Adaptable Mobile Vision Systems through Multi-Exit Neural Networks

SAMSUNG

Imperial College London

Alexandros Kouris[♣♦], Stylianos I. Venieris[*♣], Stefanos Laskaridis[*♣], Nicholas D. Lane[♣♠]

[♣]Samsung AI Center, Cambridge [♦]Imperial College London (work while intern at Samsung) [♠]University of Cambridge

*Indicates equal contribution.

Why multi-exit segmentation?

- **Multi-exit networks** [1] offer a dynamic way of performing inference, without wasting resources on redundant computation for "overthinking".
- **Multi-exit networks** comprise of early exits attached to a backbone network, acting as early-evaluation outputs at inference, based on some exit policy.
- Applying this naively in the dense task of Semantic Segmentation quickly **negates the benefits** due to heavyweight exit heads.

Introduction

Semantic segmentation is a backbone task for many vision systems, spanning from robotic navigation and self-driving cars to augmented reality and teleconferencing. Such systems often operate under **stringent** latency constraints within the limited resources of an embedded/mobile device.

We propose **MESS**, a system that derives, trains and deploys Multi-Exit Semantic Segmentation networks for the task and device at hand in a train-once-deploy-everywhere manner.

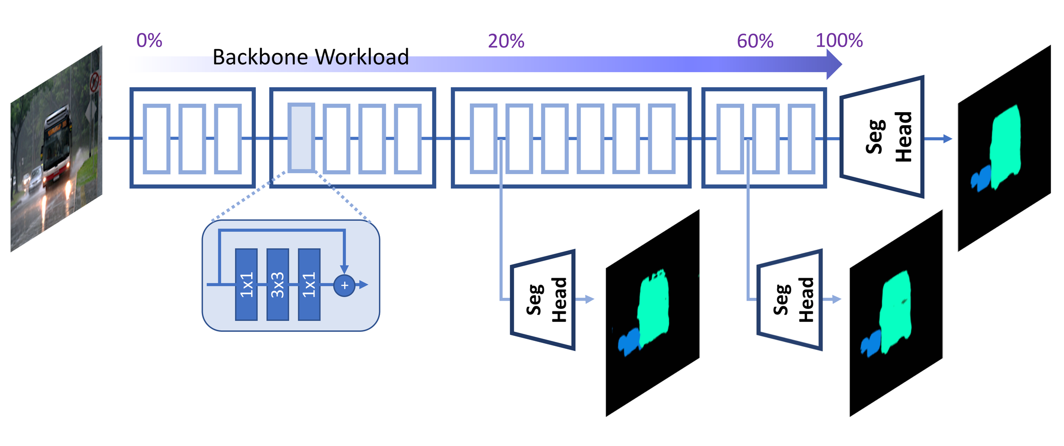

Figure: Visualisation of MESS's training process

Starting from a backbone network, MESS:

- **attaches** various segmentation heads of different configurations at various positions;
- **(pre-)trains** the overprovisioned network in an exit-aware manner;
- **searches** for the early-exit policy and configurations for the task and device at hand without the need to retrain.

Training Scheme

Training multi-exit networks

Multi-exit networks are typically trained:

End-to-end: Exits & backbone are jointly optimised. ↑ *accuracy*, ↑ *complexity*

IC-only: Backbone & exits are trained in sequence. ↑ *flexibility*, ↓ *degrees of freedom*

We propose a two-stage training scheme that combines the best of both worlds:

1. The backbone is trained considering the final and a single early exit at each iteration. Different exit points are dropped per iteration in an alternating fashion.
2. The backbone is then frozen and early exits of various configurations (*i.e.* architectures) are attached at different positions of the backbone and trained independently.

Architectural Choices

To design meaningful early-exit heads, we propose the following architectural configurations to push the extraction of semantically strong features earlier in the network and have lightweight heads:

- **Channel Reduction Module (CRM):** 1×1 convolution to reduce channels entering the segmentation head.
- **Extra Trainable Blocks**: Increase capacity of segmentation heads for meaningful semantics.
- **Rapid Dilation Increase (RDI):** Rapidly increase dilation rate for larger receptive field.
- **Selection of segmentation head**: Select from FCN and DeepLabV3 heads.

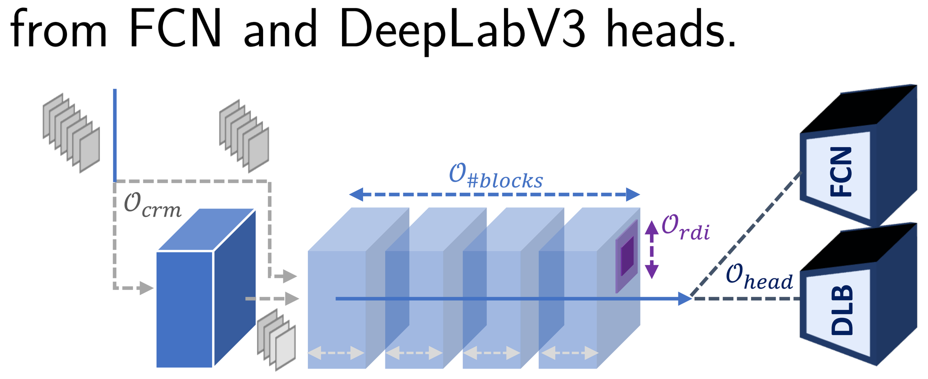

Figure: Parametrisation of segmentation head architecture.

Configuration Search

Having trained an overprovisioned network, we can select the exits for the use-case requirements (*e.g.* latency, accuracy, memory) and hardware at hand **without** the need for retraining the MESS network. Search can be performed exhaustively based on profiling from a validation set, several orders of magnitude faster than NAS-based solutions.

Deployment Scenarios

A variety of inference settings are supported by MESS networks so as to satisfy the performance needs under each device and application-specific constraints for different scenarios. These include:

1) budgeted inference

2) anytime inference

3) input-dependent inference

Prediction Confidence

The **exit policy** in multi-exit networks is usually based on a **confidence criterion** (*e.g.* top-1 softmax, entropy). To summarise per-pixel confidence values with a single per-image prediction confidence (c_i^{img}), we craft a **weighting mechanism** that considers the % of pixels in an image that yield confidence ($\boldsymbol{c}_{r,c}^{\text{map}}(\boldsymbol{y}_i)$) above a pre-specified threshold (th_i^{pix}) (Eq. (1)). Additionally, it **downweights** the contribution of pixels on semantic edges (Eq. (2,3)).

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

$$\mathcal{M} = \text{erode}(\text{cannyEdge}(\hat{\boldsymbol{y}}_i), s_i) \quad (2)$$

$$\boldsymbol{c}_{r,c}^{\text{map}}(y_i) = \begin{cases} \text{median}(\boldsymbol{c}_{w_r,w_c}^{\text{map}}(\boldsymbol{y}_i)) & \text{if } \mathcal{M}_{r,c} = 1 \\ \boldsymbol{c}_{r,c}^{\text{map}}(\boldsymbol{y}_i) & \text{otherwise} \end{cases} \quad (3)$$

where $w_l = \{l - 2 \cdot os_i, ..., l + 2 \cdot os_i\}$ is the window size of the filter. This *sets the pixels around semantic edges to inherit the confidence of their neighbouring pixel predictions.*

Experiments

Setup: MS-COCO for DRN-based [3] MESS instances on top of ResNet-50 and MobileNetV2 backbones. Latencies reported over GTX 2080 and Jetson Xavier AGX, respectively.

Quantitatively, a latency-optimised MESS instance achieves workload reduction of up to 3.36× (w/o accuracy drop) and up to 4.01× (≤ 1 pp of accuracy degradation). A MESS instance optimised for accuracy achieves mIoU of 5.33 pp higher than the baseline. Similar are the gains for MobileNetV2, with 15.7× smaller workload.

Table: MESS instance comparison to SOTA baselines.

Baseline		Backbone	Search Targets		Results: MS-COCO [2]		
			Error	GFLOPs	mIoU	GFLOPs	Latency†
DRN	(i)	ResNet50	–Baseline–		59.02%	138.63	39.96ms
Ours	(ii)	ResNet50	min	≤ 1×	64.35%	113.65	37.53ms
Ours	(iii)	ResNet50	≤ 0.1%	min	58.91%	41.17	17.92ms
Ours	(iv)	ResNet50	≤ 1%	min	58.12%	34.53	15.11ms
DRN	(v)	MobileNetV2	–Baseline–		54.24%	8.78	67.04ms
Ours	(vi)	MobileNetV2	min	≤ 1×	57.49%	8.10	56.05ms
Ours	(vii)	MobileNetV2	≤ 0.1%	min	54.18%	4.05	40.97ms
Ours	(viii)	MobileNetV2	≤ 1%	min	53.24%	3.48	38.83ms

†Measured on Nvidia: GTX for ResNet50 and AGX for MobileNetV2 backbone.

Qualitatively, we show the progressive refinement of the segmentation as an input image progresses through consecutive early exits of a MESS network:

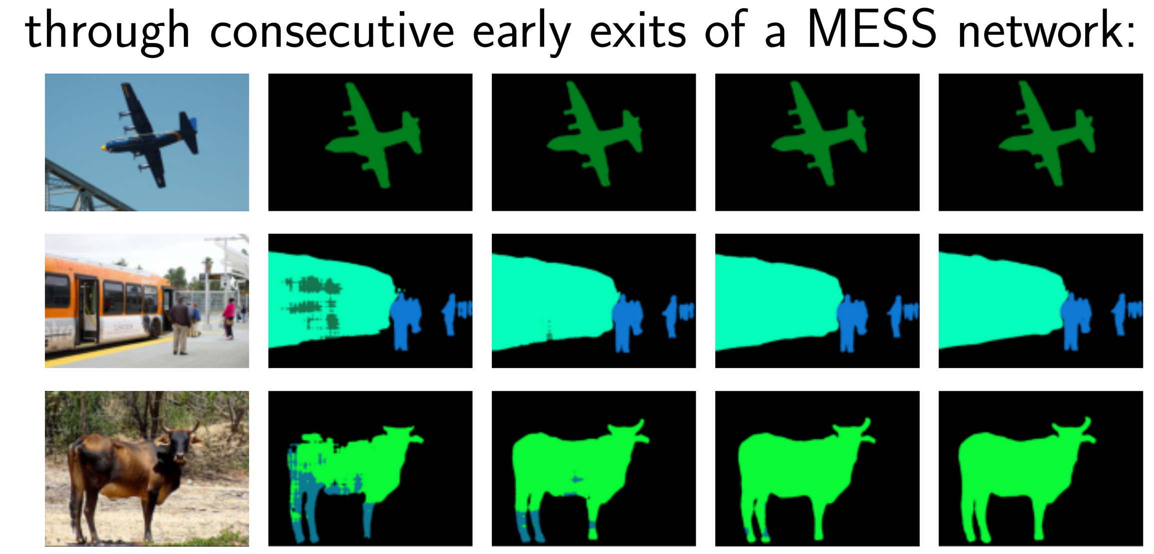

Figure: Progressive segmentation through MESS.

References

[1] S. Laskaridis, A. Kouris, and N.D. Lane. Adaptive inference through early-exit networks: Design, challenges and directions. In *EMDL*, 2021.

[2] T.Y. Lin et al. Microsoft COCO: Common Objects in Context. In *ECCV*, 2014.

[3] Fisher Yu et al. Dilated residual networks. In *CVPR*, 2017.