2025 IEEE International Conference on Machine Learning for Communication and Networking (ICMLCN) | 979-8-3315-2042-7/25/$31.00 ©2025 IEEE | DOI: 10.1109/ICMLCN64995.2025.11140234

Early-Exit Criteria for Edge Semantic Segmentation

Mateus S. Gilbert[1,2], Roberto G. Pacheco[3], Rodrigo S. Couto[1], Anne Fladenmuller[2]
Marcelo Dias de Amorim[2], Marcello L. R. de Campos[1], Miguel Elias M. Campista[1]

[1]Universidade Federal do Rio de Janeiro (UFRJ) – Rio de Janeiro, Brazil
[2]Sorbonne Université, CNRS, LIP6 – Paris, France
[3]Universidade Federal Fluminense (UFF) – Rio das Ostras, Brazil

***Abstract*—Early-exit deep neural networks (EE-DNNs) are multi-output DNNs designed for resource-constrained and latency-sensitive implementations, leveraging auxiliary output layers to partition processing between local, edge, and cloud devices. In the case of semantic segmentation, the literature lacks an efficient exit policy to stop the inference process earlier. Our contributions to fill this gap are twofold: (i) an adaptation of the normalized entropy (NE)-based exit criterion and (ii) a region-based exit criterion that compares segmentation from consecutive early exits. Our analyses reveal that the region-based approach better suits semantic segmentation because it exploits the intermediate early exits more efficiently. Our experiments show that a region-based EE-DNN delivers the same mean intersection over union as an EE-DNN with NE, saving at least 630 million floating-point operations per image on average.**

***Index Terms*—EE-DNN, edge/cloud, semantic segmentation.**

I. Introduction

Like most Computer Vision (CV) tasks, semantic segmentation, which consists in partitioning an image into semantically relevant regions, is increasingly dominated by Deep Neural Network (DNN)-based solutions [1, 2]. However, implementing DNN solutions can be difficult in resource-constrained and latency-sensitive networked scenarios. One common approach is to store these models on cloud servers, which can handle DNN demands, but communicating with the cloud can introduce significant latency and stress on the communication channel [3]. Alternative approaches such as TinyML try to enable local deployment in constrained devices by reducing DNN's implementation cost [4]. Still, these methods usually lower model complexity through parameter quantization and compression techniques, which can negatively impact performance [5].

This is where Early-Exit DNNs (EE-DNNs) come in, which enable deploying parts of the network in local, edge, and cloud devices. EE-DNNs offer a hierarchical inference framework, where early layers classify simple data and later layers take care of more complex data. The inference process can finish when the EE-DNN is confident in an output generated at an early exit. This setup enables DNN partitioning, where portions of the network computations can be performed in the edge device, restricting communications with the cloud to a few complex data.

A conventional early-exit criterion for determining when to leave an EE-DNN is to compute the Normalized Entropy (NE) of an output, under the assumption that an EE-DNN is confident in its generated output when NE is low [5]. However, adapting this exit criterion from image classification to semantic segmentation can lead to an inefficient exit policy. While NE-based exit criteria are $O(n)$ in image classification, it becomes $O(n^3)$ in semantic segmentation because it requires computing NE for each pixel. If in the former case this exit criterion is dependent only on the number of classes, in the latter the image size also impacts its complexity. To address these issues, we propose a region-based exit criterion that compares segmentation differences between outputs generated in consecutive early exits. We show how this exit criterion suits the particularities of semantic segmentation and present a first implementation using Variation of Information (VI) [6] as an exit metric. We show that with this exit metric we can implement a region-based exit criterion that is $O(n^2)$, as its complexity depends on either the image size or the number of classes, but not the two simultaneously.

We implemented EE-DNNs using NE- and region-based exit criteria, comparing their performances on the Pascal VOC dataset [7]. The results show that the region-based criterion exploits the intermediate early exits better than its NE counterpart – an EE-DNN equipped with VI delivers similar performance in terms of Mean Intersection over Union (mIoU) while requiring on average 680 million to 2.62 billion fewer Floating-point Operations (FLOPs). It means that fewer images reach the last EE-DNN exit.

In summary, the contributions of this paper are: (i) we adapt NE-based exit criteria from image classification to semantic segmentation and analyze its shortcomings, (ii) we propose region-based exit criteria for early-exit semantic segmentation and implement it using VI as exit metric, (iii) we compare the two exit criteria, showing the advantages and disadvantages of each, and (iv) we detail the fundamentals behind region-based exit criteria in semantic segmentation to help other implementations that might prefer other functions than VI as exit metric.

II. Early-Exit Semantic Segmentation

An EE-DNN is a multi-output DNN that modifies the traditional DNN with auxiliary outputs, named early exits, to perform inference attempts on early-stage data and minimize the use of the last DNN layers for most of the input data [5]. It leans on the fact that features from early layers are enough

979-8-3315-2042-7/25/$31.00 ©2025 IEEE

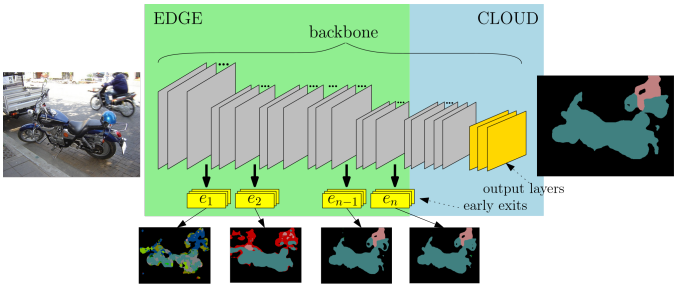


Figure 1. An EE-DNN for semantic segmentation showing the output data in early exits, and an illustration on how to distribute this DNN.

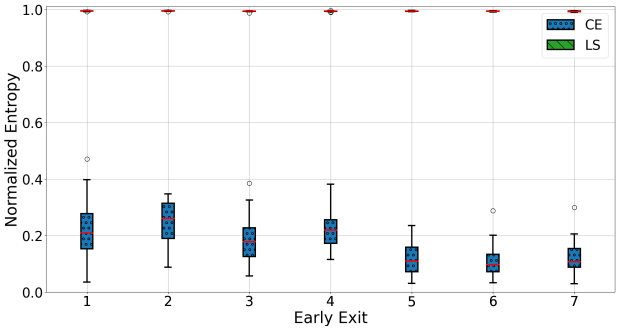


Figure 2. Average NE of EE-DNNs trained with CE and LS losses.

to infer simple data [8]. As shown in Figure 1, these early exits (e_i layers) branch out from a backbone DNN.

Thanks to the EE-DNN's multi-stage inference system, it becomes possible to place smaller Neural Networks (NNs) near the application's end devices [5, 9]. With this approach, we can balance the options of storing the whole resource-demanding backbone DNN locally or remotely (for example, in a cloud). In the latter case, transmitting data to be processed at the cloud can add significant communication delays and resource expenditure. By partitioning an EE-DNN at its early exits we can distribute the first stages in the edge, closer to end users, restricting the usage of the later resource-demanding layers, stored in the cloud, to challenging data.

EE-DNN for Semantic Segmentation.

Semantic segmentation partitions an image into regions and assigns semantic information (e.g., captions) to them. Compared to image classification, which assigns labels to a picture as a whole, semantic segmentation is concerned with pixel-level classification [1]. For example, given the leftmost picture in Figure 1 (motorcycle picture) as input, a DNN trained for semantic segmentation is expected to generate the rightmost picture as output rather than producing a set of descriptive labels.

In DNNs working with images, intermediate data tend to have a big dimensionality. Thus, it is common to design early exits that resort to strong dimensionality-reduction procedures to make them smaller and more adequate for the inference process [5]. However, these procedures are inadequate for semantic segmentation because they may destroy relevant features. Moreover, these high-dimensional data may lack some relevant low-resolution features because the first NN layers might not be capable of extracting them. Again, we can not resort to strong dimensionality reduction procedures to solve this problem, fearing the destruction of localized information [10].

As seen in Figure 1, the early exits of the EE-DNN produce segmented images with different degrees of quality. Ideally, the early exits' segmentations are sufficient for most samples [10]. Once trained, the early exits can either serve as a first attempt to conclude the inference process, much like with EE-DNNs designed for image classification, or function as an auxiliary exit for latency-sensitive applications awaiting the final output from the cloud [10].

III. Adapting NE to Semantic Segmentation

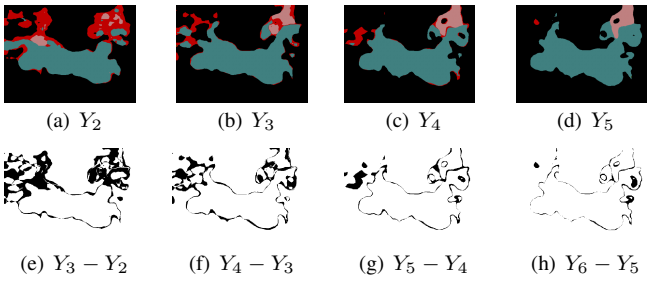
EE-DNN showed promising results in semantic segmentation [10], indicating how early exits can leverage edge support to reduce latency and resource expenditure with constant communication with the cloud. However, the proposed implementation lacked an efficient exit policy. Usually, in image classification, each early exit generates an output that is either a vector of probabilities, in which each entry corresponds to the probability of belonging to a given class or a vector of values that can be converted into probabilities. Because of this, a common exit strategy is computing the NE of this vector of probabilities and comparing it with a user-selected threshold [5]. For instance, supposing that Y_i is the output vector (with size $M \times N$) generated in an i-th early exit that classifies a set C of classes of interest with a threshold. τ_i, an EE-DNN's inference can occur when (we refer to NE as $\hat{H}$ to differentiate it from entropy):

$$\hat{H}[Y_i] = -\frac{1}{\log(\#C)} \sum_{c \in C} Y_{i_c} \log Y_{i_c} < \tau_i \qquad (1)$$

In image classification, a small entropy indicates that an early exit produced a good output as this usually occurs when one of the vector entries is significantly bigger than the others, reflecting the EE-DNN confidence in its prediction [5].

In semantic segmentation, each pixel gets its own vector of probabilities. So, a possible exit strategy is computing the NE for each pixel, taking the average from all pixels, and comparing it to the early exit's threshold. However, this approach does not scale well with image size and the size of C. In Equation 1, we can see that NE is $O(n)$, increasing linearly with the size of C. Thus, this exit strategy becomes $O(n^3)$ in semantic segmentation because we have to compute the NE for each pixel.

Preliminary experiments showed that another issue with resorting to NE in semantic segmentation lies in the training loss selection. Figure 2 presents the NE-based score of outputs from EE-DNNs trained with CE and LS losses [11], showing that the latter produces outputs with high entropy, with most having values close to 1, differently than the one that resorts to CE. From the characteristics of these loss functions, we can say that CE, a distribution-based loss [12], directly minimizes the entropy of EE-DNN's outputs, whereas LS, a region-based loss [12], penalizes segmentation mismatches without



(a) Y_2 (b) Y_3 (c) Y_4 (d) Y_5

(e) $Y_3 - Y_2$ (f) $Y_4 - Y_3$ (g) $Y_5 - Y_4$ (h) $Y_6 - Y_5$

Figure 3. Outputs of an EE-DNN and the differences (in black) between consecutive exits.

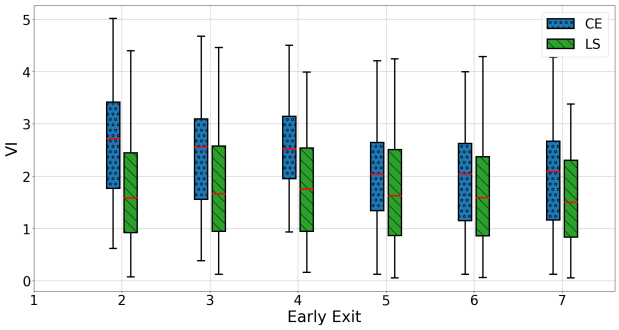


Figure 4. VI of EE-DNNs trained with CE and LS losses.

necessarily reducing entropy. Thus, employing an NE-based exit strategy can restrict our choice of loss functions, as it requires the output's entropy minimization.

IV. A Region-based Exit Criterion

The goal of semantic segmentation training, regardless of loss function choice, is to minimize the discrepancies between the DNN's segmentations and the ground truth. Hence, it's no coincidence the adoption of region-based losses in this CV problem, as they punish these regional mismatches. Using the motorcycle's images from Figure 1, Figure 3 shows how the differences between consecutive exits decrease towards later early exits. Generally, Y_i improves upon the previous exit's segmentation (Y_{i-1}). Hence, if both are close to the desired output, their differences are negligible, allowing the EE-DNN's to conclude inference. Compared to the previous exit criterion, this approach may involve the computations from one additional early exit. Still, it can offer better scalability and compatibility with a broader range of loss functions.

We present a first implementation of the proposed region-based exit criterion using VI [6] as the exit metric. To determine if an EE-DNN can stop its inference process at its i-th early exit, it computes

$$\mathrm{VI}(Y_{i-1}, Y_i) = H[Y_{i-1}|Y_i] + H[Y_i|Y_{i-1}] < \tau_i, \qquad (2)$$

$H[Y|X]$ being the conditional entropy of Y given X. We choose VI as the exit metric because it measures the information changes between two clusterings [6], which are the segmentations generated from consecutive early exits in the proposed exit criterion, and is guaranteed to be $O(n^2)$ regardless of the combination of image and C sizes (details in appendix A). Moreover, as shown in Figure 4, VI can work well with EE-DNNs trained on both distribution-based and region-based losses. Unlike NE in Figure 2, VI does not have a concentration of points around a single value. However, as seen in Figure 4, VI lacks the bounded $[0, 1]$ range of NE, with its maximum depending on the image's pixel count [6]. Still, this may not hinder finding an optimal τ_i, as early exits tend to yield VI values significantly smaller than this maximum.

V. Experimental Setup

We perform our experiments using `Pytorch`, and we make our code available at `GitHub` [13]. We train and test our models using the Pascal VOC 2012 semantic-segmentation dataset [7]. The dataset comprises $2,913$ coloured images with $6,929$ objects across 20 labelled classes, offering heterogeneity with categories like people, various animals, vehicles, and indoor objects. Originally, samples are split into two approximately equal-sized sets. We take one of these to construct our test and validation datasets, using 60% and 40% of the available images, respectively. Hence, we get a $50 : 20 : 30$ division of samples for training, validation, and testing.

We analyse EE-DNNs with seven early exits, approximately evenly spaced in terms of FLOPs, as these can provide a rough estimate of energy consumption and inference times [14]. We select a pre-trained DeepLabV3 [15] from the `torchvision` library as the backbone DNN to leverage its Atrous Spatial Pyramid Pooling (ASPP) [15] module for early exits [10]. This DeepLabV3 an mIoU of 0.71 on the test set and requires 60.60 billion FLOPs to process a 256×256 RGB image. Following Gilbert et al. [10], we trained the EE-DNNs for 250 epochs with batch sizes of 32, setting early exits' learning rates to 5×10^{-3} the backbone's to 1×10^{-4}). Additionally, we set a minimum learning rate for both to be equal to 2.5×10^{-5}.

A. Early Exit Configurations

As noted in Section II, early exits in EE-DNN for semantic segmentation avoid strong dimensionality-reduction procedures. This means that the number of computing units and layers that make up the early exit can impact the EE-DNN's FLOPs. The DeepLabV3's classifier that comes with `torchvision` [16] adopts convolutional layers with 256 output channels. Thus, we construct early exits that employ classifiers with 64 output channels in order to reduce EE-DNN's implementation costs. For example, if we pass a tensor with 10×10 pixels and 10 channels (*i.e.*, a $10 \times 10 \times 10$ tensor) through these early exits, they would perform approximately 7.8 million FLOPs, whereas they would perform 100.1 million FLOPs if they had the same configuration as the DeepLabV3's classifier.

B. Exit Criteria Evaluation

We compare the two exit criteria, assessing the trade-off between segmentation performance and computational cost by computing mIoU and FLOPs. We assess this by varying their exit thresholds in a range where the extremes reflect instances when leaving at an early exit is easy (which favours latency minimization and energy conservation over the application's

performance) and instances in which the opposite occurs. Under this premise, we have opted to set the same threshold to all early exits. Based on the results in Section IV and early experiments, we varied the VI threshold from 1.5 to 0.1 and the NE threshold from 0.15 to 0.01, with higher values reflecting scenarios where exiting the EE-DNN is easy. For NE, the threshold indicates confidence in outputs with low entropy being close to the desired results [5], whereas for VI's threshold controls the tolerable differences between consecutive segmentations to stop the inference process. We also compute the amount of images that leave each early exit to identify underutilized exits.

VI. Experimental Results

Otherwise stated, all mIoU and FLOP results are computed using 256×256 sized RGB images. Following the training configuration discussed previously, Table I brings the mIoU performance of each early exit trained with CE and LS loss functions. These results show that early exits close to the EE-DNN's input (before e_3) tend to have a better performance when trained with CE. This suggests that the EE-DNN with CE might have a slight advantage with higher threshold values that stimulate more inferences in the first early exits. After these exits, the performance becomes very similar, with the models trained with LS having a slightly better-performing early exits and CE having a slightly better output layer. Hence, there is no clear advantage in choosing one of the models based in mIoU performance alone.

Table I
mIoU of each EE-DNN trained with NE and LS losses.

Loss	e_1	e_2	e_3	e_4	e_5	e_6	e_7	out
CE	0.33	0.38	0.48	0.59	0.68	0.69	0.70	0.71
LS	0.20	0.32	0.49	0.59	0.68	0.69	0.70	0.71

In the upcoming experiments, to better distinguish these models, will refer to EE-DNN trained with LS and CE by a subscript. Additionally, when presenting model performance, mIoU performance is analysed in comparison with that of the original DNN. In this analysis, when we lower the threshold values, the rate between EE-DNN and original DNN increases because more images are segmented in later stages of the EE-DNN. Finally, as discussed previously, the model trained with LS is not suited for the NE-based exit criterion. EE-DNN$_{ls}$ with NE would send all images would be sent to the last output. Thus, its analysis is restricted solely to the region-based one.

A. Exit Criteria Impact on Computation Latency

Before comparing the performance of EE-DNNs equipped with the two exit criteria, Table II brings a comparison of the average (of $1,000$ repetitions) computation times for NE- and VI-based exit criteria on 256×256 in `Google Colab`. Whiles NE increases with the size of C, VI remains nearly unchanged. As discussed in Sections III and IV, the latter scales better with the combination of the image and C sizes than the former, which can lead to significant processing delays.

Table II
Average time (in Google Colab) to execute both exit criteria.

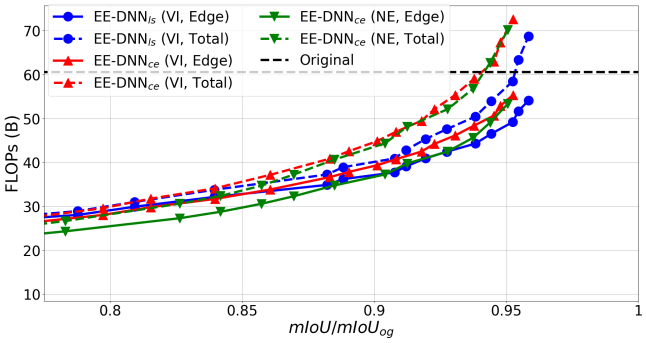
	Exit Criterion	
C	NE	VI
5	$5.62 \pm 1.48\ ms$	$5.89 \pm 1.83\ ms$
10	$11.3 \pm 2.64\ ms$	$6.71 \pm 1.72\ ms$
25	$32.6 \pm 8.37\ ms$	$6.99 \pm 1.21\ ms$
100	$122 \pm 30.4\ ms$	$8.73 \pm 1.68\ ms$

B. NE vs. VI

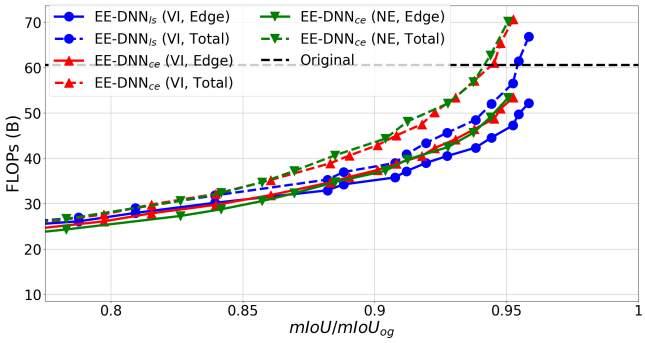
In Figure 5 we show the performance of the EE-DNN, comparing the average FLOPs performed at the DNN until it produces an inference. For lower threshold values, i.e., when we allow the EE-DNN performance to deviate more from that of the original DNN, the model that resorts to NE is capable of saving more FLOPs than the models that resort to VI. Given the regional-based exit criterion operability (discussed in Section 2), the first early exit capable of stopping the inference process is e_2, as e_1 job is to generate a reference segmentation to the following exit. Thus, it is no surprise in Figure 5(a) that, up to 0.85, EE-DNN$_{ce}$ shows the lowest FLOPs both on average and on the layers that are stored in local and edge devices. However, as we require that EE-DNN performance to deviate less from the original DNN, it is noticeable that EE-DNN$_{ls}$ equipped with the VI-based exit criterion starts to deliver a better performance in terms of FLOP, outperforming the model that resorts to NE on average from performances that obtained 0.88 and above, and on edge FLOPs from 0.94 onward. Additionally, if we focus on latency, assuming that the computations in the reference exit can be executed in parallel with the first functioning exit, EE-DNN$_{ls}$ starts outperforming EE-DNN$_{ce}$ with NE in both metrics from around 0.88 onward, as can be seen in Figure 5(b). Moreover, the average FLOPs EE-DNN$_{ls}$ starts to resemble the edge FLOPs of EE-DNN$_{ce}$ from around the same rate. Figure 5(c) complements these results, suggesting that the model that resorts to NE underused the early exits that come after e_5, with most images exiting EE-DNN$_{ce}$ in the last output, reason to why its FLOPs increases at a faster rate than the EE-DNNs that resort to VI. Additionally, this indicates that EE-DNN$_{ce}$ with NE transmits more data to the cloud, given that the last output is stored there. For instance, Figure 5(d) brings the distribution of images that were segmented in each early exits from models that obtained a performance rate close to 0.88, showing this tendency in depth.

VII. Related Work

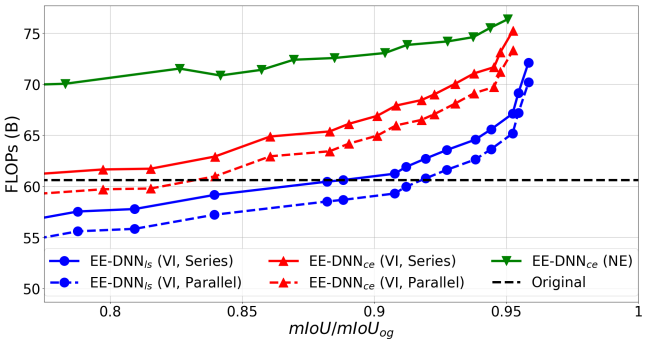
Teerapittayanon et al. [8] propose inserting additional exits to a multi-layered DNN in order to exploit primitive features from early DNN layers to enable fast and efficient image classification. They found that an EE-DNN can offer a hierarchical inference system in which early exits can be placed at local and edge devices, and the remainder of the DNN can be executed remotely at a cloud server. In this direction, Lo



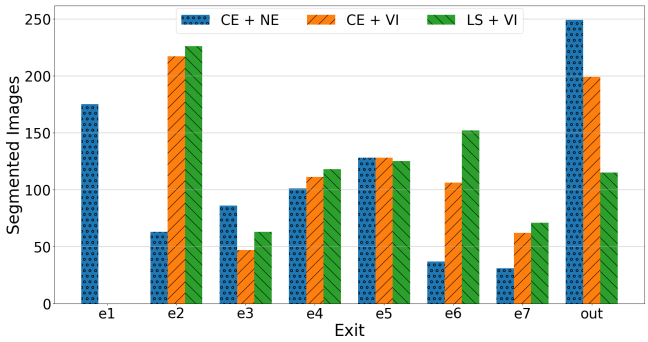
(a) FLOPs considering that first early exit is executed after reference exit.



(b) FLOPs considering that first early exit and reference exit are executed in parallel.



(c) FLOPs of images segmented after e_5.



(d) Distribution of segmented images of the models that obtained a performance rate around 0.88.

Figure 5. FLOPs count (on average) against model performance in comparison with original DNN of the lightweight EE-DNNs, using all early exits. In the graphs, $B = 10^9$ [17].

et al. [18] exploit this concept by designing a DNN with two exits that separately offer a lightweight DNN, which is placed on the edge, and another more complex model placed at a remote server (e.g., cloud) to deal with hard-to-classify data. In another work, Teerapittayanon et al. leverage the fact that an EE-DNN can be partitioned between local, edge and cloud devices to offer a hierarchical inference framework [19]. They restrict the usage of layers in higher instances to samples for which the EE-DNN is not confident in the outputs generated in an early exit. Similarly, Pacheco et al. [9] investigate how EE-DNNs can significantly reduce inference time, highlighting these models' latency advantages. However, all these findings were limited to image classification, making design choices less transferable to other CV problems. Our work addresses these problems in semantic segmentation.

Gilbert et al. proposed an EE-DNN for semantic segmentation [10], showing that early exits can produce good segmentation while requiring significantly fewer computation steps than if we traversed the entire DNN. Additionally, they discuss how their proposal can be helpful to latency-sensitive and resource-constrained implementations. However, their work does not address the problem of determining when to stop the inference process, something particularly important in edge-cloud co-inference because it determines whether an image should be transmitted to the cloud. We address this specific problem by investigating how to adapt NE, used in early-exit image classification, to semantic segmentation and by proposing a new region-based exit criterion.

VIII. Conclusion

EE-DNNs showed promising results in semantic segmentation, offering an edge-cloud co-inference setup that can reduce latency and resource cost by avoiding constant communication with a cloud server. However, it lacked something fundamental for the success of EE-DNNs deployment: an exit criterion to stop the inference process. To solve this problem, we start by adapting the existing NE-based exit criterion to the semantic segmentation scenario and identified some problems, such as poor scalability with image and classifiable class sizes, as well as the difficulty of implementing this exit criterion in models that were not trained to minimize entropy. To address these problems, we propose a region-based exit criterion that compares the differences between the segmentations generated between consecutive early exits and that allows images to leave a EE-DNN when differences are negligible. We present a first implementation of such criterion using VI [6] as a possible exit metric to enable it. The proposed VI-based exit criterion showed promising results, as it was able to work with models trained with different types of loss functions, scales better with image and class sizes, and was a component of the EE-DNN that delivered the best results in terms of FLOPs

and mIoU. In the future, it is interesting to investigate other exit metrics different from VI and exit criteria that take data transmission requirements into account, to make threshold adjustments dynamic according to network state.

ACKNOWLEDGMENT

This work was supported by CAPES (Finance Code 001), CNPq, FAPERJ (E-26/211.144/2019, E-26/202.689/2018), FAPESP (Grant 15/24494-8), Fundep (Rota 2030), and Bpifrance IE6 (France 2030 "Telecom 5G").

APPENDIX

A. Computational Complexity of VI

Consider two segmented images X and Y, with size $M \times N$ where each pixel can belong to one of C classes. A pseudo-code adaptation of the VI, equivalent to the one implemented in `scikit-image` module (used in our experiments), can be seen in Algorithm 1. To construct the contingency table P_{xy}, we need to compare each pair of pixels from X and Y, which is an $O(n^2)$ operation because it is dependent on the sizes M and N. Additionally, the resulting P_{xy} is a $C \times C$ matrix. Moving along the code, we have products between P_{xy} and diagonal matrices. Given the latter particularity, these operations are $O(n^2)$, dependent on the size of C. These operations generate $C \times C$ matrices, thus $xlog$ is also $O(n^2)$. Finally, the last computation to obtain the conditional entropies involves a matrix-vector product, which is $O(n^2)$. Thus, the Algorithm 1's complexity is $O(n^2)$.

Algorithm 1 VI between two segmented images

$P_{xy} \leftarrow$ contingency_table(X, Y)
$P_x \leftarrow P_{xy}.sum(axis = 1)$ ▷ Marginal probabilities
$P_y \leftarrow P_{xy}.sum(axis = 0)$
$P_x\_inv,\ P_y\_inv \leftarrow list(), list()$
for $c \in 0, ..., C-1$ **do**
  if $P_x[c] \neq 0$ **then**
    $P_x\_inv.append(1/P_x[c])$
  else
    $P_x\_inv.append(0)$
  end if
  if $P_y[c] \neq 0$ **then**
    $P_y\_inv.append(1/P_y[c])$
  else
    $P_y\_inv.append(0)$
  end if
end for
$P_x\_inv \leftarrow diag(P_x\_inv)$ ▷ Turn $1D$ vector
$P_y\_inv \leftarrow diag(P_y\_inv)$ ▷ into diagonal matrix
$H_{xy} \leftarrow -P_x \cdot xlogx(P_x\_inv \cdot P_{xy}).sum(axis = 1)$
$H_{yx} \leftarrow -xlogx(P_{xy} \cdot P_y\_inv).sum(axis = 0) \cdot P_y$
return $H_{xy} + H_{yx}$

function $xlogx$(X) ▷ X is an $M \times N$ matrix
  $Y \leftarrow 0_{M \times N}$
  for $m \in \{0, ..., M-1\}; n \in \{0, ..., N-1\}$ **do**
    if $X[m, n] \neq 0$ **then**
      $Y[m, n] \leftarrow X[m, n] \log_2(X[m, n]))$
    end if
  end for
  return aux
end function

REFERENCES

[1] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, P. Martinez-Gonzalez, and J. Garcia-Rodriguez, "A survey on deep learning techniques for image and video semantic segmentation," *Applied Soft Computing*, vol. 70, pp. 41–65, 2018.

[2] N. O'Mahony, S. Campbell, A. Carvalho, S. Harapanahalli, G. V. Hernandez, L. Krpalkova, D. Riordan, and J. Walsh, "Deep learning vs. traditional computer vision," in *Advances in Computer Vision: Proceedings of the 2019 Computer Vision Conference (CVC), Volume 1 1*. Springer, 2020, pp. 128–144.

[3] Y. Matsubara, M. Levorato, and F. Restuccia, "Split computing and early exiting for deep learning applications: Survey and research challenges," *ACM Computing Surveys*, vol. 55, no. 5, pp. 1–30, 2022.

[4] H. Han and J. Siebert, "Tinyml: A systematic review and synthesis of existing research," in *2022 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*. IEEE, 2022, pp. 269–274.

[5] S. Scardapane, M. Scarpiniti, E. Baccarelli, and A. Uncini, "Why should we add early exits to neural networks?" *Cognitive Computation*, vol. 12, no. 5, pp. 954–966, 2020.

[6] M. Meilă, "Comparing clusterings—an information based distance," *Journal of multivariate analysis*, vol. 98, no. 5, pp. 873–895, 2007.

[7] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results," http://host.robots.ox.ac.uk/pascal/VOC/voc2012/.

[8] S. Teerapittayanon, B. McDanel, and H.-T. Kung, "Branchynet: Fast inference via early exiting from deep neural networks," in *2016 23rd international conference on pattern recognition (ICPR)*. IEEE, 2016, pp. 2464–2469.

[9] R. G. Pacheco, K. Bochie, M. S. Gilbert, R. S. Couto, and M. E. M. Campista, "Towards edge computing using early-exit convolutional neural networks," *Information*, vol. 12, no. 10, p. 431, 2021.

[10] M. S. Gilbert, R. G. Pacheco, R. S. Couto, M. L. De Campos, and M. E. M. Campista, "Unlocking early-exiting semantic segmentation with branched networks," in *2023 IEEE Latin-American Conference on Communications (LATINCOM)*. IEEE, 2023, pp. 1–6.

[11] M. Berman, A. R. Triki, and M. B. Blaschko, "The lovász-softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4413–4421.

[12] J. Ma, J. Chen, M. Ng, R. Huang, Y. Li, C. Li, X. Yang, and A. L. Martel, "Loss odyssey in medical image segmentation," *Medical Image Analysis*, vol. 71, p. 102035, 2021.

[13] M. d. S. Gilbert, "EE-DNN for Semantic Segmentation," https://github.com/MateusGilbert/ee_semantic_segmentation, Dec. 2024.

[14] S. Laskaridis, S. I. Venieris, M. Almeida, I. Leontiadis, and N. D. Lane, "Spinn: synergistic progressive inference of neural networks over device and cloud," in *Proceedings of the 26th annual international conference on mobile computing and networking*, 2020, pp. 1–15.

[15] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," 2017.

[16] T. maintainers and contributors, "Torchvision: Pytorch's computer vision library," https://github.com/pytorch/vision, 2016.

[17] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*. PMLR, 2019, pp. 6105–6114.

[18] C. Lo, Y.-Y. Su, C.-Y. Lee, and S.-C. Chang, "A dynamic deep neural network design for efficient workload allocation in edge computing," in *2017 IEEE International Conference on Computer Design (ICCD)*. IEEE, 2017, pp. 273–280.

[19] S. Teerapittayanon, B. McDanel, and H.-T. Kung, "Distributed deep neural networks over the cloud, the edge and end devices," in *2017 IEEE 37th international conference on distributed computing systems (ICDCS)*. IEEE, 2017, pp. 328–339.