An investigation of knowledge distillation methods for underwater image semantic segmentation

Gabriel Guéganno* †, Ayoub Karine§, Thibault Napoléon†, Franck Florin* and Ayman Alfalou‡
* Thales DMS, 525 Route des Dolines, 06560 Valbonne, France.
† LabISEN, Vision-AD, ISEN Yncréa Ouest, 20 rue Cuirassé Bretagne, 29200 Brest, France.
‡ LabISEN, LSL, ISEN Yncréa Ouest, 33 Quater Chemin du Champ de Manoeuvre, 44470 Carquefou, France.
§ Université Paris Cité, LIPADE, F-75006 Paris, France.
gabriel.gueganno@isen-ouest.yncrea.fr

***Abstract*—Autonomous Underwater Vehicles (AUV) are key in seabed warfare applications. Seabed observation can be realized with an embedded camera processed with Artificial Intelligence (AI) that performs real-time semantic segmentation of the underwater images; this requires high performances and light-weight neural networks. However, AUV embedded computing capacity is limited and we cannot consider using a large network to obtain high performances. Knowledge distillation (KD) is a method for teaching a lightweight network with a large pre-trained supervisor network. This makes KD an ideal solution for training a network both highly efficient and capable of operating in real time embedded on a AUV. In this context, we evaluate different semantic segmentation network architectures, initially designed for urban datasets such as Cityscapes, over an underwater dataset: the semantic Segmentation of Underwater IMagery (SUIM) dataset. In addition, we evaluate the capacity of several KD methods to transfer knowledge in the underwater image domain with the SUIM dataset.**

Experiments demonstrate that the change of domain from urban scenes to underwater scenes achieves good results, both for semantic segmentation and KD.

***Index Terms*—knowledge distillation, semantic segmentation, autonomous underwater vehicle**

I. Introduction

The underwater domain holds a strategic importance for defense applications. In particular, the surveillance of the seabed is crucial for maintaining national security as it enables the detection of suspicious objects like mines or other threats. In this context, AUV play an important role in monitoring this environment, ensuring that security forces can respond swiftly and effectively to any underwater threat. However, those AUV need to be equipped correctly to move underwater without difficulty, and to detect threats.

Vision in underwater context can be performed using AI techniques applied to images/videos acquired by underwater acquisition systems. The sonar imagery approach can be very efficient as it can work from long distance to the seabed and with a good resolution [1]. Synthetic aperture sonar can be used to get high-resolution images of the sea bed. An alternative solution is to use an underwater optical imaging system to get to know the environment around the AUV [2]. Optical images in an underwater context have some drawbacks, specifically the effective distance, but they allow gathering real-time vision of the environment in front of the AUV, and some works have managed to improve underwater performances of optical devices [3].

With prior treatment, the AUV can use the video information to navigate. This treatment can be performed by deep learning with semantic segmentation algorithms. However, this raises a question about real time usability. Deep learning is sometimes not well suited to this application. From this perspective, it is of paramount importance to use methods of compression to obtain light-weight and real time neural networks to compute the semantic segmentation of the environment of the AUV. In addition, it goes together with the notion of ecodesign. A compressed network will require less energy per inference and it can therefore reduce the negative environmental impact of the AUV.

Semantic segmentation, which aims to predict a class to every pixel of an image, is an important topic in autonomous navigation. Autonomous driving and AUV navigation require indeed the knowledge of the environment to move correctly and semantic segmentation provides this type of information. Semantic segmentation has made great strides thanks to advances in Deep Learning, mainly in the urban and aerial domains [4], with databases such as ADE20K [5] or Cityscapes [6]. More occasionally, a few adaptations have also been made in the underwater domain [2], [7].

With the evolution of research in semantic segmentation, two families of neural networks have emerged: one based on the Fully Convolutional Network (FCN) architecture [8] and another based on the Vision Transformer (ViT) architecture [9]. Among the architectures that took the FCN as a base, some of them achieved very good performances such as PSPNet [10] or DeepLab [11]. On the side of the ViT-based architecture, we can cite the Segformer architecture [12] that achieved stunning performances on the semantic segmentation task. However, regardless of the architecture used, those good performances were obtained by wide architectures with many parameters. This limits the possibilities of using these architectures for real-time purposes. The computing capability of autonomous AUV is greatly limited, thus using the best performing semantic segmentation architectures is not an option. In addition, large neural network inference on images after being trained is energy consuming, and this must also be taken into account to embed a neural network in an AUV with limited energy

capacity.

However, one can surpass those limitations by various methods of neural network compression: pruning, quantization [13]. Those methods aim to reduce the size of the neural networks while limiting performance loss. A part of the research in this domain is dedicated to reducing the number of floating-point operations and the size of the models. One can do it either by designing light-weight models, by trying to remove useless parameters from wide architectures with network pruning or by using parameters quantization to reduce the size and the inference time of a neural network. These methods will be efficient in terms of computation time but often reduce the performance of the resulting compressed neural networks. Another field of research tries to overcome this issue: KD [14]. The main objective of KD is to train on a specific task a light-weight neural network, called the student network, under the supervision of a larger and more efficient neural network, called the teacher network. By doing so, one raises the performances of the student network without any augmentation of its size and number of parameters.

In this paper, we propose an investigation about the use of common semantic segmentation architectures and KD methods in an underwater context. To the best of our knowledge, KD were never exploited to train lightweight neural networks in an underwater context. Our investigation therefore focuses on the feasibility of using it in this domain. We focus on the following points:

- We evaluate the performances of three semantic segmentation architectures, designed for urban scene, on an underwater database SUIM [7]. With this part of the work we highlight the necessity to use KD for the training of light-weight neural network.
- We evaluate three KD methods, specifically, two conceived for FCN based architectures and one conceived for ViT based architecture.

In Part II, we present related work in the fields of semantic segmentation and KD. In Parts III and IV, we introduce some generalities of semantic segmentation and KD and describe the different methods considered for the investigation. Finally, we present and analyze the results obtained with these methods on the SUIM database in Part V.

II. Related Work

A. Semantic segmentation

Semantic segmentation is among the most important fields of computer vision with the constant development of autonomous driving and aerial surveillance. The principle behind semantic segmentation is the auto-encoder: an architecture composed of an encoder and a decoder. Among the most basic architectures of semantic segmentation, we can find the Fully Convolutional Network (FCN) [8] that only use convolutions, pooling and skip connections. PSPNet [10], which is a FCN, introduces the pyramid pooling that is a method to focus on the context by gathering information from different scales. The exclusive use of the convolution can cause problems related to receptive fields. When using conventional convolutions, the collection of information between two distant areas of an image can prove difficult due to the low receptive fields of the convolution. An answer to this issue is the atrous/dilated convolution used for example in DeepLab [11] and Dilation [15], which are both FCN. This type of convolution allows aggregating information from a wider zone without using spatial pooling. It has to be noted that all these methods are based on convolutional neural networks such as ResNet [16] or VGG [17] if one needs good performances, or MobileNet if one wants low latency computation.

Another way to tackle the receptive field issue is using a self-attention-based architecture, and specifically a ViT based architecture [9]. ViT uses the self-attention to gather information of the whole images given as input to compute the feature maps, even in the first layers of the network. ViT was originally designed for tasks as detection or classification, but recent works used this architecture for semantic segmentation. A solution adopted by the SEgmentation TRansformer (SETR) architecture [18] to use ViT for this task is to keep the encoder and add a simple decoder. Other architectures like Swin Transformer or Segformer [12] complexify the Transformer encoder by splitting it in several Transformer block. As for classical CNN encoders, each Transformer block computes a feature map with decreasing spatial size and increasing number of channels as one goes deeper in the encoder. To go further, both architectures propose several encoders with different sizes. Other methods propose using the Transformer as a decoder to compute the semantic segmentation. Mask2Former [19] for example uses a variation of the Transformer with masked self-attention to compute the semantic segmentation.

However, both convolution-based and Transformer-based architectures are often quite cumbersome. This implies the need to reduce their size so that they can be embedded in AUV.

B. Knowledge distillation

The basic concept of KD was introduced with the classification task by Hinton *et al.* [14], by comparing class probabilities of teacher and student models. The main objective of KD is to find a trade-off between the performance of the student model after the distillation and its size. According to Yang *et al.* [20], we can separate KD in three different types: response-based, feature-based and relation-based KD. Firstly, response-based KD corresponds to the distillation of information between the outputs of teacher and student models. Secondly, feature-based KD goes deeper in the networks by distilling knowledge directly between feature maps of teacher and student models. Finally, relation-based KD is a bit more complex as it distills knowledge between different samples of the training dataset or even different layers of the teacher and student models. With the success of KD to obtain performing lightweight classification models, this approach was also applied to the semantic segmentation task.

Among the first work done in the domain of FCN distillation, Xie *et al.* [21] propose to distill two types of knowledge

from the teacher network to the student networks simultaneously: first, the zero-order knowledge, the pixel probabilities, and secondly, the first-order knowledge, the sum of differences between a pixel and its eight neighbors. He *et al.* [22] use the knowledge from affinity maps to distill long-range information from the teacher's features to the student, it is feature-based KD. To tackle inconsistency caused by the difference in terms of shape between the teacher's and the student's features, a pre-trained auto-encoder is used to transform the teacher's features. Lui *et al.* [23] propose structural KD in two different ways. First, a pairwise distillation which aims to transfer pairwise pixel similarity to the student network, and secondly, a holistic distillation that uses a generative adversarial learning approach to align the segmentation map computed by the student network on the one of the teacher network. The Cross Image Relational Knowledge Distillation method (CIRKD) proposed by Yang *et al.* [24] is a relation-based KD that distillates two types of knowledge pixel-to-pixel information and pixel-to-region information. The distillation is performed either on mini-batches or on a memory bank composed of information extracted from the teacher network during the previous steps of the training. With the Channel-Spatial Knowledge Distillation method (CSKD), Karine *et al.* [25] used the similarity between the channels and the pixels of the intermediate feature maps to transfer knowledge from the teacher network to the student network.

As ViT-based architectures are more recent than FCN-based architectures, fewer methods have been implemented. Liu *et al.* propose with TransKD [26] a way to make profit of the specificities of the ViT architecture, and more specifically the Segformer [12] architecture. In the encoding part of the architecture, we find two types of information, the patch embeddings, that are fed into the different Transformer blocks, and the corresponding output feature maps. The distillation is performed between the patch embeddings and the feature maps of the student network and the teacher network.

III. Considered Semantic Segmentation Architectures

A. Principle

The objective of semantic segmentation is to assign to each pixel of an image a class prediction. The networks used for this task are generally composed of two parts. Firstly, an encoder that extracts from the input image a feature map $f \in \mathbb{R}^{d \times w \times h}$, where w, h and d denote respectively the height, the width and the number of channels of the feature map. Secondly, a decoder that computes a logit prediction $z \in \mathbb{R}^{C \times W \times H}$ from the feature map, where W and H denote the height and the width of the image and C denote the number of classes considered for the prediction. Finally, the semantic segmentation of the input image $y \in \mathbb{R}^{C \times W \times H}$ is obtained pixel by pixel by applying successively a softmax function, to get the probability distribution corresponding to the pixel, and an argmax function, to get the class prediction of the pixel.

For a classical semantic segmentation training, the loss used to compare each pixel of the logit prediction of the network with the corresponding ground truth is the cross-entropy measurement:

$$\mathcal{L}_{CE} = -\frac{1}{HW} \sum_{i}^{HW} CE\left(\sigma\left(z_i\right), y_i^{GT}\right) \tag{1}$$

Here, y_i^{GT} denotes the ground truth label of the i-th pixel of the image, z_i denotes the i-th pixel of the network logit prediction, σ denotes the softmax function and CE denotes the cross-entropy loss.

B. Considered methods

1) FCN: The first FCN-based semantic segmentation architecture considered is PSPNet. This semantic segmentation architecture capture global contextual information based on a pyramid pooling module. This module extracts features at multiple spatial scales, enabling the network to understand the overall structure of the scene while retaining the finest details. PSPNet combines these multi-scale features via a pyramid parsing module, which improves the representation of the overall context in the final feature map.

The second FCN-based semantic segmentation architecture considered in this investigation is DeepLabV3. This architecture uses atrous/dilated convolutions to extract features at different levels of resolution while maintaining high spatial resolution. DeepLabV3 also incorporates an improved version of the spatial pyramid pooling technique, called Atrous Spatial Pyramid Pooling (ASPP), to efficiently capture contextual information at multiple scales. By combining these features, DeepLabV3 improves segmentation performance, while mitigating the problem of loss of fine detail in segmentation objects.

DeepLabV3 and PSPNet have proven their effectiveness in the field of semantic segmentation by achieving state-of-the-art performances across various benchmark. Given their widespread adoption and established performance, these architectures provide a strong basis for further developments in knowledge distillation.

For both architectures, feature extraction is performed using a ResNet18, ResNet101 or MobileNetV2 convolutional neural network.

2) ViT: The ViT-based semantic segmentation architecture considered in this investigation is Segformer. Segformer is a semantic segmentation architecture that combines the strengths of FCN-based and ViT-based architectures. It uses a hierarchical structure with a multistage feature extraction process, where each stage applies a mix of convolutional layers and Transformer encoders. This design allows Segformer to efficiently capture both local and global contextual information. There are several versions of Segformer, depending on the size of the part dedicated to feature extraction. Unlike traditional ViT-based architectures, some versions of Segformer are lightweight and computationally efficient, making it suitable for real-time applications and knowledge distillation.

IV. Considered Knowledge Distillation Methods

A. Principle

When using KD to train a semantic segmentation network, two losses are generally considered. The first loss is the cross-entropy loss, presented in equation (1), used on the student network logit prediction. The second loss compares information between the teacher network and the student network. According to the method, this KD loss $\mathcal{L}_{KD}$ can be divided into several sub-losses, and we will see that it is the case for the different methods discussed in this paper. Generally speaking, the global loss $\mathcal{L}_{global}$ to train student can be formulated as follows:

$$\mathcal{L}_{global} = \mathcal{L}_{CE} + \lambda_{KD} \times \mathcal{L}_{KD} \tag{2}$$

Here, λ_{KD} denotes a coefficient that weights KD loss against the cross-entropy loss.

Inspired by Hinton *et al.*, a basic way to formulate the KD loss for semantic segmentation is to use the Kullback-Leibler divergence to compare the logit predictions of the teacher z^T and the student z^S. This loss can be formulated as follows:

$$\mathcal{L}_{KD} = \frac{1}{HW} \sum_{i}^{HW} KL\left(\sigma\left(\frac{z_i^S}{T}\right), \sigma\left(\frac{z_i^T}{T}\right)\right) \tag{3}$$

Here, KL denotes the Kullback-Leibler divergence and T denotes a temperature. With this basic loss function, the knowledge is distilled from the teacher output to the student output. Thus, KD is response based in this case.

B. Considered methods

In this section, the precise description of the different KD methods considered for this investigation has been placed in the appendices A, B and C.

1) Distillation from FCN: The first FCN-based method we consider for this investigation is CIRKD, proposed by Yang *et al.* [24]. This KD method is relation- and feature-based. It means that information from different images is used for the distillation during one step of the training, and particularly the feature maps extracted from the encoder of the semantic segmentation network are used. Three different distillations are conducted during the training of a student network with CIRKD, in addition to the classical KD, presented in equation (3).

The details of the method are presented in Appendix A. The complete distillation loss of the CIRKD method if formulated as follows:

$$\begin{aligned}\mathcal{L}_{CIRKD} =& \mathcal{L}_{CE} + \mathcal{L}_{KD} + \lambda_1 \mathcal{L}_{CIRKD}^{p2p-batch} \\ &+ \lambda_2 \mathcal{L}_{CIRKD}^{p2p-memory} + \lambda_3 \mathcal{L}_{CIRKD}^{p2r-memory}\end{aligned} \tag{4}$$

Here, $\mathcal{L}_{CIRKD}^{p2p-batch}$, $\mathcal{L}_{CIRKD}^{p2p-memory}$ and $\mathcal{L}_{CIRKD}^{p2r-memory}$ denote the three distillation losses proposed by the method. The losses $\mathcal{L}_{CE}$ and $\mathcal{L}_{KD}$ are the ones presented in equations (1) and (3) respectively. Finally, λ_1, λ_2 and λ_3 are weights set to 1, 0.1 and 0.1 respectively, as proposed in the original paper.

The second FCN-based method we consider for this investigation is CSKD, proposed by Karine *et al.* [25]. This method is feature- and response-based: information from the output of the teacher network and from intermediate calculations are used to distill information to the student. Two different distillations are conducted in addition to the classical KD, presented in equation (3). For each distillation one uses a dedicated module to catch information of feature maps, a Channel self-Attention Module (CAM) and a Position self-Attention Module (PAM).

The details of the method are presented in Appendix B. The complete distillation loss of the CSKD method if formulated as follows:

$$\begin{aligned}\mathcal{L}_{CSKD} =& \mathcal{L}_{CE} + \lambda_{KD} \mathcal{L}_{KD} \\ &+ \lambda_{PAM} \mathcal{L}_{CSKD}^{PAM} + \lambda_{CAM} \mathcal{L}_{CSKD}^{CAM}\end{aligned} \tag{5}$$

Here, $\mathcal{L}_{CSKD}^{PAM}$ and $\mathcal{L}_{CSKD}^{CAM}$ denote the two distillation losses proposed by the method. The losses $\mathcal{L}_{CE}$ and $\mathcal{L}_{KD}$ are the ones presented in equations (1) and (3) respectively. Finally, λ_{PAM} and λ_{CAM} are weights for the PAM and CAM losses set to 0.8 and 0.3 respectively, as proposed in the original paper.

2) Distillation from ViT: The ViT-based method we consider for this investigation is TransKD, proposed by Liu *et al.* [26]. This method is specific to the Segformer semantic segmentation architecture. As this architecture is based on the ViT architecture, two types of features can be found in the encoder: classical feature maps and patch embeddings. The method proposes one distillation for each type of feature. More specifically three variations of the patch embedding distillation are proposed.

The details of the method are presented in Appendix C. The complete distillation loss of the TransKD method if formulated as follows:

$$\mathcal{L}_{TransKD} = \mathcal{L}_{CE} + \mathcal{L}_{TransKD}^{PE} + \mathcal{L}_{TransKD}^{F} \tag{6}$$

Here, $\mathcal{L}_{TransKD}^{PE}$ and $\mathcal{L}_{TransKD}^{F}$ denote the two distillation losses proposed by the method. Depending on the variation of TransKD, $\mathcal{L}_{TransKD}^{PE}$ takes different forms. The loss $\mathcal{L}_{CE}$ is the one presented in equation (1).

V. Experimentation

A. Dataset

The underwater dataset we use for the investigation is SUIM [7]. This dataset is composed of 1635 underwater images annotated for semantic segmentation among 8 different classes. As proposed by the authors of the dataset, images are partitioned in two sub-datasets, 1525 are dedicated to the training phase and 110 are dedicated to the validation and testing phases. The size of images fluctuates from 375×590 to 960×1280 pixels, but most of the images ($\sim$80%) have a size of 480×640 pixels.

B. Experiments Details

1) Semantic Segmentation Architectures: For the different experiments, we use three semantic segmentation architectures. We use a DeepLabV3 and a PSPNet both with a ResNet101 encoder as a teacher network for the CIRKD

and the CSKD methods as they were designed for FCN architectures. The student networks used with these methods are also DeepLabV3 and PSPNet with different encoders, ResNet18 and MobileNetV2. For the experiments performed on the TransKD method, we use as a Teacher a Segformer with encoder MiT-B4 or MiT-B2 as TransKD is designed for this architecture. The student network is a Segformer with a MiT-B0 encoder.

2) Metric: The metric we use to evaluate the different trained networks is the mean intersection over union ($mIoU$). For a given class c, the IoU is formulated as follows:

$$IoU = \frac{|A \cap B|}{|A \cup B|} \tag{7}$$

Here, A denotes the area assigned to the class by the evaluated network on the prediction of an image and B denotes the area of the same class on the ground truth semantic segmentation of the image. To obtain the $mIoU$ we simply mean the IoU over all the classes.

3) Training Setup: The training setup is the same for all the experiments performed, with the exception of a few points or training for architectures based on FCN or ViT. In terms of data augmentation on the input images, we apply random horizontal flip, random scaling from 0.5 to 2 and crop of size 240×320. A stochastic gradient descent (SGD) with a momentum of 0.9 is used to update the parameters of the networks during training. The total number of iterations is 40000, and the trained network is tested on the validation dataset every 800 iterations. The batch size is set to 16, but experiments were also made with a batch size of 8. The learning rate is where there is a difference between FCN- and ViT-based architectures. For the FCN-based architectures, the initial learning rate is set to 0.02 while it is set to 0.0002 for the ViT-based architectures. In both cases the learning rate is updated at each iteration by multipling it by $(1 - \frac{iter}{Total_{iter}})^{0.9}$, where $iter$ denotes the current iteration and $Total_{iter}$ denotes the total number of iterations.

C. Semantic Segmentation Results

In this section, we compare the results of the different semantic segmentation architectures that we will use either as students or teacher in the following.

In Table I and Table II, we show the results on the dataset SUIM of the different FCN-based and ViT-based semantic segmentation architectures respectively. A first observation we can make is that regardless of the semantic segmentation architecture family, a logical trend can be observed. The more parameters a neural network has, the better its performance. The only exception is the DeepLabV3 associated with a MobileNetV2 encoder that has better performances than the PSPNet associated with a ResNet18 encoder although it has fewer parameters.

If we now compare the FCN- and ViT-based architectures, it clearly appears that the different Segformers outperform the convolutional networks. If we look first at the performances of the Segformer MiT-B0, it outperforms FCN-based architectures with a comparable number of parameters. If we look now at the performances of the two other Segformer architectures, they clearly outperform every other architectures. We particularly note the Segformer MiT-B2, which, with a reasonable number of parameters, achieves high performances.

TABLE I
RESULTS ON SUIM OF SEMANTIC SEGMENTATION ON THE DIFFERENT FCN-BASED ARCHITECTURES. R AND MN DENOTE RESNET AND MOBILENET ENCODERS RESPECTIVELY.

Method	Params (M)	mIoU (%)
PSPNet R101	68.1	69.50
DeepLabV3 R101	61.1	69.66
DeepLabV3 R18	13.6	67.72
PSPNet R18	12.9	64.60
DeepLabV3 MNV2	3.2	66.03

TABLE II
RESULTS ON SUIM OF SEMANTIC SEGMENTATION ON THE DIFFERENT VIT-BASED ARCHITECTURES.

Method	Params (M)	mIoU (%)
Segformer MiTB4	64.0	72.20
Segformer MiTB2	27.4	70.92
Segformer MiTB0	3.7	68.37

However, it is worth noting that for both FCN- and ViT-based architectures, there is a relatively large gap between the performances of the lightest and heaviest neural networks. This justifies the use of KD in this context, to improve the performances of the DeepLabV3 ResNet18, the DeepLabV3 MobileNetV2, the PSPNet ResNet18 and the Seformer MiT-B0.

D. Knowledge Distillation Results

1) Results: In this section, we analyze the results obtained on the SUIM database with the different KD methods presented in Part IV. In Table III and Table IV, we present the results of the different KD methods applied to the neural network studied in the previous part. The Table III regroups the results of the CIRKD and the CSKD methods applied to the FCN-based semantic segmentation neural networks DeepLbV3 and PSPNet, while Table IV regroups the results of the different TransKD methods applied to the ViT-based semantic segmentation neural network Segformer.

It is worth noting that the results obtained with those methods on the Cityscapes dataset are very good, for all pairs of teacher and student CIRKD, CSKD and TransKD methods allow improving the performances of the student.

For the FCN-based architectures, depending on the choice of the student and the teacher, the results can be significantly different. If we first look at the training performed with the DeepLabV3 ResNet101 as a teacher, using CSKD seems to be more pertinent associated with this teacher as every student trained with CSKD and supervised by it outperforms the same student trained with CIRKD. Moreover, for all students,

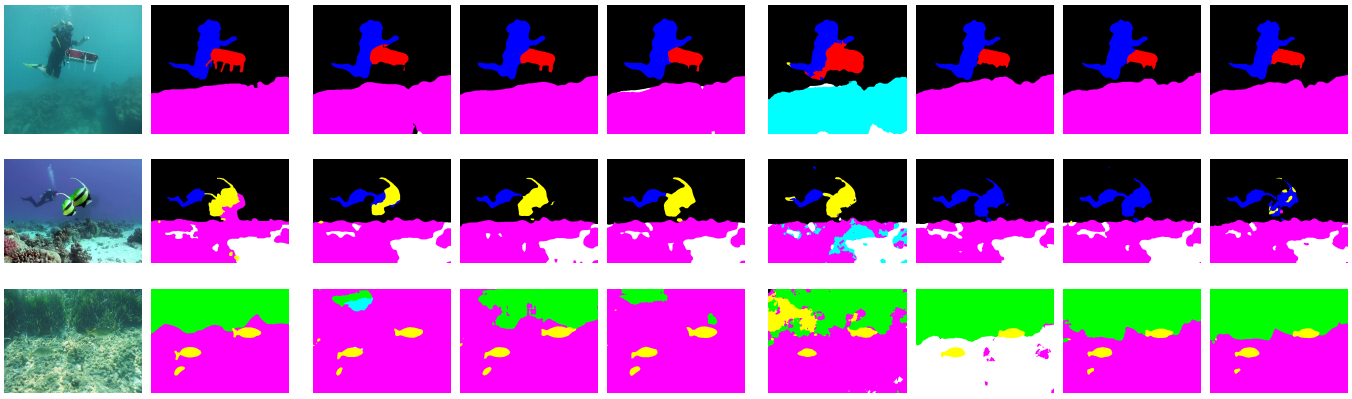

Fig. 1. Qualitative semantic segmentation results on the SUIM dataset. First two columns represent the considered image and the corresponding ground truth. Next three columns correspond to the results of CIRKD and CSKD methods using a student DeepLabV3 ResNet18. Last four columns correspond to the results of the three variations of TransKD methods using a student Segformer MiT-B0.

TABLE III
RESULTS ON SUIM OF KD ON THE DIFFERENT FCN-BASED ARCHITECTURES. THE LETTER T DENOTES THE TEACHER NETWORKS, AND THE LETTER S DENOTES THE STUDENT NETWORKS. R AND MN DENOTE RESNET AND MOBILENET ENCODERS RESPECTIVELY.

Method	Params (M)	mIoU (%)
T: DeepLabV3 R101	**61.1**	**69.66**
S: DeepLabV3 R18	13.6	67.72
+ CIRKD		65.73
+ CSKD		68.57
S: DeepLabV3 MNV2	3.2	66.03
+ CIRKD		64.66
+ CSKD		65.61
S: PSPNet R18	12.9	64.60
+ CIRKD		64.41
+ CSKD		66.86
T: PSPNet R101	**68.1**	**70.08**
S: DeepLabV3 R18	12.9	64.60
+ CIRKD		67.75
+ CSKD		66.70
S: PSPNet R18	13.6	67.72
+ CIRKD		69.98
+ CSKD		68.63

TABLE IV
RESULTS ON SUIM OF KD ON THE DIFFERENT VIT-BASED ARCHITECTURES. THE LETTER T DENOTES THE TEACHER NETWORKS, AND THE LETTER S DENOTES THE STUDENT NETWORKS.

Method	Params (M)	mIoU (%)
T: Segformer MiT-B2	**27.4**	**70.92**
S: Segformer MiT-B0	4.6	68.37
+ TransKD-Base		67.71
+ TransKD-EA		68.31
+ TransKD-GLMixer		68.58
T: Segformer MiT-B4	**64.0**	**72.20**
S: Segformer MiT-B0	4.6	68.37
+ TransKD-Base		62.15
+ TransKD-EA		62.77
+ TransKD-GLMixer		65.11

the performance obtained after training with CIRKD was lower than that obtained with conventional training without KD. For its part, with the exception of the DeepLabV3 MobileNetV2 student, all the training carried out with CSKD improved the performance of the student. In conclusion, the best performance is obtained from CSKD with a DeepLabV3 ResNet101 teacher. However, the best results are obtained when the same architecture is used as a teacher and student, in this case a DeepLab ResNet101 teacher and a DeepLab ResNet18 student.

Regarding to the training performed with the PSPNet ResNet101 as a teacher, the results are completely different. This time the CIRKD method allows obtaining better results than the CSKD method. In both cases, the performances of the student trained with a KD method are better than those obtained with a student trained without KD. In the same way as above, the student with the same architecture as the teacher obtains the better performances, in this case the PSPNet ResNet18 with the PSPNet ResNet101 teacher.

As for the FCN-based architectures, the result of KD on ViT-based architecture is very dependent on the choice of the student and the teacher. If we consider first the Segformer MiT-B2 teacher, the results are mixed. For all the TransKD variations, the performances of the student trained with KD are very close to the results of the student trained without, and only the TransKD-GLMixer allows to barely exceed it.

The different trainings performed with the Segformer MiT-B4 teacher are quite disappointing. None of the methods works with this choice of teachers. In all cases, the performance of the student trained with TransKD is far inferior to that of the student trained alone. The process of KD in this case doesn't transmit information, but rather loses it.

We show in fig. 1 some results of KD on the test dataset of SUIM. A first observation can be made is that for all images, the DeepLabV3 ResNet18 student seems to offer smoother segmentation than the MiT-B0 Segformer. The most

obvious example is the boundary between classes. For the DeepLabV3 ResNet18 they are clearly defined, even if they sometimes stray from the class boundaries of ground truth. On the contrary, for the Segformer MiT-B0, boundaries are noisy, even if the zones correspond to the right classes in relation to the ground truth. After the application of the different KD methods, we observe for all students an amelioration in the segmentation. For the DeepLabV3 ResNet18, the segmentation is more accurate and for the Segformer MiT-B0, boundaries are less noisy. If we compare the results of the students after KD, similar results can be observed for the first image. For the next two images, there is confusion for some classes for all students. For the second image, a class is incorrectly predicted by the Segformer MiT-B0 students trained with TransKD, whereas it was correctly predicted without knowledge distillation. For the last image, it is the DeepLabV3 ResNet18 students trained with CIRKD and CSKD that fail to correctly predict a class in the background. We conclude, however, that in all cases, there is still room for improvement to get closer to the ground truth.

2) Interpretation: The poor results of the CIRKD method in our investigation can be explained by the choice of the dataset. This method is based on a memory bank composed of pixel and region embeddings collected during the training. However, the SUIM dataset is very uneven in terms of class distribution. For example, the class "Robot" is very underrepresented in the dataset, thus its representation in the memory bank can be erroneous or incomplete and it can negatively impact the training for this class.

Other results can be explained by the difference between the number of parameters of both networks and by the size of the dataset. As the SUIM dataset is relatively small in comparison to datasets such as Cityscapes, the student has a limited diversity in the knowledge that is transferred from the teacher to learn the task of semantic segmentation. If in addition to that, the difference in terms of parameter is too high between the teacher and the student, it is harder for the student to reproduce the behavior of the teacher network and it leads to poor performances. It is the case for the Segformer MiT-B0 trained under the supervision of the Segformer MiT-B4, and the DeepLabV3 MobileNetV2 trained under the supervision of the DeepLab ResNet101.

However, it appears that KD can be effective in an underwater context. The CSKD method shows good performance, especially when the student and the teacher share the same architecture. For the ViT-based architecture, the results are mixed. If the networks trained without KD achieve good performances, the best we can obtain with KD is to equalize those performances. However, the KD in the field of ViT-based architectures is still in development and improvement is expected in the future. Another way of improving results could be to expand the underwater dataset, either by finding new images, which would involve both finding and annotating these images, or by improving the data augmentation process upstream of training.

VI. Conclusion

This paper presents an investigation of different KD methods for underwater image semantic segmentation. An alternative to KD would have been to reduce the size of large, high-performing networks using the pruning or quantization methods, mentioned in the introduction. We have not yet considered these methods, but it could be interesting to compare them with KD to compare the performance obtained, both in terms of network compression and semantic segmentation quality.

The KD methods considered in this investigation were CIRKD and CSKD for the FCN-based architectures and TransKD for the ViT-based architectures. The results of the investigation show that KD methods originally designed for urban semantic segmentation applications, and therefore not taking into account certain aspects such as water turbidity, can be effective in an underwater context. However some limitations reduce the effectiveness of KD such as the limited number of annotated images of underwater scenes. The difference in the number of parameters between the teacher and the student turned out to be a point of interest. Too great difference added to the limited amount of data cancels out the benefits of KD. The aim of this investigation was to evaluate the ability to compress neural networks for embedding in AUV. We investigated KD showing that it is possible to reduce the size of the network by a factor of 5 for the FCN-based architectures, while limiting the degration in performance.

As a perspective, we are planning to introduce a new KD method which takes into account the problems mentioned above. Another perspective would be to find a better set of parameters for the training. For this study, the choice of parameters, in particular the λ parameters used to weight the different losses, were the same as those proposed in the original papers of the methods used. One way of improving the performance of the KD methods used could be to carry out a grid search to optimize these parameters.

Acknowledgment

This research is financially supported by the National Association for Research and Technology (ANRT). All works were done in collaboration between Thales and ISEN Yncréa Ouest which are part of the GIS CORMORANT.

References

[1] A. Karine, N. Lasmar, A. Baussard, and M. El Hassouni, "Sonar image segmentation based on statistical modeling of wavelet subbands," in *2015 IEEE/ACS 12th International Conference of Computer Systems and Applications (AICCSA)*. IEEE, 2015, pp. 1–5.

[2] T. Le Pennec, M. Jridi, C. Dezan, A. Alfalou, and F. Florin, "Underwater exploration by AUV using deep neural network implemented on FPGA," in *Pattern Recognition and Tracking XXXI*, vol. 11400. SPIE, 2020, pp. 61–66.

[3] J. Hajjami, T. Napoléon, and A. Alfalou, "Adaptation of Koschmieder dehazing model for underwater marker detection," in *Pattern Recognition and Tracking XXXI*, vol. 11400. SPIE, 2020.

[4] A. Toker, M. Eisenberger, D. Cremers, and L. Leal-Taixé, "Satsynth: Augmenting image-mask pairs through diffusion models for aerial semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 27 695–27 705.

[5] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ADE20K dataset," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 633–641.

[6] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3213–3223.

[7] M. J. Islam, C. Edge, Y. Xiao, P. Luo, M. Mehtaz, C. Morse, S. S. Enan, and J. Sattar, "Semantic segmentation of underwater imagery: Dataset and benchmark," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 1769–1776.

[8] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[9] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2020.

[10] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2881–2890.

[11] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFS," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2017.

[12] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: simple and efficient design for semantic segmentation with transformers," *Advances in Neural Information Processing Systems*, vol. 34, pp. 12 077–12 090, 2021.

[13] T. Liang, J. Glossner, L. Wang, S. Shi, and X. Zhang, "Pruning and quantization for deep neural network acceleration: A survey," *Neurocomputing*, vol. 461, pp. 370–403, 2021.

[14] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[15] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[17] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[18] S. Zheng, J. Lu, H. Zhao, X. Zhu, Z. Luo, Y. Wang, Y. Fu, J. Feng, T. Xiang, P. H. S. Torr *et al.*, "Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 6881–6890.

[19] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention mask transformer for universal image segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 1290–1299.

[20] C. Yang, X. Yu, Z. An, and Y. Xu, "Categories of response-based, feature-based, and relation-based knowledge distillation," in *Advancements in Knowledge Distillation: Towards New Horizons of Intelligent Systems*. Springer, 2023, pp. 1–32.

[21] J. Xie, B. Shuai, J.-F. Hu, J. Lin, and W.-S. Zheng, "Improving fast segmentation with teacher-student learning," in *British Machine Vision Conference*, 2018, p. 205.

[22] T. He, C. Shen, Z. Tian, D. Gong, C. Sun, and Y. Yan, "Knowledge adaptation for efficient semantic segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 578–587.

[23] Y. Liu, K. Chen, C. Liu, Z. Qin, Z. Luo, and J. Wang, "Structured knowledge distillation for semantic segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 2604–2613.

[24] C. Yang, H. Zhou, Z. An, X. Jiang, Y. Xu, and Q. Zhang, "Cross-image relational knowledge distillation for semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 319–12 328.

[25] A. Karine, T. Napoléon, and M. Jridi, "Channel-spatial knowledge distillation for efficient semantic segmentation," *Pattern Recognition Letters*, 2024.

[26] R. Liu, K. Yang, A. Roitberg, J. Zhang, K. Peng, H. Liu, Y. Wang, and R. Stiefelhagen, "TransKD: transformer knowledge distillation for efficient semantic segmentation," *IEEE Transactions on Intelligent Transportation Systems*, 2024.

[27] S. Kornblith, M. Norouzi, H. Lee, and G. Hinton, "Similarity of neural network representations revisited," in *International conference on machine learning*. PMLR, 2019, pp. 3519–3529.

[28] P. Chen, S. Liu, H. Zhao, and J. Jia, "Distilling knowledge via knowledge review," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 5008–5017.

APPENDIX A
CIRKD

A. Mini-batch-based Pixel-to-Pixel Distillation

The first distillation is performed among a mini-batch of N images $\{x_n\}_{n=1}^N$. The objective is here to compute the similarity between intermediate features obtained from the images of the mini-batch for the student network and for the teacher network. Then the similarities of the student network are compared with the ones of the teacher network to obtain a loss measurement. For the mini-batch, one considers the spatially flattened intermediate feature maps on the network that are noted $\left\{f'_n \in \mathbb{R}^{d\times hw}\right\}_{n=1}^N$. For two images x_k and x_l of the mini-batch, with $k, l \in \{1, 2, \ldots, N\}$, the cross-image pairwise similarity matrix is $s_{kl} = f_k^T f_l$. Now we can compute the similarity between images of the mini-batch, a first loss can be formulated as follows:

$$\mathcal{L}_{CIRKD}^{p2p-batch} = \frac{1}{N^2}\sum_{k=1}^{N}\sum_{l=1}^{N}\mathcal{L}^{p2p}\left(s_{kl}^S, s_{kl}^T\right) \tag{8}$$

with:

$$\mathcal{L}^{p2p}\left(s_{kl}^S, s_{kl}^T\right) = \frac{1}{hw}\sum_{i=1}^{hw} KL\left(\sigma\left(\frac{s_{kl,i}^S}{\tau}\right), \sigma\left(\frac{s_{kl,i}^T}{\tau}\right)\right) \tag{9}$$

Here, τ denotes a temperature, s_{kl}^S and s_{kl}^T denote the similarity between the k-th and the l-th feature maps of the mini-batch for the student network and the teacher network respectively, and for both student and teacher networks, $s_{kl,i}$ denote the i-th row of the similarity matrix.

B. Memory-based Pixel-to-Pixel Distillation

Mini-batch size is often small, this is why a second distillation is performed using a memory bank. The memory bank takes the form of a class aware pixel queue $\mathcal{Q}_p$ updated with a *first-in-first-out* strategy. For a class, every corresponding pixel prediction is similar, thus it is not useful to fill the memory bank with every pixel corresponding to this class. In this context, at each iteration of the training, only a small number of pixel embeddings extracted from the teacher feature map are used to update the pixel queue.

With the same notations as before, for an image x_n of a given iteration, the flatten intermediates feature maps of the student and teacher networks are respectively $f_n^S \in \mathbb{R}^{d\times hw}$ and $f_n^T \in \mathbb{R}^{d\times hw}$. One forms a matrix $V_p \in \mathbb{R}^{K_p\times d}$ composed of K_p pixel embeddings sampled from the pixel queue $\mathcal{Q}_p$.

Then, for both student and teacher, the similarity matrices between the corresponding feature map and the matrix V_p are computed as follows:

$$p^S = \left(V_p f_n^S\right)^T \in \mathbb{R}^{hw \times K_p}, p^T = \left(V_p f_n^T\right)^T \in \mathbb{R}^{hw \times K_p} \quad (10)$$

Finally, the knowledge from the teacher is distilled to the student by using the Kullback-Leibler divergence once again between both similarities:

$$\mathcal{L}_{CIRKD}^{p2p-memory} = \frac{1}{hw} \sum_{i=1}^{hw} KL\left(\sigma\left(\frac{p_i^S}{\tau}\right), \sigma\left(\frac{p_i^T}{\tau}\right)\right) \quad (11)$$

Here, for both student and teacher network, p_i denote the i-th row of the similarity matrix.

C. Memory-based Pixel-to-Region Distillation

The last distillation is similar to the memory-based pixel-to-pixel distillation, except this time instead of using a queue composed of pixel embedding, a region queue $\mathcal{Q}_r \in \mathbb{R}^{C \times N_r \times d}$ composed of region embeddings is used, where N_r is the length of the queue for each class. The region queue is composed of C channels, one for each class, and each channel is filled with N_r region embedding of size d. Instead of updating the queue with several pixel embeddings at each iteration, $\mathcal{Q}_r$ is updated with the mean value of all pixel embeddings for each class.

Here again, for an image x_n of a given iteration, the spatially flatten intermediates feature maps of the student and teacher networks are respectively $f_n^S \in \mathbb{R}^{d \times hw}$ and $f_n^T \in \mathbb{R}^{d \times hw}$. One forms this time a matrix $V_r \in \mathbb{R}^{K_r \times d}$ composed of K_r pixel embeddings sampled from the region queue $\mathcal{Q}_r$. Then, for both student and teacher, the similarity between the corresponding feature map and the matrix V_r is computed as follows:

$$r^S = \left(V_r f_n^S\right)^T \in \mathbb{R}^{hw \times K_r}, r^T = \left(V_r f_n^T\right)^T \in \mathbb{R}^{hw \times K_r} \quad (12)$$

Finally, the knowledge from the teacher is distilled to the student by using the Kullback-Leibler divergence once again between both similarities:

$$\mathcal{L}_{CIRKD}^{p2r-memory} = \frac{1}{hw} \sum_{i=1}^{hw} KL\left(\sigma\left(\frac{r_i^S}{\tau}\right), \sigma\left(\frac{r_i^T}{\tau}\right)\right) \quad (13)$$

D. Complete Distillation Process

Finally, the complete distillation loss of the CIRKD method if formulated as follows:

$$\begin{aligned} \mathcal{L}_{CIRKD} = &\mathcal{L}_{CE} + \mathcal{L}_{KD} + \lambda_1 \mathcal{L}_{CIRKD}^{p2p-batch} \\ &+ \lambda_2 \mathcal{L}_{CIRKD}^{p2p-memory} + \lambda_3 \mathcal{L}_{CIRKD}^{p2r-memory} \end{aligned} \quad (14)$$

Here, λ_1, λ_2 and λ_3 are weights set to 1, 0.1 and 0.1 respectively.

APPENDIX B
CSKD

A. CAM

The objective of the CAM is to capture information between the channels of a semantic segmentation network output. For this, the module computes a matrix to weight the feature map and select the channel information. This operation is done for both student and teacher networks and the resulting matrix are used for the distillation.

Given a feature map $f \in \mathbb{R}^{d \times h \times w}$, it is flattened spatially to get $f' \in \mathbb{R}^{d \times hw}$. To obtain the relation between the i-th and the j-th channels of the feature map f', the channel attention weights $\omega_{ji}^{CAM} \in \mathbb{R}$ are computed as follows:

$$\omega_{ji}^{CAM} = \frac{exp\left(\frac{f_j' \cdot f_i'}{\tau}\right)}{\sum_{i=1}^{C} exp\left(\frac{f_j' \cdot f_i'}{\tau}\right)} \quad (15)$$

Here, f_j' and f_i' denote respectively the j-th and the i-th rows of the feature map f', and τ denotes a temperature. Once the channel attention weights ω_{ji} are computed, one can obtain the weighted feature $e^{CAM} \in \mathbb{R}^{d \times h \times w}$, channel by channel, as follows:

$$e_j^{CAM} = \beta \sum_{i=1}^{d} \left(\omega_{ji}^{CAM} f_i'\right) + f_j' \quad (16)$$

The resulting channel of the final feature map is the original feature map channel added to the weighted sum of all channels multiplied by a parameter β learned during the training.

B. PAM

The objective of the PAM is to capture spatial information between the pixels of a semantic segmentation network output. As for the CAM, the starting point is the logit output of the network f. With two additional convolution layers, two new feature maps $a, b \in \mathbb{R}^{d_r \times h \times w}$ are computed from f, with $d_r < d$, and they are flattened spatially to obtain feature maps $a, b \in \mathbb{R}^{d_r \times hw}$. Then similarly to the CAM, position attention weights $\omega_{ji}^{PAM} \in \mathbb{R}$ are computed as follows:

$$\omega_{ji}^{PAM} = \frac{exp\left(\frac{a_i \cdot b_j}{\tau}\right)}{\sum_{i=1}^{C} exp\left(\frac{a_i \cdot b_j}{\tau}\right)} \quad (17)$$

Here, a_i and b_j denote respectively the j-th and the i-th rows of the feature map a and b, and τ denotes a temperature, the same used in equation (15). For the next step one computes, with another convolution layer, a new feature $c \in \mathbb{R}^{d \times h \times w}$ and flatten it spatially to obtain $c \in \mathbb{R}^{d \times hw}$. Now the position attention weights ω_{ji} are computed, one can obtain the weighted feature $e^{PAM} \in \mathbb{R}^{d \times h \times w}$, position by position, as follows:

$$e_j^{PAM} = \alpha \sum_{i=1}^{hw} \left(\omega_{ji}^{PAM} c_i\right) + a_j \quad (18)$$

The resulting position embedding of the final feature map at the position j is the pixel at the position j in the feature map

a added to the weighted sum of all the pixels of the feature map c multiplied by a parameter α learned during the training.

C. Complete Distillation Process

The metric used to compare the feature maps obtained with CAM and PAM for the student network and the teacher network is the centered kernel alignment (CKA) [27]. If one notes $e^{-,S}$ and $e^{-,T}$ the feature obtained with CAM or PAM for the student network and the teacher network respectively, the CKA metric is defined as follows:

$$CKA\left(e^{-,S}, e^{-,T}\right) = \frac{\left\|(e^{-,T})^T e^{-,S}\right\|_F^2}{\left\|(e^{-,S})^T e^{-,S}\right\|_F \left\|(e^{-,T})^T e^{-,T}\right\|_F} \quad (19)$$

Here $\|\cdot\|_F$ denote the F-norm. this metric has values between 0 and 1. From here, two losses can be defined, one for CAM and the other for PAM:

$$\mathcal{L}_{CSKD}^{CAM} = -log\left(CKA\left(e^{CAM,S}, e^{CAM,T}\right)\right) \quad (20)$$

$$\mathcal{L}_{CSKD}^{PAM} = -log\left(CKA\left(e^{PAM,S}, e^{PAM,T}\right)\right) \quad (21)$$

Finally, the complete distillation loss of the CSKD method if formulated as follows:

$$\begin{aligned}\mathcal{L}_{CSKD} =& \mathcal{L}_{CE} + \lambda_{KD}\mathcal{L}_{KD} \\ &+ \lambda_{PAM}\mathcal{L}_{CSKD}^{PAM} + \lambda_{CAM}\mathcal{L}_{CSKD}^{CAM}\end{aligned} \quad (22)$$

Here, λ_{PAM} and λ_{CAM} are weights for the PAM and CAM losses set to 0.8 and 0.3 respectively.

APPENDIX C
TRANSKD

A. Patch Embedding Distillation

For the patch embedding distillation, the basic method is called TransKD-Base. To perform the distillation one uses a module, composed of dense layers, on the student network patch embeddings to match the size of the teacher patch embeddings. Then a mean square error (MSE) loss is used to compare the student and teacher patch embeddings as follows:

$$\mathcal{L}_{TransKD}^{PE-Base} = \sum_{i=1}^{4} \lambda_{PE,i} MSE\left(\mathcal{M}_{PE-Base}\left(e_i^S\right), e_i^T\right) \quad (23)$$

Here, $e_i^S \in \mathbb{R}^{N\times d_i^S}$ and $e_i^T \in \mathbb{R}^{N\times d_i^T}$ denote the patch embedding of the i-th Transformer block of the encoder for the student and the teacher respectively, with $i \in \{1,2,3,4\}$, N the number of patches used for the patch embedding and d_i^S and d_i^T the patch embedding size of the i-th Transformer block for the student and the teacher respectively. Then, λ_{PE} denotes a weight vector set to $[0.1, 0.1, 0.5, 1]$. Finally, $\mathcal{M}_{PE-Base}$ denotes the dense module used to resize the student patch embedding.

The two other variations, TransKD-EA and TransKD-GLMixer, use the previous method as a base. For TransKD-EA, an embedding assistant is used in addition to the dense layer module to obtain a more adapted version of the student patch embedding. For the TransKD-GLMixer method, only the last patch embedding distillation is modified. In addition to the dense module, the student patch embedding passes through another module composed of convolutions and attention computations to mix global and local information. In both cases, the patch embedding loss is also computed with the MSE.

B. Feature Map Distillation

The feature map distillation is common to all three variations of TransKD. Despite the fact that it is not commonly done by most of feature-based distillation methods, information is here distilled across different depths of the Segformer encoder. For a given Transformer block of the student encoder, one notes f_i^S the output feature map of this block, with i the index of the Transformer block. To be compared with the corresponding feature map f_i^T of the teacher encoder, f_i^S is mixed with the feature map computed by the next Transformer block with a module noted $\mathcal{M}_F$ to obtain a new feature map g_i^S. The operation is done as follows:

$$g_i^S = \mathcal{M}_F\left(f_i^S, g_{i+1}^S\right) \quad (24)$$

Then, g_i^S passes through a convolution layer to obtain the final feature maps $f_i^{final,S} = conv_{3\times3}\left(g_i^S\right)$. Finally, for each Transformer block, $f_i^{final,S}$ is compared to f_i^T using the hierarchical context loss (HCL) [28]:

$$\mathcal{L}_{TransKD}^{Feature} = \sum_{i=1}^{4} \lambda_{F,i} HCL\left(f_i^{final,S}, f_i^T\right) \quad (25)$$

Here, λ_F denotes a weight vector set to $[1, 1, 1, 1]$.

C. Complete Distillation Process

Finally, the complete distillation loss of the TransKD method if formulated as follows:

$$\mathcal{L}_{TransKD} = \mathcal{L}_{CE} + \mathcal{L}_{TransKD}^{PE} + \mathcal{L}_{TransKD}^{F} \quad (26)$$