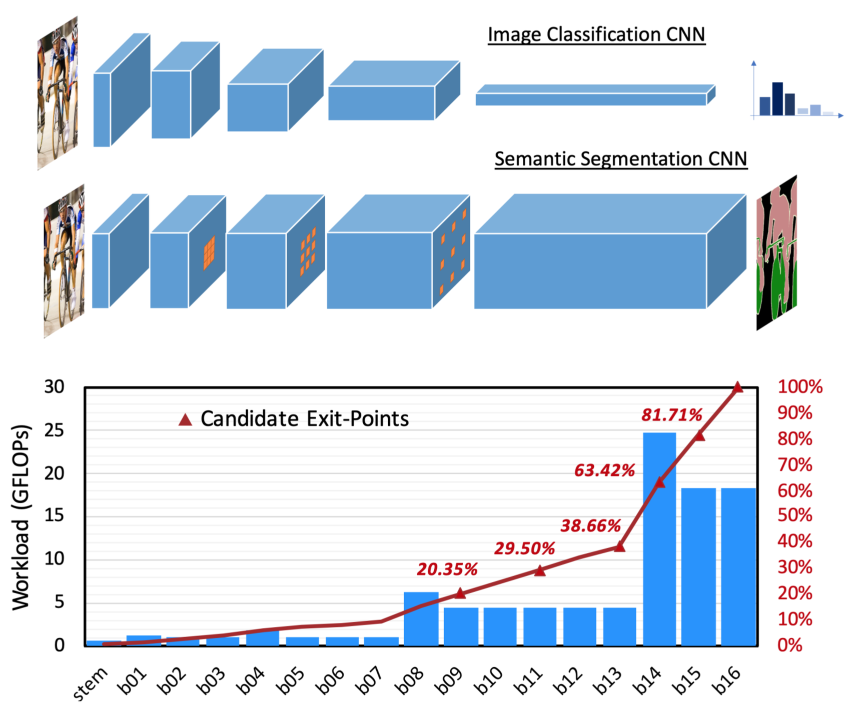
- Up to 2.8x faster inference, with the same accuracy
- Up to 5.3 pp higher accuracy, for the same computation budget

MESS networks also offer:

- Better (but comparable) speed-accuracy trade-off than SOTA NAS-crafted models, while being 10x faster to train.
- Up to 3x faster inference than uniform-exit MESS instances (status-quo).
- Gains that translate to realistic latency speed-ups out-of-the-box.
- Remain adaptable post-training via search (<1GPUh).

Candidate Exit-Points

Given a (user-provided) semantic segmentation model, MESS framework measures the workload/latency of each layer to identify N candidate exit-points with approximately equidistant workload distribution.



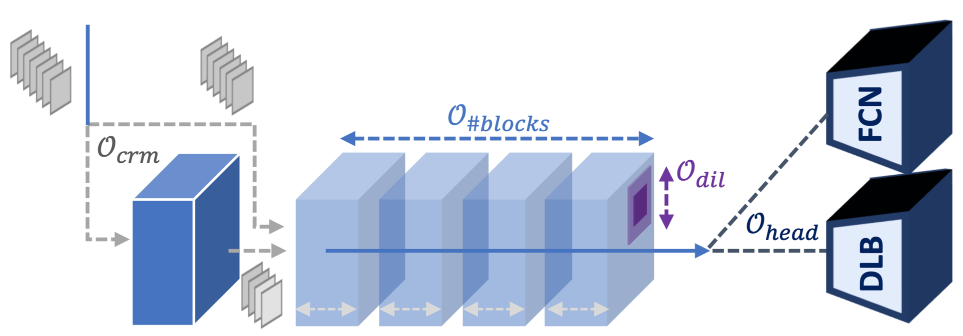
$$\mathcal{S} = (\mathcal{S}^1_{\text{exit}}+1)\times(\mathcal{S}^2_{\text{exit}}+1)\times \dots \times(\mathcal{S}^N_{\text{exit}}+1)$$

Exit Architecture

Different exit-point (depths) benefit from different architectural configuration of exits. Shallow exits suffer from *limited receptive field* and *weak semantics*, whereas deeper ones can introduce significant computational overhead due to large incoming feature volume. Early-exits can incorporate:

1. ***Channel Reduction module:*** reducing the number of channels and this computational overhead.
2. ***Extra Trainable Blocks:*** To remedy weak semantics.
3. ***Rapid Distillation Increase:*** To increase receptive field.
4. ***Segmentation head:*** FCN and DeepLab-based segmentation output layers supported.

$$\mathcal{S}^i_{exit} = \mathcal{O}_{crm}\times\mathcal{O}_{\#blocks}\times\mathcal{O}_{dil}\times\mathcal{O}_{head}$$



MESS Training

MESS networks are trained by a novel 2-stage scheme:

Stage1 - Exit-aware pre-training: trains the backbone in an exit-aware manner, pushing the extraction of semantically strong features to shallow layers, without committing to any exit configuration, or affecting the accuracy of deeper exits:

$$\mathcal{L}^{\text{batch}(j)}_{\text{pretrain}} = \sum_{i=1}^{N-1} \mathbb{1}(j \bmod i = 0)\cdot \mathcal{L}_{\text{CE}}(\boldsymbol{y}_i, \hat{\boldsymbol{y}}) + \mathcal{L}_{\text{CE}}(\boldsymbol{y}_N, \hat{\boldsymbol{y}})$$

Stage2 – Frozen backbone KD: Trains all candidate exit configurations, without any interference with the backbone (or between them). Allows massive parallelism during training, and easy interchange of exits upon deployment. Backed by a novel self-distillation scheme that filters the information propagating from the final exit to the earlier ones (allowing only "easy" pixels through).

$$\mathcal{L}_{\text{PFD}} = \sum_{i=1}^{N} \alpha\cdot\mathbb{1}(\hat{\boldsymbol{y}}_N = \hat{\boldsymbol{y}})\mathcal{L}_{\text{CE}}(\boldsymbol{y}_i, \hat{\boldsymbol{y}}) + (1-\alpha)\cdot\mathcal{L}_{\text{KL}}(\boldsymbol{y}_i, \boldsymbol{y}_N)$$

Search and Deployment

Upon deployment, MESS support different settings:

(a) Budgeted Inference: static workload-lighter sub-models (up to an exit) are extracted and deployed.

(b) Anytime Inference: every sample goes through several exits sequentially, providing an early actionable result while progressively refining its prediction.

(c) Input-Dependent Inference: similar to anytime, but each sample dynamically finalises its output at a different depth, based on a difficulty-aware exit policy.

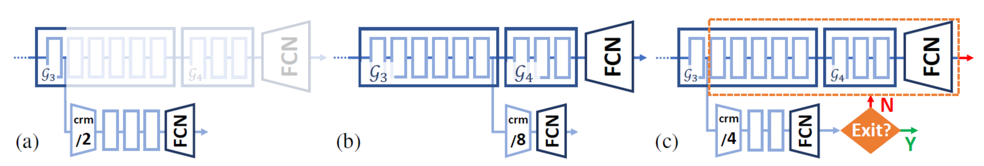
MESS instances can be optimised to:

- Minimise cost, given an accuracy constraint:

$$s^\star = \arg\min_{s\in\mathcal{S}} \{\text{cost}(s) \mid \text{acc}(s) \geq th_{\text{acc}}\}$$

- Maximise accuracy, given a cost constraint:

$$s^\star = \arg\max_{s\in\mathcal{S}} \{\text{acc}(s) \mid \text{cost}(s) \leq th_{\text{cost}}\}$$



Exit-Policy

We establish a mechanism to capture segmentation confidence on a *per-image* granularity, reducing the *per-pixel* confidence values provided with the model prediction. Our metric considers the percentage of pixels with high prediction confidence (above a tunable threshold) in the dense output (instead of naïve avg.).

$$c_i^{\text{img}} = \frac{1}{RC}\sum_{r=1}^{R}\sum_{c=1}^{C}\mathbb{1}(\boldsymbol{c}^{\text{map}}_{r,c}(\boldsymbol{y}_i) \geq th_i^{\text{pix}})$$

Predictions in pixel along the semantic boundaries of each object are naturally under confident. Thus we introduce a mechanism to down-weight the contribution of these pixels to the final result.

$$\mathcal{M} = \text{erode}(\text{cannyEdge}(\hat{\boldsymbol{y}}_i), s_i)$$

$$\widehat{\boldsymbol{c}^{\text{map}}_{r,c}}(y_i) = \begin{cases} \text{median}(\boldsymbol{c}^{\text{map}}_{w_r,w_c}(\boldsymbol{y}_i)) & \text{if } \mathcal{M}_{r,c} = 1 \\ \boldsymbol{c}^{\text{map}}_{r,c}(\boldsymbol{y}_i) & \text{otherwise} \end{cases}$$

At runtime, a *per-exit tunable threshold* is used to evaluate whether each sample should exit-early or continuer to the next (deeper) exit: $c_i^{\text{img}} \geq th_i^{\text{img}}$

Results

MESS End-to-End Evaluation

Method		Backbone*	Head	Search Targets: Error	Search Targets: GFLOPs	Results: MS COCO: mIoU	Results: MS COCO: GFLOPs	Results: MS COCO: Latency†	Results: PASCAL VOC: mIoU	Results: PASCAL VOC: GFLOPs	Results: PASCAL VOC: Latency†
DRN [68]	(i)	ResNet50	FCN	–Baseline–		59.02%	138.63	39.96ms	72.23%	138.63	39.93ms
Ours	(ii)	ResNet50	FCN	min	$\leq 1\times$	64.35%	113.65	37.53ms	79.09%	113.65	37.59ms
Ours	(iii)	ResNet50	FCN	$\leq 0.1\%$	min	58.91%	41.17	17.92ms	72.16%	44.81	18.63ms
Ours	(iv)	ResNet50	FCN	$\leq 1\%$	min	58.12%	34.53	15.11ms	71.29%	38.51	16.80ms
DLBV3 [6]	(v)	ResNet50	DLB	–Baseline–		64.94%	163.86	59.05ms	80.32%	163.86	59.06ms
Ours	(vi)	ResNet50	DLB	min	$\leq 1\times$	65.52%	124.10	43.29ms	82.32%	124.11	43.30ms
Ours	(vii)	ResNet50	DLB	$\leq 0.1\%$	min	64.86%	69.84	24.81ms	80.21%	65.29	24.14ms
Ours	(viii)	ResNet50	DLB	$\leq 1\%$	min	64.03%	57.01	20.83ms	79.30%	50.29	20.11ms
segMBNetV2 [50]	(ix)	MobileNetV2	FCN	–Baseline–		54.24%	8.78	67.04ms	69.68%	8.78	67.06ms
Ours	(x)	MobileNetV2	FCN	min	$\leq 1\times$	57.49%	8.10	56.05ms	74.22%	8.10	56.09ms
Ours	(xi)	MobileNetV2	FCN	$\leq 0.1\%$	min	54.18%	4.05	40.97ms	69.61%	3.92	32.79ms
Ours	(xii)	MobileNetV2	FCN	$\leq 1\%$	min	53.24%	3.48	38.83ms	68.80%	3.60	31.40ms

*Dilated network [68] based on backbone CNN. †Measured on: GTX for ResNet50 and AGX for MobileNetV2 backbone.

MESS Training Evaluation

stage 1:

Method	Init.	Loss	$\mathcal{E}_1$	$\mathcal{E}_2$	$\mathcal{E}_3$	$\mathcal{E}_4$	$\mathcal{E}_5$	$\mathcal{E}_6$
(i) Baseline Init.	ImageNet	$\mathcal{L}_{\text{CE}}(\mathcal{E}_6)$	-	-	-	-	-	59.02%
(ii) **E2E** [21,53]	ImageNet	$\mathcal{L}_{\text{CE}}(\mathcal{E}_1)+...+\mathcal{L}_{\text{CE}}(\mathcal{E}_6)$	29.02%	40.67%	48.64%	51.69%	55.34%	58.33%
(iii) Exit-aware Init.	ImageNet	Eq. (2) (**Ours**)	28.21%	39.61%	47.13%	50.81%	56.11%	**59.90%**
(iv) **Frozen** [25,28]	(i)	$\mathcal{L}_{\text{CE}}(\mathcal{E}_1)$, ..., $\mathcal{L}_{\text{CE}}(\mathcal{E}_5)$	23.94%	31.50%	38.24%	44.73%	54.32%	59.02%
(v) **Ours (§4.3)**	(iii)	$\mathcal{L}_{\text{CE}}(\mathcal{E}_1)$, ..., $\mathcal{L}_{\text{CE}}(\mathcal{E}_5)$	**32.40%**	**43.34%**	**50.81%**	**53.73%**	**57.9%**	**59.90%**

* Experiments repeated 3 times. The sample stdev in mean IoU is at most ± 0.09 in all cases.

stage 2:

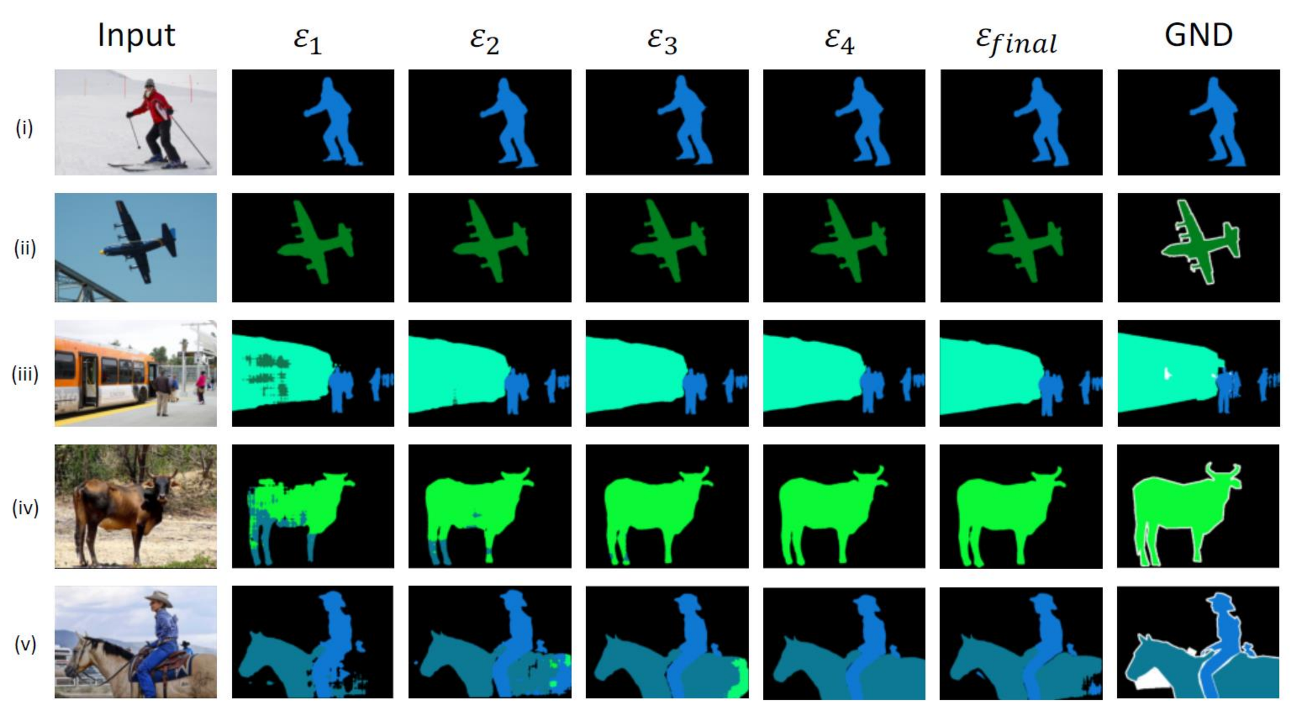
Method	Loss	DRN-50: $\mathcal{E}_1$	DRN-50: $\mathcal{E}_2$	DRN-50: $\mathcal{E}_3$	MobileNetV2: $\mathcal{E}_1$	MobileNetV2: $\mathcal{E}_2$	MobileNetV2: $\mathcal{E}_3$
E2E [21,53]	CE	49.96%	55.40%	58.96%	31.56%	41.57%	51.59%
KD [19]	KD	50.33%	55.67%	59.08%	31.04%	41.93%	51.66%
SelfDistill [73,48,72]	CE+KD	50.66%	55.91%	58.84%	32.08%	41.96%	51.58%
Ours (§4.3)	PFD	**51.02%**	**56.21%**	**59.36%**	**33.36%**	**42.95%**	**52.20%**

CE=Cross-entropy, KD=Knowledge Distillation, PFD=Positive Filtering Distillation

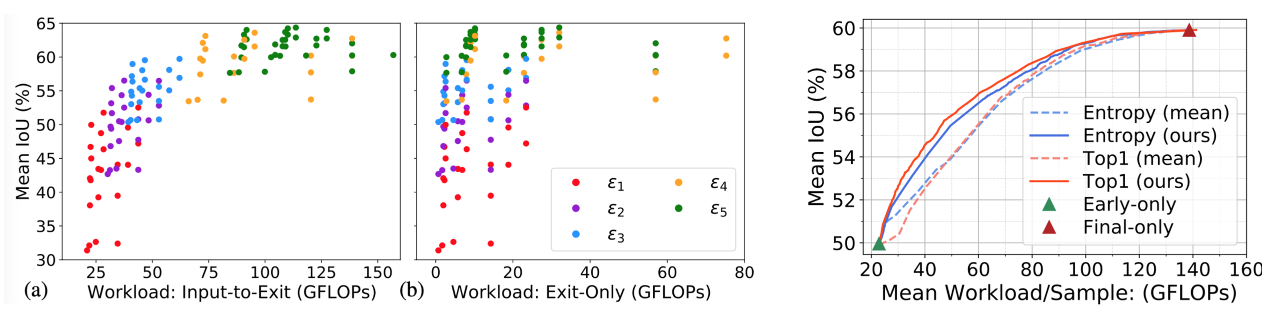
Reach out:

a.kouris@samsung.com
s.venieris@samsung.com
mail@stefanos.cc
nic.lane@samsung.com

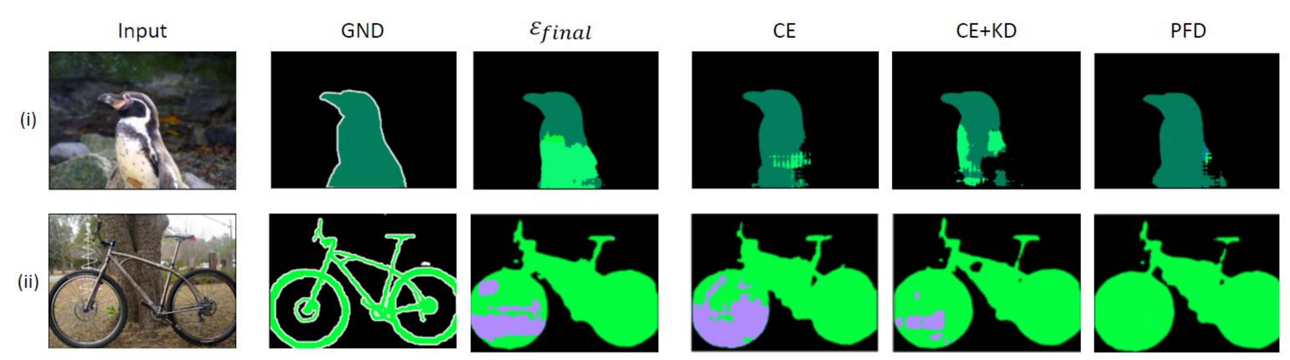
Learn more:



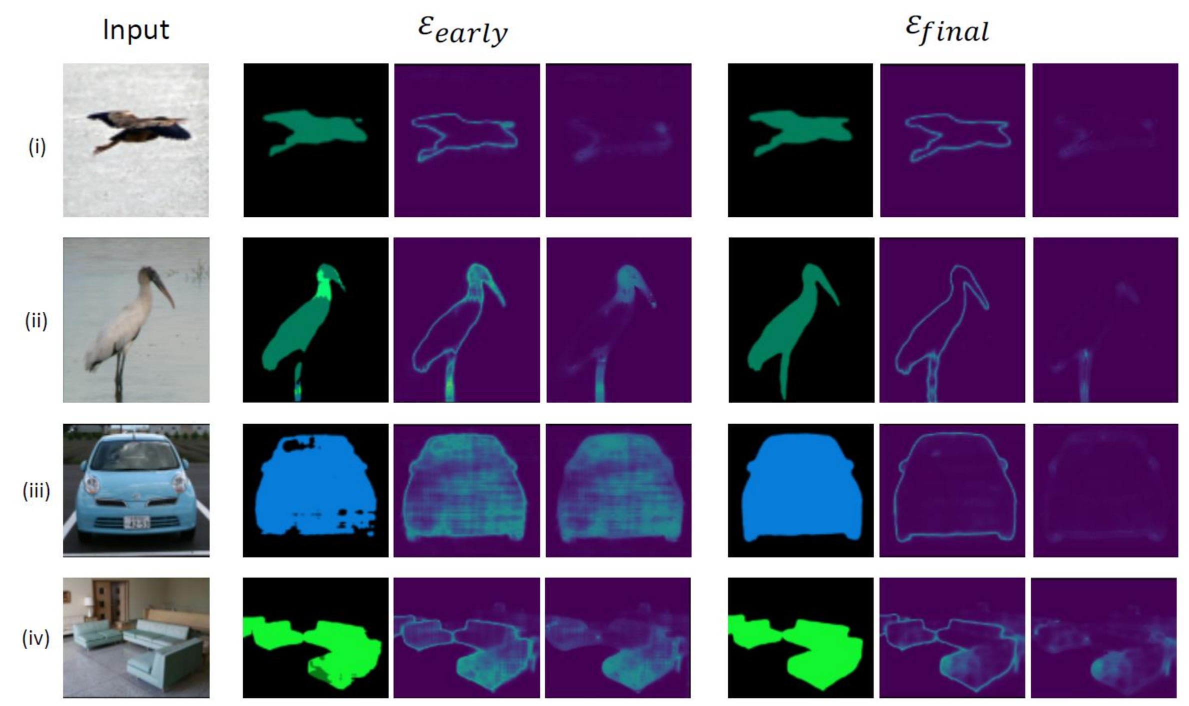
Segmentation predictions from consecutive exits of a MESS instance based on DRN-50.



(left): MESS search space for DRN-50. (right): Speed-accuracy trade-off in a 2-exit MESS model.



Comparison of distillation schemes used during MESS 2nd-stage training (frozen backbone).



Prediction confidence visualization (before and after) proposed refinement.