IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 8 November 2024, accepted 8 December 2024, date of publication 12 December 2024, date of current version 3 January 2025.

Digital Object Identifier 10.1109/ACCESS.2024.3516519

RESEARCH ARTICLE

Fine-Tunned Segment Anything Model (SAM) for Reservoir Extractions Compared With Popular CNNs: An Experiment for Space-Borne Synthetic-Aperture Radar Images

NGUYEN HONG QUANG[1], HANNA LEE[1], EUI-MYOUNG KIM[2], AND GIHONG KIM[3]
[1]Institute for Smart Infrastructure, Gangneung–Wonju National University, Gangneung-si, Gangwon-do 25457, South Korea
[2]Department of Drone and GIS Engineering, Namseoul University, Cheonan 31020, South Korea
[3]Department of Civil and Environmental Engineering, Gangneung–Wonju National University, Gangneung-si, Gangwon-do 25457, South Korea

Corresponding author: Gihong Kim (ghkim@gwnu.ac.kr)

This work was supported in part by Satellite Information Application of Korea Aerospace Research Institute (KARI), and in part by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education under Grant 2021R1A6A1A03044326.

ABSTRACT The freshwater resource is invaluable and indispensable for any nation like the Republic of Korea. Recently, deep learning (DL), AI models have become more popular and applied frequently for surface water studies. The Segment Anything Model (SAM) has been developing sharply and takes an adaptable approach with the ability to perform zero-shot inference. Although, pre-trained SAM was trained with millions of images (a billion masks), applying it to remote sensing data reveals limitations of inaccurate results and unlabeled classes, particularly for the more complex and noise data from Synthetic-aperture Radar Images. Hence, we fine-tune the SAM model and other popular CNN models of YOLOv8, U-net$_{(ResNet50)}$, and DeepLab$_{(ResNet50,EfficientNet)}$ for lake semantic segmentation using multi-SAR RS datasets of Kompsat-5, ALOS-2, Sentinel-1, and a combination of the three datasets (data link) for model result comparisons. This study's accuracy assessment showed the SAM was the most precise model (accuracy overall ≈ 0.95) followed by the DeepLab$_{(ResNet50)}$, YOLOv8, U-net$_{(ResNet50)}$, and DeepLab$_{(EfficientNet)}$ model. Almost all models segmented highly accurate lake areas fitted well with ground-truth masks, nevertheless, the SAM (code link) presented the most well-performance model. The YOLO deals well with larger datasets and requires deeper trains to gain higher accuracy outputs. In addition, this research investigated the responses of each DL model to the SAR dataset proving a need for fine-tuning model results on SAR RS for better lake segmentations.

INDEX TERMS Fine-tunning, Republic of Korea, reservoir, SAM, space-borne SAR.

I. INTRODUCTION

The attention of authorities/decision markers and scientists has been drawn to freshwater issues due to their importance [1] in several aspects such as agricultural irrigation, electrical hydropower, climate regulation on the positive side and flooding potential [2], and environmental changes on the negative aspects. Dozens of methods developed for water bodies from space data with wide variations of accuracies and scales like thresholding, classification algorithms, object-based analysis, multispectral indices, and newer machine learning and deep learning approaches [3], [4] have assisted better water-related management and monitoring of this valuable resource. Besides, research scientists endeavor for better cutting-edge techniques to optimize water extractions from space data. Convolutional Neural Networks (CNNs) have demonstrated a promising method of robust, accurately identifying water bodies by learning from large training datasets [5].

The CNNs have indicated a powerful and common in deep learning and contributed to escalating applications in geosciences like water body extractions [2], integrating with

The associate editor coordinating the review of this manuscript and approving it for publication was Cheng Hu.

TABLE 1. Some underline work using SAM and remote sensing data for water extraction.

Model name	Topic	Best accuracy metrics	Used data
Panoptic perception model [27]	Using the SAM model to support a semi-automatic segmentation annotation system	Overall accuracy (OA): 0.69 (all objects)	FineGrip Dataset (2,649 remote sensing images)
Multiresolution Segmentation (MRS) and SAM [21]	Comparing two methods of image segmentation for the parcel, industry, roof, road, water and single tree	IoU of 0.63, Precision of 0.86, F1 of 0.77 for water segmentation of SAM	WorldView-3
SAM and Class Activation Maps (CAM) [22]	Accurate extraction of rooftop photovoltaic (solar panel)	SAM merged CAM: OA=0.851, IoU=0.593, precision = 0.679, F1=0.730	Aerial images with annotated solar photovoltaic arrays and installation metadata [28]
SAM [29]	Fine fine-tuning on aerial images for or multi-scale feature fusion and generating segmentation masks	mIoU@256=0.76, F1@256= 0.86, mIoU@512=0.77, F1 @512 =0.86	33 true aerial orthophoto images of the Vaihingen and Potsdam dataset
Multispectral Automated Transfer Technique (MATT) [24]	Transposing the SAM segmentation masks from RGB images for automatically segmenting and labeling multispectral imagery	The highest mAP RGB sensor of 0.84	Drone videos and images
FastSAM [30]	extracting visual representations (houses) in remote sensing scenes	OA= 0.99, mIoU= 0.92, F1= 0.95	LEVIR-CD dataset (Google Earth images), WHU-CD dataset (aerial benchmark dataset), CLCD dataset (GaoFen-2 satellite), and S2Looking dataset (VHRs collected by various satellites)
SAM [23]	Multi-object detection and segmentation (airplanes, ships, houses…)	$AP_{mask} = 0.68$, $AP^{50}_{mask} = 0.92$, $AP^{75}_{mask} = 0.75$	WHU building dataset, the NWPU VHR-10 dataset, and the SAR Ship Detection Dataset (SSDD) dataset
PerSAM and PerSAM-F [31]	Combining a text-prompt-derived general example with one-shot training to improve accuracy and underscore SAM's potential for deployment in remote sensing imagery and reducing the need for manual annotation	For Satellite Multiclass using PerSAM-F; IoU = 0.45, Pixel Acc. = 0.96	UAV, Airborne, and Satellite images
On water application SAM and other CNNs [26]	"Accurate segmentation of river water in close-range Remote Sensing (RS) images"	OA=0.96, IoU= 0.93, precision=0.98, Recall=0.94, Fs=0.96	Close-Range Remote Sensing Imagery (LuFI-RiverSnap.v1)
SAM [25]	Changes in flood extent, roughness coefficient and mountain streambed are estimated using remote sensing images	For roughness: (IoU) ranges from 0.55 for grass to 0.82 for shrubs/trees; for flood: IoU = 0.94	Terrestrial photos
Segment Any Stream (SAS) from fine-tuned SAM [32]	Mapping water extents in the Mackinaw watershed	IoU = 0.76	High-resolution aerial imagery

the Google AI platform [6], landslide detection [7], many fields of study, practical medical [8], [9], and object detections [10], [11]. The advantages and limitations of deep learning are frequently discussed. Although CNNs gain the ability to learn and extract meaningful features from low-level feature edges and texture to high-level concepts of object shape and semantic concepts as well as state-of-the-art performance to achieve various computer vision tasks like image classification, semantic segmentation [4], [12], and earth surface mapping [6], [13], challenges pose to uncertainty and error of model results working with complex airborne synthetic-aperture radar (SAR) images which have much noise (speckles) [14], ambiguous object edges and that cannot be avoided [15]. Even the water bodies are the most distinguishable objects in the SAR images due to low values of backscatter of radar beans on the unwavering water surfaces [16], current SAMGeo employed image encoder of Masked Autoencoder (MAE) pre-trained Vision Transformer (ViT) adopted for geospatial data has been released [17] remaining some limitations of unlabeled training [18], and unappealing results in remote sensing applications [19]. Hence, the number of studies in this domain is considered negligible [20].

A. RELATED WORK

This section highlights the latest previous studies that applied SAM, integration of SAM with other networks, or using SAM to support supplementation tasks of feature segmentation in geosciences and water studies (Table 1). This summary indicates plenty of the SAM's applications in nonidentical topics of parcels, roofs, roads, water, and trees [21] and solar panels [22], transportation of airplanes, and ships [23] e.g. in the remote sensing domain. Additionally, the abilities of the SAM to work with a wide range of data types including high-resolution remote sensing, aerial orthophoto, drone (also video) [24], and terrestrial imagery [25] are indicated. The aforementioned studies employed different metrics to evaluate the model performances with the most commonly assessed ratios being the OA and IoU and the accuracy varied from 0.6 to 0.9.

Besides the SAM, other popular CNNs such as U-net, DeepLad, and EfficientLab have been applied for close-range imagery for water body segmentation showing the outstanding performance of the SAM model compared to the rest-employed networks [26]. Yakiyama et al. deployed the SegNet to automatically segment river water gaining an accuracy above 0.9 using images captured by an RGB sensor. Recently, Pedro et al. trained the Mask R-CNN model for surface water mapping using the high-resolution images of PlanetScope showing the model's ability to mask the small ponds with Intersection over Union (IoU) from 0.62 to 0.95. The survey of Huang et al. [4] demonstrated an overview of related works of deep learning models for the semantic segmentation of remote sensing images (SSRSI). The survey additionally summarised the remote-sensed data sources used for the segmentation domain. It gave main research directions of supervised SSRSI, semi-and weakly supervised, unsupervised domain adaption (UDA), multi-modal data fusion, and a pre-trained model for SSRSI. This study also posed the main challenges of large-scale variation, class imbalance, large image size, and limited labeled data of remote sensing compared to natural images. You Look Only Once (YOLO) is a rapidly developing model for numerous applications in many study fields, however, the main uses are on natural images. That is why we compare the YOLO (v8) results with the SAM performances in this study.

B. INTRODUCTION TO THE USE OF SAM FOR WATER BODY SEGMENTATION

The Segment Anything Model (SAM) developed by Meta AI has been proven a groundbreaking method of image segmentation, working with a wide range of image datasets [33]. Although over 1 billion masks from 11 million photos are used to train the base SAM and gain capabilities on numerous tasks [34], the model's applications in geosciences are limited and results are often unsatisfactory due to the different special imaging properties of remote sensors compared to common RGB natural cameras [35]. Currently, there has been some work on using the SAM for generating segmentation results, however, it is limited to the absence of class information and the fragmentation and inaccuracy of the predicted boundaries [36]. As the model supports any purpose of segmenting objects, input prompts are essential to specify the modeling target. On the other hand, the SAM has abilities of zero-shot and text prompts (besides input (points) and box prompts) which seems to be improper for segmentations of remote sensing data since mostly the classified features must be specified and labeled properly. A new version of SAM for geospatial data called GeoSAM is adopted from pre-trained SAM using final Vision Transformer (ViT) but trained with 44 thousand complex masks published with 4 model sizes of ViT-tiny, l, b, and h offering its ability to work with remote sensing data with several options. Besides, there is an open-source Python package integrating the GeoSAM with popular Python libraries like Leafmap [37], rasterio [38], and geopandas [39] providing users with a friendly interface and minimal effort of coding [37].

C. WHY A FINE-TUNED SAM IS NEEDED

Based on the recommendation of [31] and [40] about fine-tuning the SAM model might enhance the proficiency of the model on remote sensing data, we have done a pre-test of the SAM model to extract the reservoir boundary of some lakes using high-resolution SAR images of Kompsat-5 (Korea Multi-Purpose Satellite-5) images and found unsatisfactory results shown in Figure 1. Since zero-shot and text prompts did not work for the Kompsat-5 images, we tested one-shot, five-shots (3 foreground, 2 background), and box prompts shown in columns 2,3 and 4 of Figure 1. The box prompt seemed to be the most appropriate prompting method, however, it still generated large uncertainty in all model sizes (b and h), except a fine segmentation (Fig.1x). In addition, using the SAM automatic semantic segmentation, objects are segmented without labels tagged to them [25] and in geosciences it makes limited sense. Although more than 1.1 billion masks trained the SAM [19] mostly from natural images, SAM does often generalize well with overhead images (remote sensing) but fails in some cases due to the unique characteristics of remote-sensed imagery [40] and the number of masks for training from SAR image remains unknown. Hence, the SAM-producing unsatisfactory result of water surface extractions on the SAR images is explainable due to the dissimilar characteristics between optical and SAR images.

D. THIS STUDY OUTLINES

we work toward addressing the challenges, achieving the following contributions:

1) We fine-tuned SAM using the collections of around 1400 SAR images and more than 2000 masks for surface water semantic segmentation for the first time. This could enhance the SAM's ability to mask the lake's boundary on SAR images while its capability is still limited (code link).
2) Extensive evaluation of other current popular CNNs (U-net$_{(\text{ResNet50})}$, DeepLab$_{(\text{ResNet50})}$, DeepLab$_{(\text{EfficientNet})}$ comparing with most curent developing models of YOLOv8 and SAM.
3) Collecting SAR data over Korean territory, processing the data, and extracting the masks and labels. This is the most time-consuming task but important as the models cannot be trained without input data.
4) Performing extensive modeling and analyses of the model outputs and evaluating the model performances. From cross-comparisons of the model and used datasets, we suggest the appropriate model and the best data for the aim of lake extraction and future study orientation.

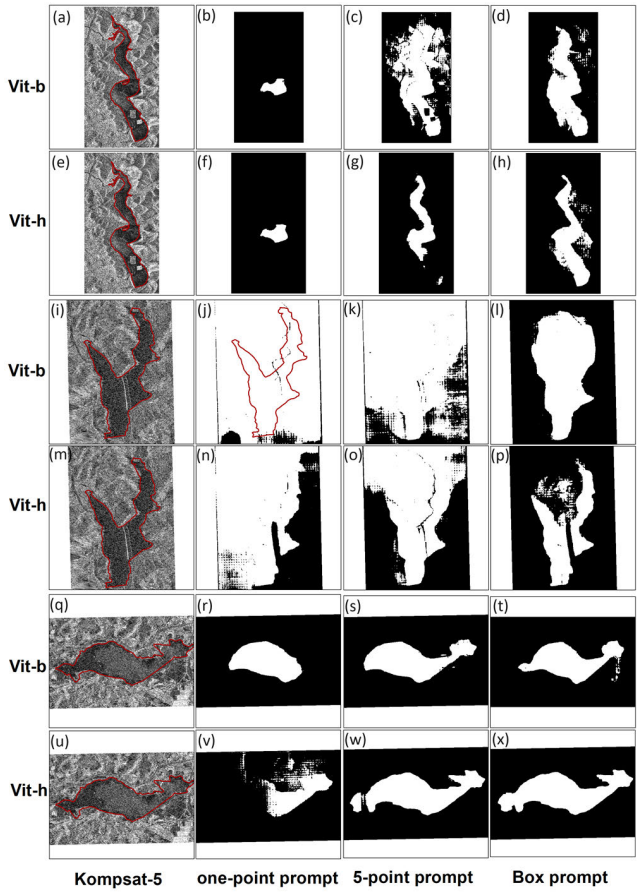

FIGURE 1. Pre-test for SAM performance on Kompsat-5 images.

II. REGIONS OF INTEREST AND DATA PREPARATION

A. KOREAN RESERVOIRS

Mountainous areas occupy a major proportion of Korean territory with a mixture of mountains, hills, and coastal plains [41]. The country has neam annual precipitation lower (1300 mm) in the north and higher on the southern coast (1800 mm) [42]. Due to this kind of terrain and climatic conditions, freshwater management has become more important to secure the indispensable drinking and irrigating water resource. Thus, thousands of reservoirs and ponds have been built, spreading in the entire Korea, however, they are located more in the South (Figure 1). Most of the lakes/reservoirs (90%) are small and medium sizes (storage capacity < 1 million m^3) and dozens of them are larger than a hundred hectares. 54% of the lakes were constructed 70 years ago [43]. The lake surfaces are changing conditions such as freezing in winter, vegetated (lotus, floating duckweed), and clearer in early spring and autumn. Therefore, monitoring the lake water status from space is a challenging task in South Korea. Although your experiment is on the South Korean regions as a case study, after fine-tuning the DL models, they can be applied to any other regions.

B. REMOTE SENSING DATA

Six scenes of level L1 Setinel-1A scanned at the Interferometric Wide swath (IW) mode covering the entire South Korean territory provided freely by the European Space Agency (ESA), 16 images processed at level 1D (provided by Korea Aerospace Research Institute (KARI)), and 59 ALOS/PALSAR-2 (ALOS-2) processed at level 2.2 provided without charges by Japan Aerospace Exploration Agency (JAXA) were collected for clipping sub-images (photos) for model input data preparation. Other information about the data resolution, and acquisition date are summarized in Table 2 and the image footprints are depicted in Fig. 2.

TABLE 2. Summary of SAR remote sensing data collected for lake/reservoir extractions over the South Korean territory.

Sensor	Number of scene	Processed level	Resolution (m)	Acquisition Date
Sentinel-1A	6	L1	10	2021.11.25 – 2023.12.02
Kompsat-5	16	1D	1.1 (ST) 6.25 (EW)	2020.02.11 – 2023.08.11
ALOS/ PALSAR-2 (ALOS-2)	59	2.2	7	2006.09.17 – 2011.02.20

C. METHODOLOGY

1) FRAMEWORK OF METHODOLOGY

The general working procedures of this study for lake boundary segmentation are depicted in Fig. 3 with three main parts: data preparation, model training, and model deployment (inferencing). In the first part,

$$D_{image} \text{ including } D_{train} = \{(I_{n\_train}, W_x, H_y)_{i=0}^{n\_train-1}, D_{val} = (I_{n\_val}, W_x, H_y)_{i=0}^{n\_val-1}, \text{ and } D_{test} = (I_{n\_test}, W_x, H_y)_{i=0}^{n\_test-1}\}$$

are created. Before all images used for training and validation are masked and labeled "lake", they are resized to make their size identical (D_{mask} including

$$D_{train\_mask} = \{(I_{n\_train\_mask}, W_x, H_y)_{i=0}^{n\_train\_mask-1}, \text{ and } D_{val\_mask} = (I_{n\_val\_mask}, W_x, H_y)_{i=0}^{n\_val\_mask-1}\}).$$

In the model fine-tunning phase, we developed the SAM and YOLOv8 semantic segmentation models in PyTorch (Appendix A) and the U-NET$_{(ResNet50)}$, DeepLabV3$_{(ResNet50)}$, DeepLabV3$_{(EfficientNet)}$ in TensorFlow framework (Appendix B). Additionally, the pre-trained SAM encoder is frozen to exploit the pre-trained parameters learned from the large dataset. The zero-shot prompt is given to eliminate the human interaction in the fine-tuned process. We applied the Adaptive moment estimation (Adam) to keep track of the exponentially decaying average of past gradients, learning rate, and bias correction in all the models (f_M). In the testing phase, the Adam is re-used to calculate the loss function

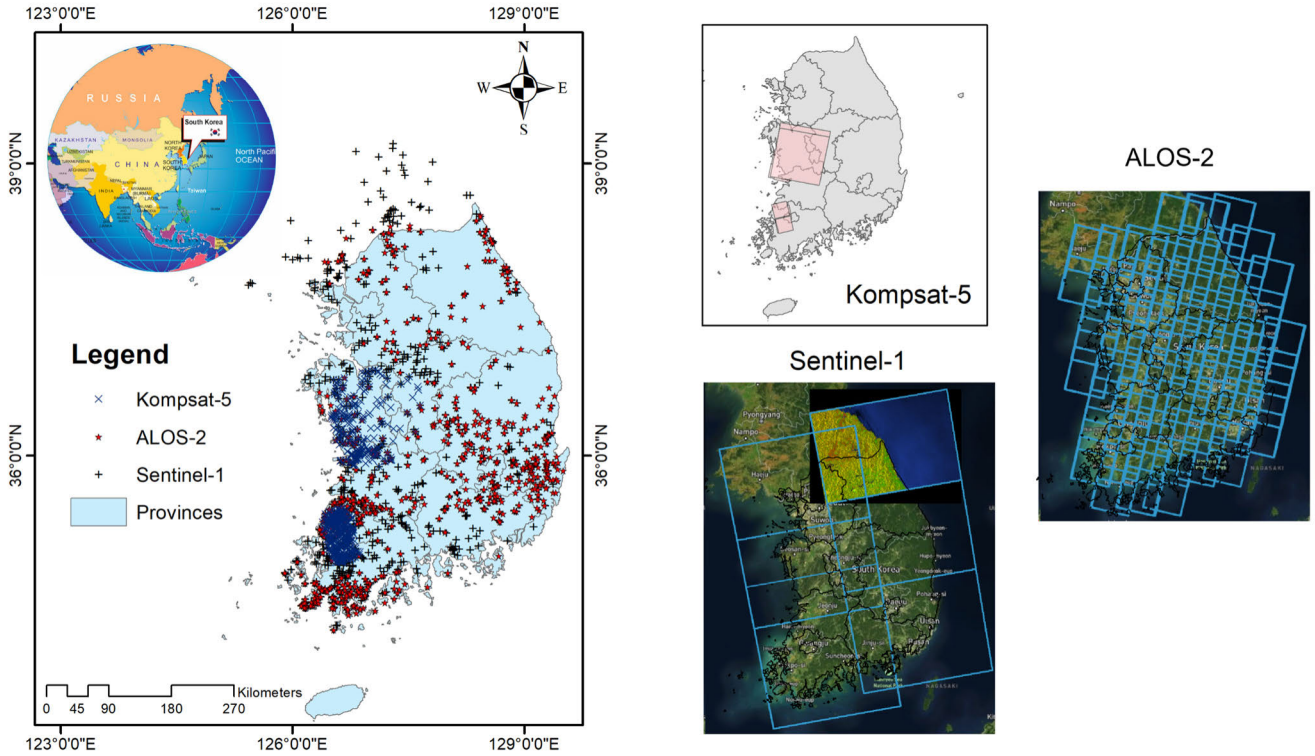

FIGURE 2. Maps of lake locations and acquired SAR remote sensing scene footprints for this study experiment.

values (L_V) using the validation dataset. We saved the best model checkpoint for the model inferencing stage. Finally, the saved model checkpoints are recalled to segment the lake boundaries and overlaid on them on the test images in D_{test} for visualizations.

To optimize the training strategy, we start to train all models for 50 epochs with a learning rate of 10^{-4}. After analyzing accuracy metrics, particularly the loss function, and segmentation results, we will decide which model must be trained deeper to gain higher accuracy. By doing that much training time will be saved.

2) DATA PROCESSING

As only the Sentinel-1 images were collected at the lower level of L1, we did image pre-processing including Orbit Correction, Thermal Noise and RGD Border Noise Removal, Radiometric Calibration, Speckle Filter (Lee Sigma), and Range Doppler Terrain Correction for enhancement of images quality. The data providers already pre-processed the ALOS/PALSAR-2 (ALOS-2) and Kompsat-5 images and we just converted the digital values to decibels (dB) for better visualization. Since the images are clipped in different sizes, we resized them to a width of 1024 and a height of 720 pixels. There is a large number of masking and annotation (labeling) tools. We elected the CVAT tool and masked the lake boundaries for our convenience. Although some automatic and semi-automatic masking tools are available,

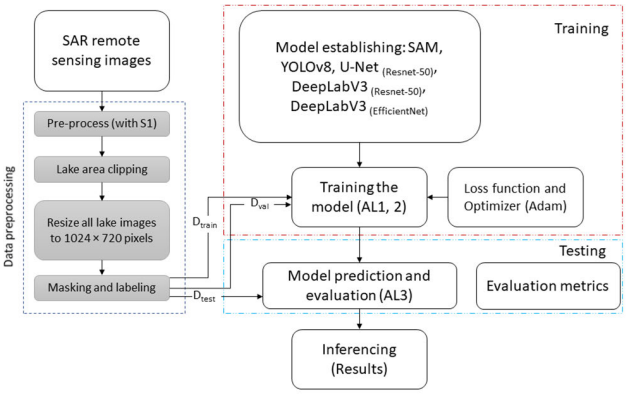

FIGURE 3. Workflow of lake surface semantic segmentation using Deep Learning (DL) models and SAR images, AL = algorithm, S1 = Sentinel-1.

we do it manually to use our maximum knowledge of drawing accurate lake boundaries for the models learned from rather than from other models. The outcome of this step is the $D_{train/val(image,mask)/test(image)}$, summarized in Table 3, ready for the model inputs.

3) SAM DESCRIPTION

A new Segment Anything Model (SAM) model has been introduced by Meta AI Research since 2023 [34] and is well-trained using a billion masks and around 11 million

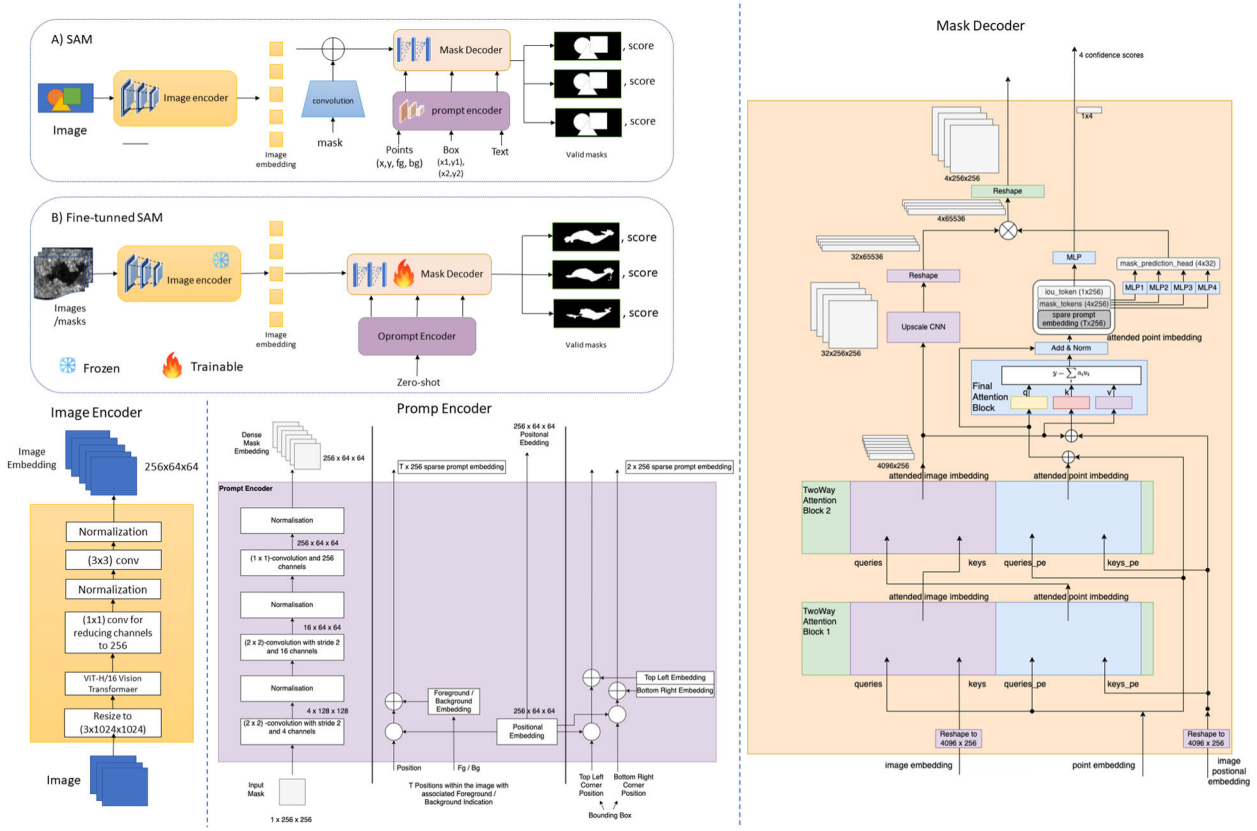

FIGURE 4. The architecture of A) SAM, adopted from [26], [34], B) our fine-tunned SAM.

TABLE 3. Statistics of datasets extracted from SAR remote sensing images.

Sensors	Number of images	Number of masks (lakes)	Train/validation
K5	323	602	258/65
AL2	553	625	442/111
S1	513	940	410/103
Total	1389	2167	1110/279

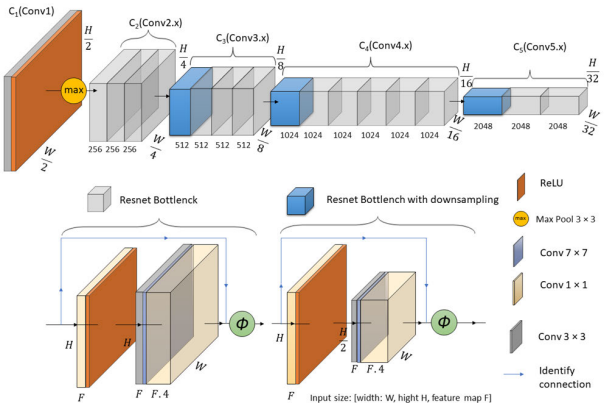

FIGURE 5. The ResNet-50 backbone for PyTorch implementation adopted from [26], [51].

licensed and privacy-respecting photos captured worldwide. Thus, its name is given by claiming its ability to segment

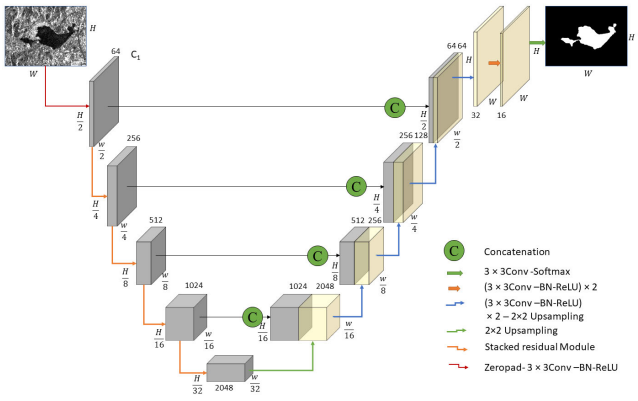

FIGURE 6. The architecture of U-net$_{(ResNet50)}$, modified from [26], [54].

any real objects without additional training [44]. The SAM includes three main components of image encoder, a flexible prompt encoder, and a mask decoder (Fig. 4A).

a: IMAGE ENCODER

An inputted image is computed once in the image encoder before applying a prompt for the model. In this step, the model uses a Vision Transformer (ViT) that undergoes pre-training through the application to exploit the changing variant of the masked autoencoders (MAE) technique. Formerly,

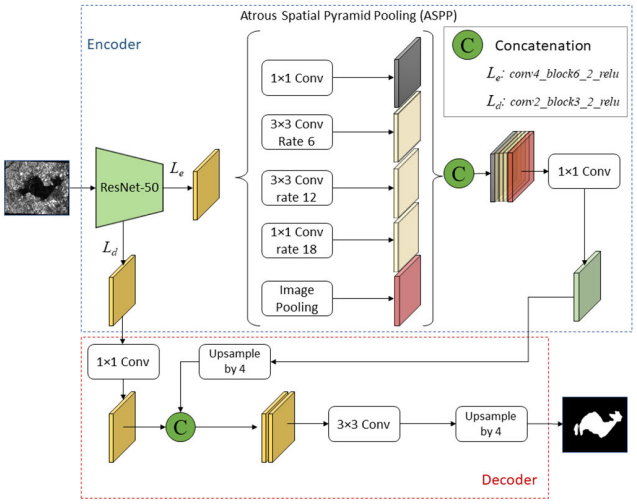

FIGURE 7. The architecture of DeepLab$_{(ResNet50)}$, adopted from [26], [55].

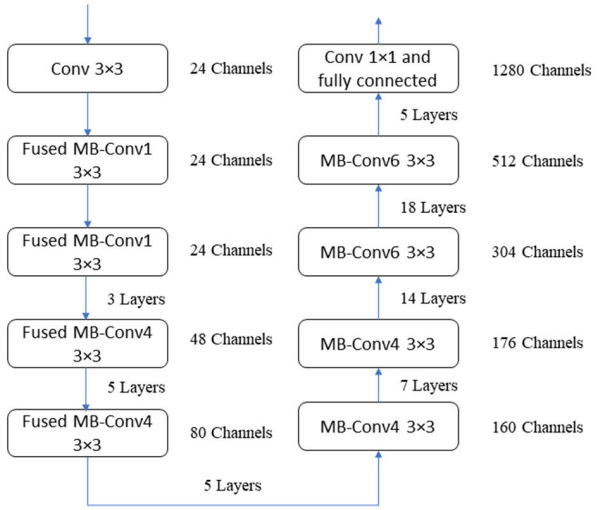

FIGURE 8. The EfficientNetV2 architecture, adopted from [57].

TABLE 4. Model size in respective of the number of parameters and mean training time per epoch in second.

Model	Num. parameters (millions)	Average training time per epoch (seconds)			
		Kompsat-5	ALOS-2	Sentinel-1	Combination
U-net$_{(ResNet50)}$	61	85	86	120	330
DeepLab$_{(ResNet50)}$	39	31	35	85	165
DeepLab$_{(EffcicientNet)}$	30	5	16	20	39
YOLOv8x*	72	8	9	11	19
SAM(vit-h)*	636	78	82	195	394

the ViT is adjusted slightly to handle high-resolution inputted datasets [45].

b: PROMPT ENCODER

Inputted images are embedded and passed through convolutions with prompt inputs to the decoder stage. Two desperate prompts are the dense prompt (masks) and sparse (SAM provides zero-shot prompts in case of automatic tasks, points, boxes, and text) prompts. The model encodes the tex-prompt using pre-existing text encoder from Radford et al. [46]. Besides, the dense prompts are embedded employing convolutions and summed together the image embedding element-wise.

TABLE 5. Accuracy metrics calculated for DL models' validations using the Kompsat-5 dataset.

Model	OA	IoU	Recall	Precision	Fs	K
U-net$_{(ResNet50)}$	0.870	0.796	0.837	0.844	0.838	0.810
DeepLab$_{(ResNet50)}$	0.903	0.836	0.888	0.869	0.877	0.849
DeepLab$_{(EffcicientNet)}$	0.861	0.794	0.845	0.828	0.830	0.807
YOLOv8x	0.769	0.706	0.748	0.743	0.743	0.717
SAM(vit-h)	0.921	0.847	0.877	0.895	0.882	0.859

TABLE 6. Accuracy metrics calculated for DL models' validations using the ALOS-2 dataset.

Model	*OA*	*IoU*	*Recall*	*Precision*	*Fs*	*K*
U-net$_{(ResNet50)}$	0.891	0.807	0.866	0.852	0.855	0.834
DeepLab$_{(ResNet50)}$	0.910	0.822	0.878	0.875	0.872	0.848
DeepLab$_{(EffcicientNet)}$	0.904	0.806	0.859	0.871	0.861	0.837
YOLOv8x	0.818	0.712	0.775	0.768	0.768	0.742
SAM(vit-h)	0.911	0.811	0.846	0.890	0.862	0.840

TABLE 7. Accuracy metrics calculated for DL models' validations using the Sentinel-1 dataset.

Model	*OA*	*IoU*	*Recall*	*Precision*	*Fs*	*K*
U-net$_{(ResNet50)}$	0.913	0.901	0.917	0.927	0.915	0.905
DeepLab$_{(ResNet50)}$	0.904	0.903	0.904	0.905	0.904	0.901
DeepLab$_{(EffcicientNet)}$	0.894	0.893	0.894	0.894	0.894	0.893
YOLOv8x	0.855	0.853	0.855	0.855	0.853	0.853
SAM(vit-h)	0.988	0.982	0.985	0.990	0.988	0.982

TABLE 8. Accuracy metrics calculated for DL models' validations using a combination of Kompsat-5, ALOS-2, and Sentinel-1 datasets.

Model	OA	IoU	Recall	Precision	Fs	K
U-net$_{(ResNet50)}$	0.888	0.780	0.785	0.785	0.833	0.842
DeepLab$_{(ResNet50)}$	0.969	0.888	0.948	0.928	0.938	0.908
DeepLab$_{(EffcicientNet)}$	0.955	0.875	0.925	0.925	0.920	0.885
YOLOv8x	0.955	0.870	0.900	0.962	0.925	0.889
SAM(vit-h)	0.971	0.893	0.925	0.944	0.930	0.906

c: MASK DECODER

The mask decoder effectively connects the image embedding and prompt embedding to generate output masks. The output token is inherited from a previous study by Dewi et al. [47] and employed an amended version of the Transformer decoder block from Vaswani et al. [48]. Finally, the mask generation is completed using a dynamic mask prediction head after the decoder block.

4) RESNET-50

The ResNet-50 model has proven its robust deep learning architecture with extreme efficacy in many computer vision applications. There are 50 layers built in the ResNet-50 architecture (Fig. 5) incorporating a bottleneck design

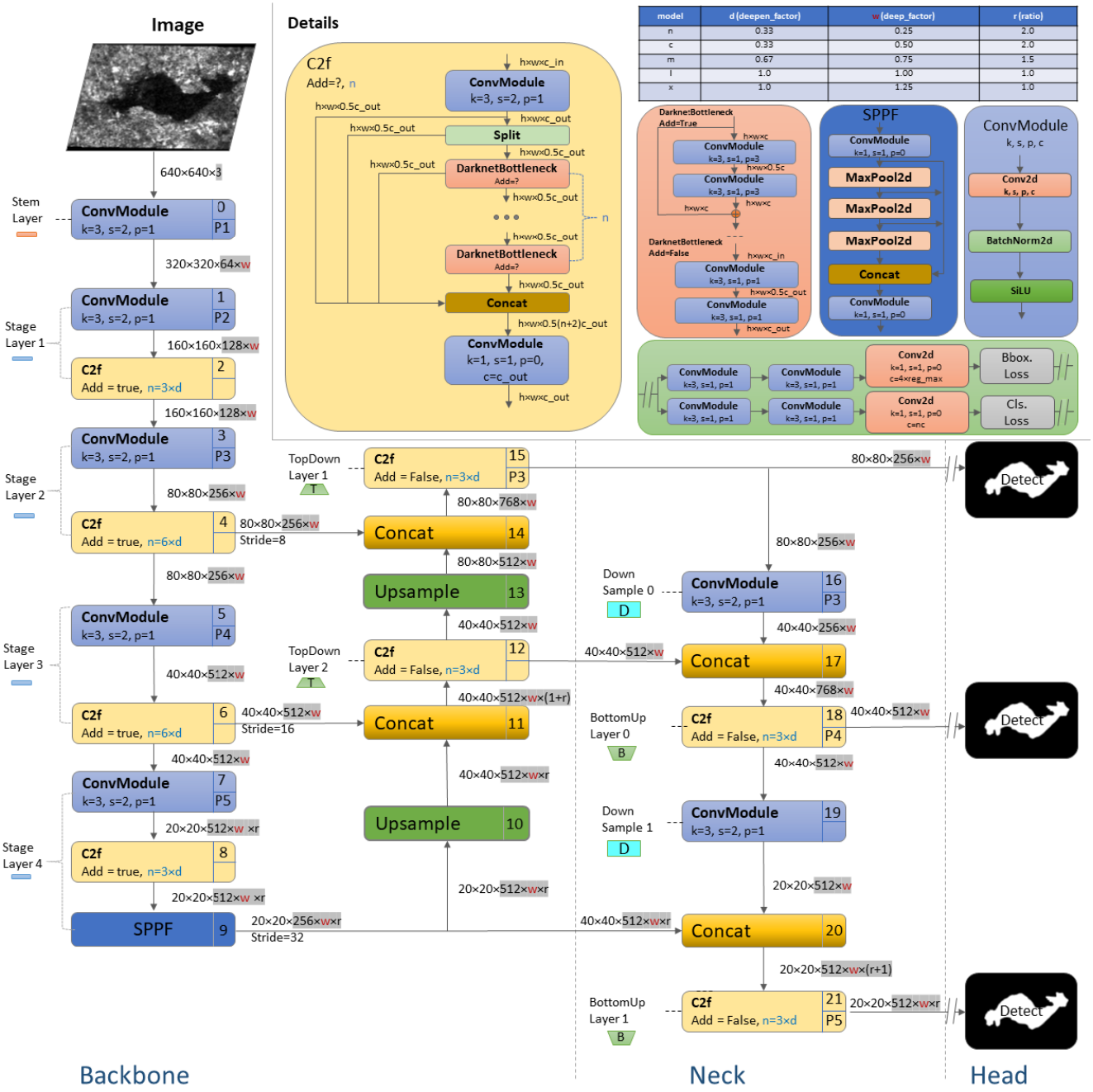

FIGURE 9. The YOLOv8 architecture with a modified CSPDarknet53 backbone (adopted from [59]).

for its base building blocks [49]. The model is well recognized for its exceptional capacity and efficient train design. The residual connections or facilitation of information propagation is the ResNet architecture incorporated skip connections. This is done throughout the network while mitigating the issue of disappearing gradients. Extensive photo datasets such as ImageNet have been used for pre-training and rendering the model for diverse image classification applications [50].

5) U-NET$_{(RESNET50)}$

U-Net is one of the well-known CNN models for its superior image segmentation performances of several types of images and datasets [52]. The U-net architecture including an encoder for downsampling, a decoder for upsampling, and feature fusion forms an U-shaped figure (Fig. 6) [53]. The ResNet50 used in this study serves as the encoder, meanwhile, a custom decoder presupposes the upsampling and convolutional layers. The integrating

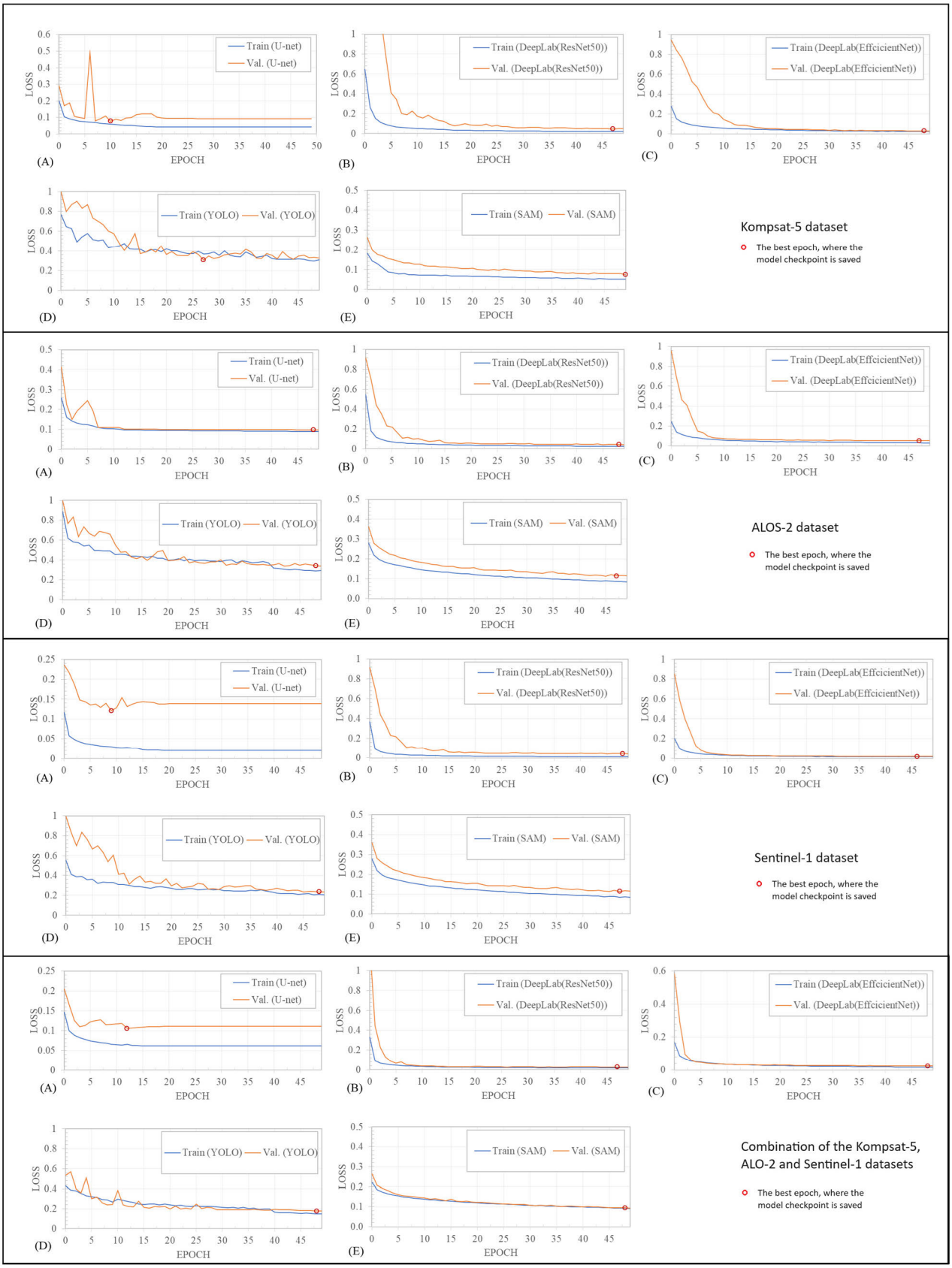

FIGURE 10. Training and validating (Val.) loss calculated for (A) U-net$_{(RetNet50)}$, (B) DeepLab$_{(RetNet50)}$, (C) DeepLab$_{(EfficientNet)}$, (D) YoloV8, and (E) SAM for reservoir segmentation with different SAR dataset.

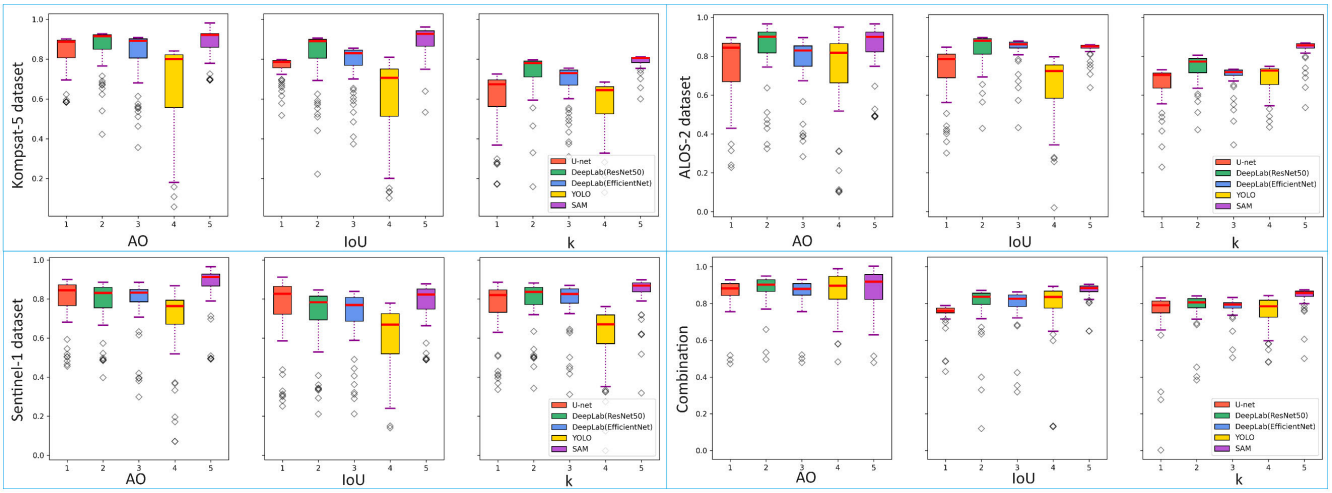

FIGURE 11. Model accuracy metrics of overall accuracy (OA), IoU, and K calculated in the validations for the SAR datasets.

TABLE 9. Summary of model performance quality for DL Lake semantic segmentation using the SAR remote sensing data.

Model	Accuracy		Visualization	Simplicity and Speed	Robustness and generality
	Overall	Individual dataset			
U-net$_{(ResNet50)}$k	****	***	****	***	***
DeepLab$_{(ResNet50)}$k	****’	****	****	****	****
DeepLab$_{(EfficientNet)}$k	***	***	****	*****	***’
YOLOv8x†	****	***	*****	***	*****
SAM(vit-h)†	*****	*****	*****	*	*****

One star indicates the worst performance and five stars indicate the best performance, ’ = plus 0.5, k and † refer to Keras-TensorFlow and Pytorch platforms (Speed is uncomparable between the two platforms), respectively. * YOLO has a low level of simplicity but it is fast.

features from different resolutions are assisted by skip connections [54].

6) DEEPLAB$_{(RESNET50)}$

DeepLab models featuring Atrous Spatial Pyramid Pooling (ASPP) and an encoder-decoder structure have been developed as a series and the DeepLabV3+ is a variant of this stream [55]. The ASPP performs pooling operations at different grid scales and the ResNet50 is integrated into the encoder pathway. The concatenation connects the 1×1 convolution and the upsampling in the decoder phase to generate the semantic segmentation masks (Fig. 7). In the upsampling procedure, the spatial resolution is enhanced by transposed convolutions and enriched by skip connections.

7) EFFICIENTNETV2 ARCHITECTURE

In 2021, Huang [56] proposed a family of CNNs including the EfficientNet and it did the scaling on the CNN like depth/wide and how depth/wide the network in the respective number of layers and the image resolution. In the EfficientNet, the dimensions of the network are scaled up (24 channels to 1280 channels for the M-size network) using a compound scaling approach [57]. The Mobile inverted Bottleneck Convolution (MB-Conv) as a baseline network and scale-up EfficientNet is employed in Fig. 8. As the EfficientNetV2 [58] has high performances, short training time, and training efficacy, it was selected to be integrated with the DeepLab in this study. The EfficientNetV2 is developed in small (S), medium (M), and large (L) sizes, Fig. 8 is a demonstration of the M-size network.

8) YOLOV8 ARCHITECTURE

You Only Look Once (YOLO) is a fast-growing model in the computer vision domain, introduced in 2015 [11], and currently (7/2024) V10 is proposed. Version 8 was released in January 2023 by Ultralytics [44] and built based on the foundations of YOLOv6 and YOLOv7. The YOLOv8 architecture inherits the backbone design from the formal version of 5 with changes in the CSPLayer and the C2f module (Fig. 9). Combining the contextual and high-level characteristics of cross-stage partial bottleneck with two convolutions in the C2f results in an improvement of accuracy and efficiency the YOLOv8 backbone [44]. Five model scales of YOLOv8n (nano), YOLOv8s (small), YOLOv8m (medium), YOLOv8l (large), and YOLOv8x (extra-large) with the same model size of 640 pixels are developed for options of uses. More complex models typically generate

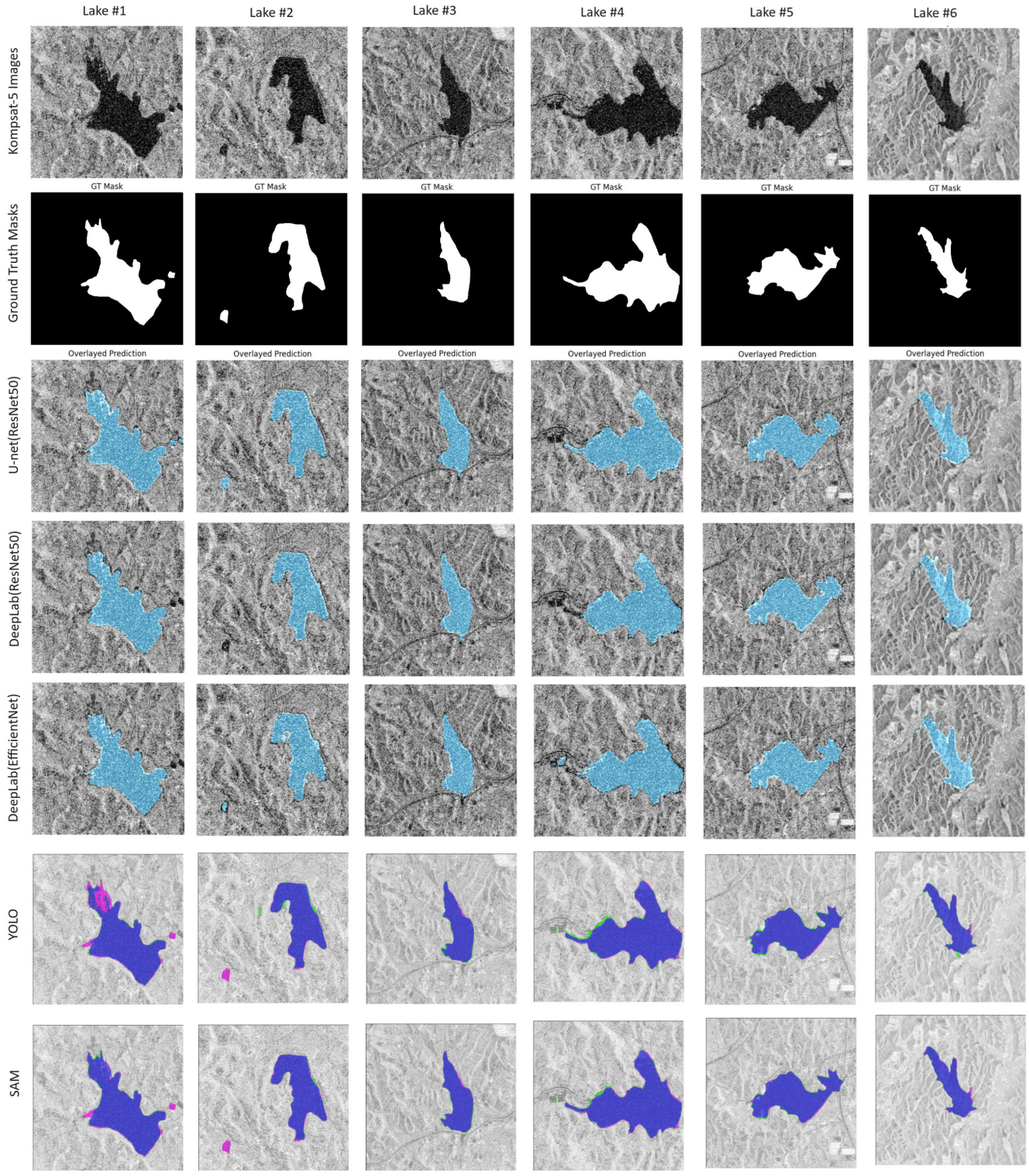

FIGURE 12. Deep learning model segmentation results on Kompsat-5 images, GT Mask is a short form of ground-truth mask.

higher accuracy of model outputs, however, correspondingly higher demands on computational speed and memory are required.

9) MODEL EVALUATION METRICS

For the model performance evaluation, we rely on common evaluated metrics which are frequently used in deep learning

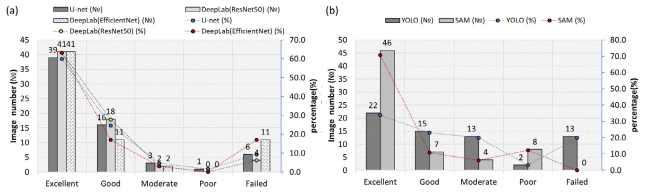

FIGURE 13. Results of model segmentation quantity and quality of the U-Net, DeepLab$_{(ResNet50)}$, DeepLab$_{(EfficientNet)}$ (a), YOLOv8x, and SAM(vit-h) (b), experiment on the Kompsat-5 dataset.

model evaluations calculated based on agreement between predicted and ground-truth masks. These metrics are the overall accuracy (OA), Intersection over Union (IoU), Recall, Precision, coefficient(Fs), and Kappa (k) solved by:

$$OA = 1 - \frac{TP + TN}{TP + FP + FN + TN} \times 100\% \quad (1)$$

$$IoU = \frac{TP}{TP + FP + FN} \times 100\% \quad (2)$$

$$Recall = \frac{TP}{TP + FN} \times 100\% \quad (3)$$

$$Precision = \frac{TP}{TP + FP} \times 100\% \quad (4)$$

$$F_s = \frac{2TP}{2TP + FP + FN} \times 100\% \quad (5)$$

$$k = \frac{P_0 - P_e}{1 - P_e}$$

$$P_e = \frac{(TP + FN)(TP + FP)}{\tau^2} + \frac{(FN + TN)(FP + TN)}{\tau^2},$$

$$P_0 = \frac{TP + TN}{\tau}, \tau = TP + FP + FN + TN \quad (6)$$

where, TP stands for True Positive which refers to the number of pixels accurately segmented as lakes; FP is the short of False Positive meaning the land pixels are wrongly classified as lake surface (label 1). The TN (True Negative) and FN (False Negative) indicate the land pixels are correctly segmented as background and the lake pixels are wrongly categorized as non-water (label 0), respectively.

We applied the model evaluation quantification scheme proposed by [26] based on the IoU values to interpret and categorize the model performances into qualified labels of "Excellent" when the IoU $>$ 0.90, "Good" if 0.850 $\leq$ IoU $\leq$ 0.90, "Moderate" if 0.75 $\leq$ IoU $\leq$ 0.85, "Poor" if 0.65 $\leq$ IoU $<$ 0.75, and "Failed" if IoU $>$ 0.65. This method of interpreting model performances is applied constantly for each model implementation.

III. EXPERIMENTS

A. MODEL SETUP

In the fine-tuning phase, we modified the origin SAM architecture from [60] with three new parts (Fig. 4B) of 1) Image patches: the inputted images and masks are pacified to the shape of [61] respectively for fitting well with the torch computational size. 2) Tensor filter: the patched images and masks are filtered for valid indices ($\neq 0$ or lake mask/masks present in the mask patches), meaning that only tensors with mask valid indices and corresponding images are kept and passed through the convolutions and this also assisted the boosted computation and reduced time of processing. 3) We trained the fine-tuning SAM (from pre-traind SAM, the "sam-vit-base") with bounding boxes as prompts which are defined by regular grids (20 $\times$ 20 pixels) in the presence of the ground truth masks. The model trains were implemented by our resource of a 13th Gen Intel(R) Core (TM) i9-13900 2.00 GHz Desktop with 64.0 GB (63.7 GB usable) RAM and NVIDIA GeForce RTX 4090 GPU (55.9 GB memory), applied the algorithm 1 (appendix A) for model train, evaluation and algorithm 3 (appendix C) for the model inferences. Similarly, the YOLOv8 proceeded with the Pytorch platform based on its original code.

Other models of U-net$_{(ResNet-50)}$, and DeepLab$_{(ResNet-50,EfficientNetV2)}$ of lake surface segmentation are performed in Python utilizing the TensorFlow core library within Python 3 Google Compute Engine backend (GPU). For the accelerated computations in the Colab, the resources of an NVIDIA Tesla T4 (15GB) GPU, an Intel Xeon CPU with two cores running at 2.30 GHz, and 12.7 GB of RAM were utilized. All the model code is written in Jupyter Notebook ($*$.ipynb) to exploit some advantages such as flexible run management, and friendly code editing interfaces. The model performances are summarized within Algorithm 2 (appendix B) and Algorithm 3 for the model training, evaluation, and deployment. All the model trains (including SAM and YOLO) on the datasets using default parameters and 50 epochs, batch size of 8 images, and a learning rate of 1$\times$ 10$-$5. The model parameters from the checkpoint with the lowest validation loss were saved for the model inferences.

Table 4 summarizes the number of DL model parameters and the computational time (in seconds) of each model epoch. Overall, there are positive proportions between the model sizes (number of parameters including trained and non-trainable) and the computational time. The lightest model of DeepLab$_{(EffcicientNet)}$ consumed 5 seconds to process an epoch with the Kompsat-5 data and 39 seconds with the combination of all the SAR datasets. On the other hand, more complex models and larger input data prolonged the model processing time of the U-net$_{(ResNet50)}$ model to 5.5 minutes for the combination data. It is noted that the YOLOv8 and SAM models (marked by $*$) were implemented in the local PC and GPU system boosting the second largest model (YOLOv8) to only 8 seconds for the Kompsat-5 dataset and 19 seconds for the combined dataset. The biggest SAM (636 million parameters) needed 78 and 394 seconds to complete an epoch of the Kompsat-5 and the combination datasets, respectively. Hence, the SAM model took nearly 5.5 hours to process the whole model train of 50 epochs on the combination data.

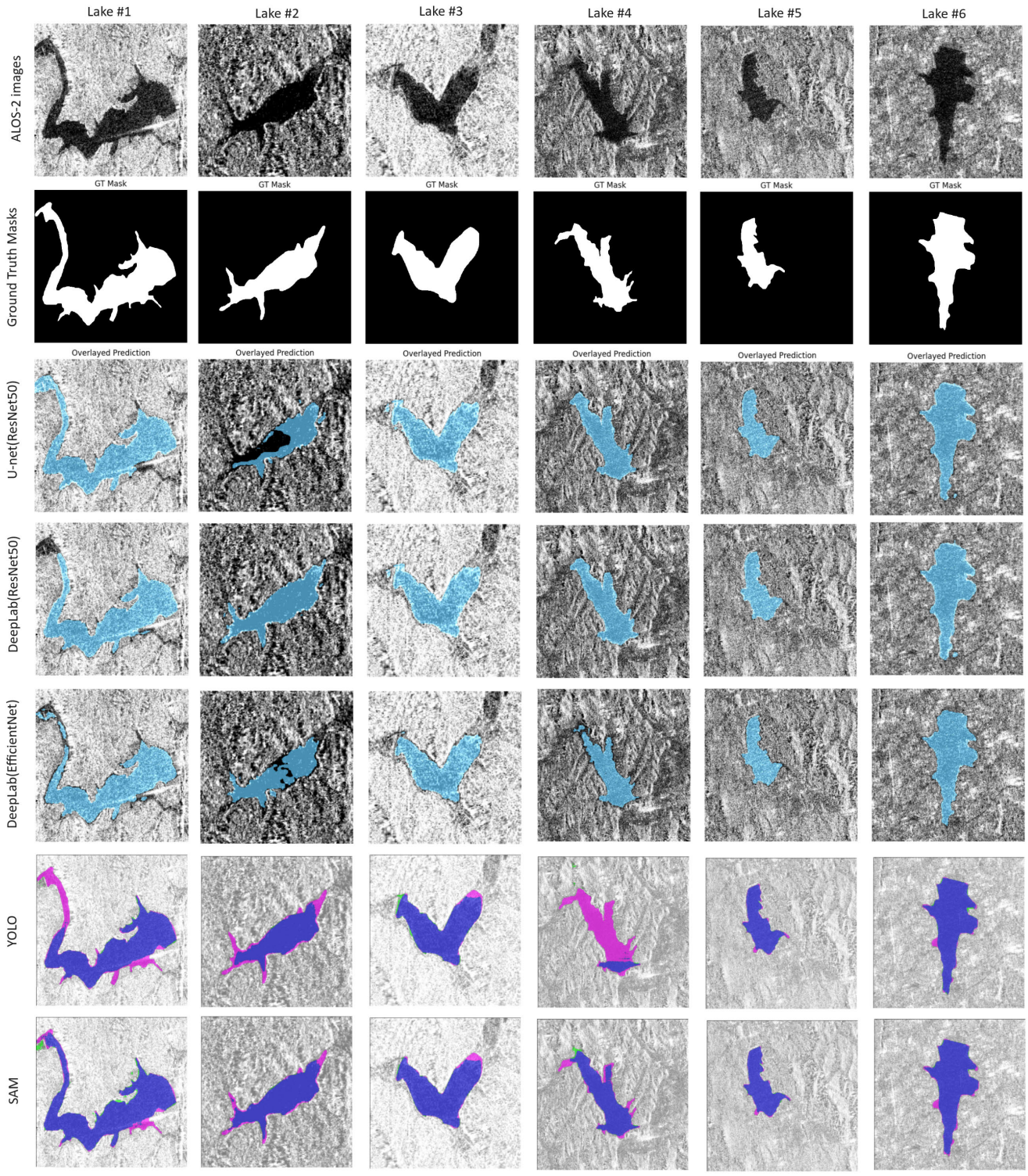

FIGURE 14. DL segmentation results on ALOS-2 images, GT Mask is a short form of ground truth mask.

B. EXPERIMENT RESULTS

This section presents model performance evaluations by the loss function for all models and datasets (Fig. 10), model accuracy metrics of overall accuracy (OA), IoU, and Kappa (k) (Fig. 11). The sub-sections describe selected showcased segmented lakes for each dataset.

Loss Functions: Fig. 10 shows different scenes of the model's loss functions (LF) in training and validating

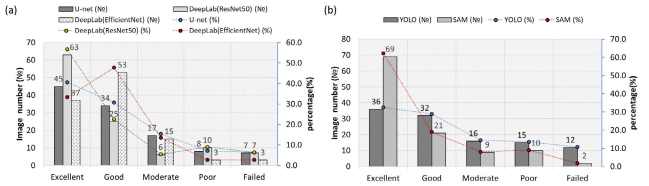

FIGURE 15. Results of model segmentation quantity and quality of the U-Net, DeepLab$_{(ResNet50)}$, DeepLab$_{(EfficientNet)}$ (a), YOLOv8x, and SAM(vit-h) (b), experiment on the ALOS-2 dataset.

50 epochs, however, the loss values decreased and the LF values of the training were higher than the values of validation meaning the good signs of the model convergence. Nonetheless, there were some epochs where the validating FL values were greater than those of training for the YOLO model using the Kompsat-5 and the combination dataset. After epoch 40th, the YOLO's FLs came to the normal scene of the training values smaller than the validating values. The U-net's FL (A) was stable and improved slightly after the 15th epoch. In contrast, the YOLOv8 LF (D) continuously fluctuated and decreased until epoch 50 (around 0.2) for all the datasets and its LFs remained highest compared to all other models. The LFs of the DeepLab$_{(ResNet50)}$ (B) and DeepLab$_{(EfficientNet)}$ (C) sunk sharply in the first 10 epochs and remained at low values of nearly zero. The SAM LFs (E) gradually decreased for all datasets and reached the lowest values at the end of the training process. There was no obvious difference in the LFs between the uses of the datasets except for the more stability of the LFs and the U-net train and validation started with low values of LF (0.28) of the combination of all data and reached the best epoch early at the 9th. The model checkpoints were saved at the $\pm$ 10 epoch for U-net$_{(ResNet50)}$, at the 27th epoch for YOLO using Kompsat-5 data, and all others saved at the end of the training for model inferencing (employment) marked by the red circles.

Overall accuracy (OA), Intersection over Union (IoU), and Kappa (k): The box plots in Figure 11 show the SAM segment performances (pink boxes) were most accurately depicted in several instances where the OA, IoU, and k median values (red lines) were higher and their outliers (the error caps and the black diamond markers) smaller than those of the other models. On the contrary, YOLO's accuracy metrics were at their lowest values (median values marked in red around 0.8), and its outliers were deeper with some values approaching 0.1. However, the YOLO metrics were improved in their values much when the model was applied to the combination dataset. The DeepLab$_{(RetNet50)}$ and DeepLab$_{(EfficientNet)}$ performances (green and blue boxes) were in second and third place, respectively. The U-Net accuracy metrics were slightly higher than the YOLO's values in general, however, the U-Net performed well with the Sentinel-1 data by the OA, IoU, and K were slightly under the SAM metrics. In a cross-comparison between the metrics, the OA values were higher than the IoU and the K values for all datasets and there was no clear difference between the IoU and K values. Furthermore, the combination of all datasets showed the highest metric values, shortest error caps, and fewer outliers (black diamond markers). There was no clear prevalence of the Kopmsat-5, ALOS-2, and Sentinel-1 over each other based on the box plots of OA, IoU, and K. We might need to assess other metrics of each dataset for more details.

1) ON KOMPSAT-5 DATASET

a: MODEL ACCURACY METRIC COMPARISON

Comparisons of the precision metrics of the five models calculated in the model's validations applied for the Kompsat-5 dataset can be provided in Table 5. The SAM(vit-h) was the most accurate with the largest values of OA, IoU, precision, Fs, and K with the five largest values marked in red. The results demonstrated that the DeepLad$_{(ResNet50)}$ validation was the second most accurate performance based on the OA, IoU, Precision, Fs, and K values. They were slightly lower than the SAM's metrics and the DeepLad$_{(ResNet50)}$'s Recall surpassed all other values. The YOLOv8x was the poorest, however, all the metric values were higher than 0.7 and Fig. 10 showed its LFs could be further improved after the 50th epoch. The performances of U-net$_{(ResNet50)}$ and DeepLab$_{(EffcicientNet)}$ models were similar. These model accuracy metrics might partly related to the segmentation results as the model took the images randomly.

b: SHOWCASED EXAMPLES OF LAKE SEGMENTATIONS

These are good examples of DL lake semantic segments using SAR images of Kompsat-5 when prompts were not applied to any model (Fig. 12) and the promptless were also used for other following datasets. Although the Kompsat-5 images contain much noise on the water surfaces, the six lakes were segmented accurately by all the models. Despite some false positives (FT) of the small pond in the images (Lake #1 and #2) segmented by the SAM, YOLO (red mask), (run on Pytorch platform), and DeepLab$_{(ResNet50)}$ models (without predicted mask), and some small areas of false negatives (FN) in all images depicted by the green areas (YOLO and SAM) and light blue of U-net, DeepLab$_{(ResNet50,EfficientNet)}$ (on Tensorflow, Keras platform), most lake areas were segmented at high accuracy (approximately 90%). In this example set, the U-net, DeepLab$_{(ResNet50)}$, and SAM results surpassed the DeepLab$_{(EfficientNet)}$ and YOLOv8 results faintly by visual assessment. Note that all the images in a column are the same, the images of the YOLO and SAM rows were displayed in the brighter mode to highlight the predicted masks. That is applied identically to other showcased examples.

c: MODEL QUALITY ASSESSMENT

Fig. 13 shows lake prediction quality classifications divided into quality sets ranging from "excellent" to "failed " for the Kompsat-5 validation images by (a) U-Net, DeepLab$_{(ResNet50,EfficientNet)}$ (group Keras, TensorFlow) and (b) YOLO, SAM (group Pytorch). The DeepLab$_{(ResNet50)}$ and DeepLab$_{(EfficientNet)}$ shared the same 41 excellent images out

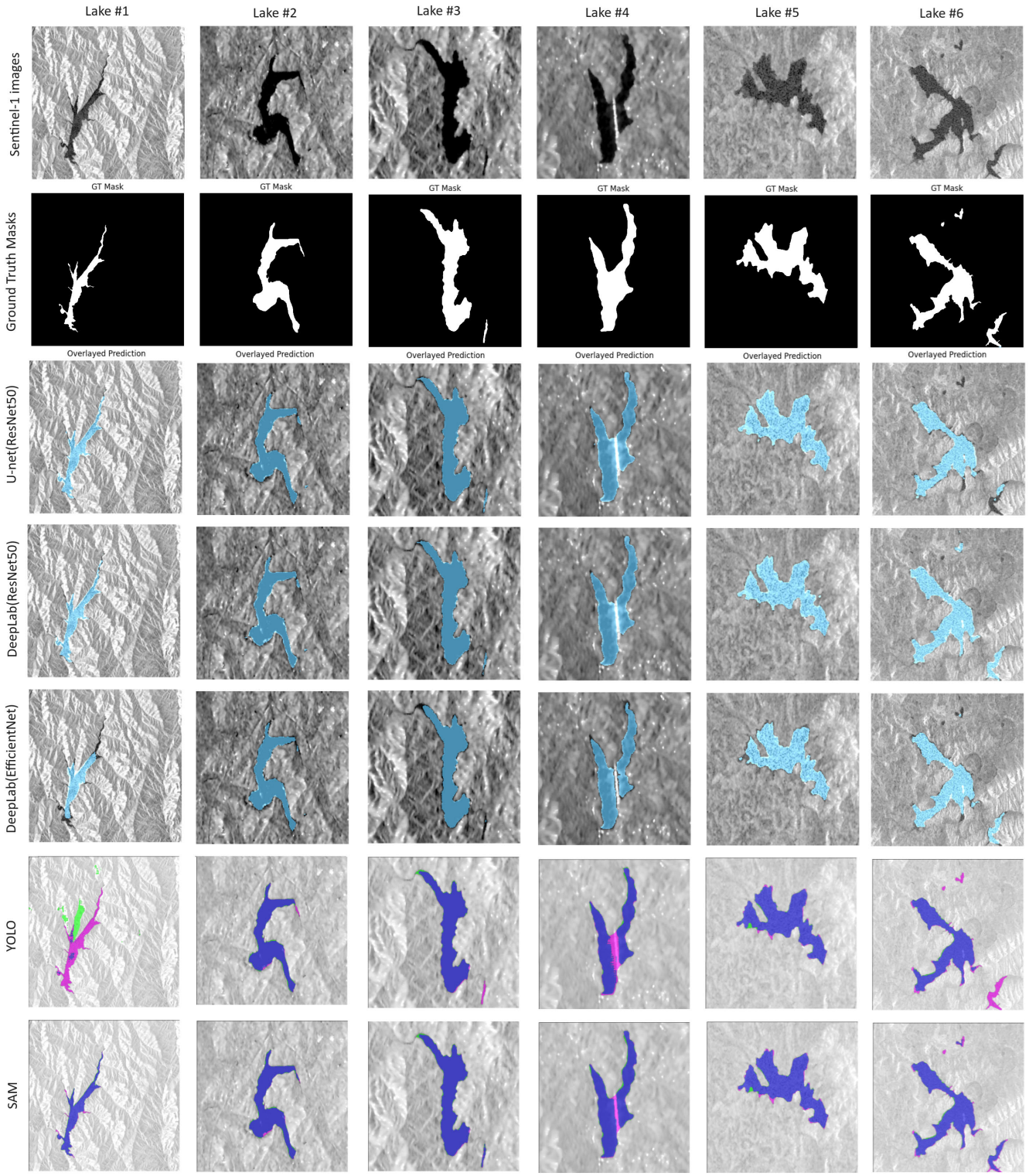

FIGURE 16. DL segmentation results on Sentinel-1 images, GT Mask is a short form of ground truth mask.

of a total of 65 images occupied around 60%. However, the good DeepLab$_{(\text{ResNet50})}$ images were 18 while categorized 11 for DeepLab$_{(\text{EfficientNet})}$, hence, the DeepLab$_{(\text{ResNet50})}$ validation surpassed the DeepLab$_{(\text{EfficientNet})}$'s in general. Both models had no poor image and a low number of moderate. The U-net model quality was lowest among the Keras,

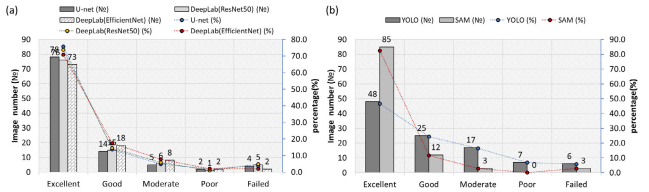

FIGURE 17. Results of model segmentation quantity and quality of the U-Net, DeepLab$_{(ResNet50)}$, DeepLab$_{(EfficientNet)}$ (a), YOLOv8x, and SAM(vit-h) (b), experiment on the Sentinel-1 dataset.

TensorFlow group. SAM predominated YOLO in quality with 46 images (71%) labeled as excellent for the SAM and 22 excellent images (32%) for YOLO and the poor and failed cases of the SAM model were also lower than those of YOLO. Only in the good category, the YOLO had more images (15≈24%) classified into ''good'' while the SAM had 7 (11%).

2) ON ALOS-2 DATASET

a: MODEL ACCURACY METRIC COMPARISON

There was a different story that the DeepLab$_{(ResNet50)}$ surpassed all other DL models using the ALOS-2 dataset while its IoU, Precision, Fs, and K values were greatest (marked in red in Table 6). The SAM(vit-h) was in second place followed by U-Net, DeepLab$_{(EffcicientNet)}$, and YOLOv8x. Compared to the Kompsat-5 metrics (Table 5), the ALOS-2 matrics were marginally higher, however, the its averaged IoU (0.792) was lower than the sum of Kompsat-5's IoU (0.796). That might be somewhat depicted in the segmentation results of the ALOS-2 dataset.

b: SHOWCASED EXAMPLES OF RESERVOIR SEGMENTATIONS

The clear demonstrations of excellent predicted masks by the DeepLab$_{(ResNet50)}$ model overlaid with the ALOS-2 images (Fig. 14) with just a small false negative (FN) on the top-left of Lake #1. More errors or false negatives (FN) and false positives (FP) appeared in the results of the DeepLab$_{(EfficientNet)}$ and YOLO models with mostly FN cases (non-predicted mask of DeepLab$_{(EfficientNet)}$ and red areas of YOLO model). Remarkedly, nearly the entire lake in image Lake #4 was not predicted as surface water by the YOLO model and was a ''poor'' case. There were several small non-water areas segmented as lake (FN) compared to the ground truth masks by the U-net and DeepLab$_{(ResNet50)}$ in the top-left of Lake #3 and at the bottom of Lake #6 images. These FN predictions (green masks) were also found in the results of the SAM model. The results of lake segmentation of all five models agreed well with the model accuracy metrics even though these were randomly selected images.

c: MODEL QUALITY ASSESSMENT

In the Keras, TensorFlow group (Fig 15a), the DeepLab$_{(ResNet50)}$ performed well repeatedly with 63 images classified as ''excellent'' occupying 57% of all validated images (111). However, the DeepLab$_{(EfficientNet)}$ had 53 images (48%) of good level and only 15 images (13.5%) of moderate quality images compared to 17 images (15.3%) of the DeepLab$_{(ResNet50)}$ model. U-net's results were the poorest, however, had a low number of poor and failed classifications as well. In the Pytorch group (Fig 15b), the SAM presented a prevailing model over the YOLO in semantic segmentation lakes using SAR images with 62% of excellent images compared to 32.4% of the YOLO model. In addition, the SAM results had only 10 and 2 images categorized as ''poor'' and ''failed'' performance, respectively. Another good aspect of the YOLO model is that it produced good results for 32 images, otherwise, SAM produced good results for 21 images.

3) ON SENTINEL-1 DATASET

a: MODEL ACCURACY METRIC COMPARISON

Sentinel-1 presented the best data for lake segmentations using DL models where most of the metric values were higher than those of Kompsat-5 and ALOS-2 datasets (Table 7). Comparing the results of different DL models we found that the SAM model produced the highest accuracy metric values with impressive overall accuracy (OA) of 0.988, and IoU of 0.982, for example. An interesting finding was that the U-net with an OA of 0.913 surpassed the other models in the Keras-TensorFlow model group, followed by the DeepLab$_{(ResNet50)}$, and DeepLab$_{(EfficientNet)}$. There was a remarkable improvement of the YOLOv8x performances but its accuracy metrics remained the lowest among all the models.

b: SHOWCASED EXAMPLES OF RESERVOIR SEGMENTATIONS

The most accurate set of DL-predicted masks overlaid with the test images of Sentinel-1 is depicted in Fig. 16. Despite the complexity in sharp of Lakes #1, 5, and 6, and much noise in the lake surfaces of image Lake #5, and 6, the U-net, DeepLab$_{(ResNet50,EfficientNet)}$, and SAM models segmented the lake masks fitted well with the ground-truth masks. The water bodies (foreground) of Lake #2, and 3 were in the perfect condition for the model prediction with no noise and contrasted well with the land (background), low values of received backscatters. Therefore, all models segmented the lake areas correctly. The foreground of Lake #4 was fine but affected by a bridge and the SAR backscatter value on the bridge was very high which led to the DeepLab$_{(EfficientNet)}$, YOLO, and SAM did not segment the bridge as lake surface. That is an example of the challenges of water surface segmentation in dealing with objects on them. Furthermore, Lake #1 consists of several branches (octopus tentacles) that could be a reason for the poor and nearly failed segmentations of the DeepLab$_{(EfficientNet)}$, and YOLO models. In addition, the image of Lake #6 includes some small ponds that were not recognized and segmented by most models except the DeepLab$_{(ResNet50)}$ with a positive true

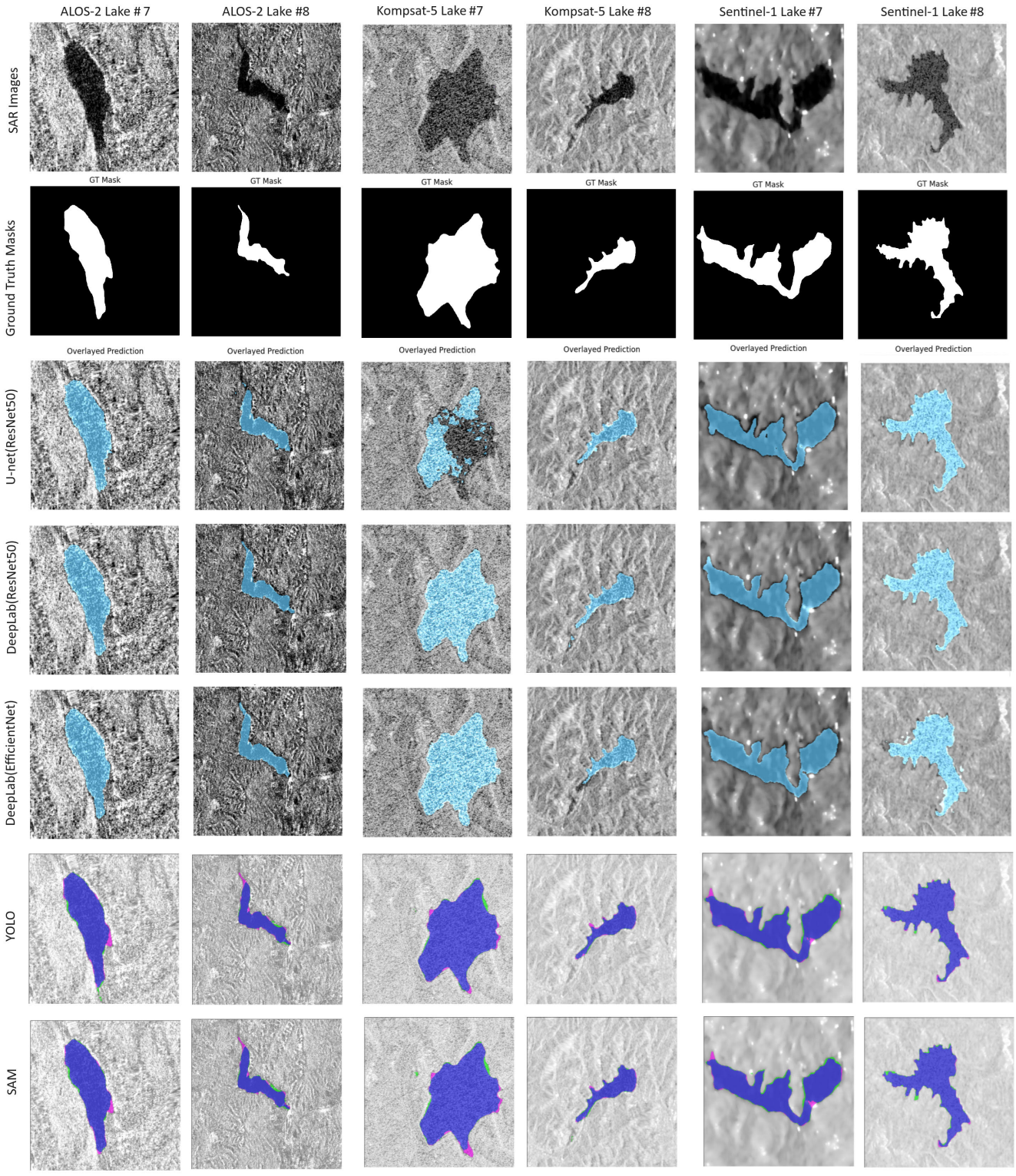

FIGURE 18. DL segmentation results on the combination of Kompsat-5, ALOS-2, and Sentinel-1 images, GT Mask is a short form of ground truth mask.

(PT) of the bigger pond on the image top. Cross-comparing between the models, the U-Net, DeepLab$_{(\text{ResNet50})}$, and SAM were in the first class and the differences between them were not clearly shown. Otherwise, the DeepLab$_{(\text{EfficientNet})}$ and YOLO's performances were moderately less accurate. To recap, all the models segmented the lake areas

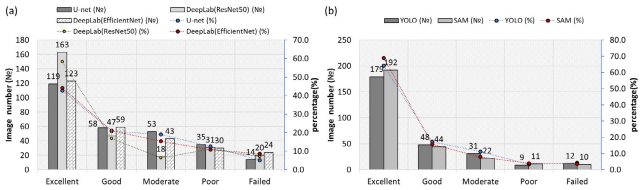

FIGURE 19. DL model segmentation quantity and quality of the U-Net, $\text{DeepLab}_{(\text{ResNet50})}$, $\text{DeepLab}_{(\text{EfficientNet})}$ (a), YOLOv8x, and SAM(vit-h) (b), experiment on the combination of Kompsat-5, ALOS-2, and Sentinel-1 datasets.

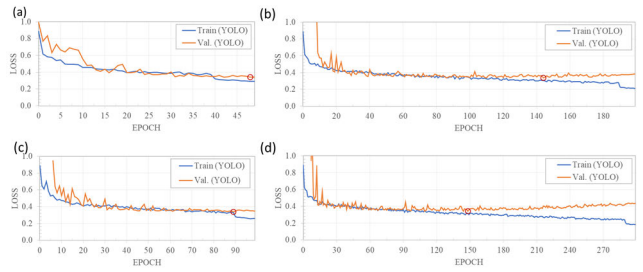

FIGURE 20. YOLOv8x loss function trained for 50 epochs (a), 100 epochs (b), 200 epochs (c), and 300 epochs (d).

well when trained and validated for the Sentinel-1 dataset.

c: MODEL QUALITY ASSESSMENT

Similar to the model accuracy metric comparisons (Table 7), there was no surprise that the U-net with the highest metric values produced more segmented images (78, 75.7%) classified as ''excellent'' and the slightly under this quality level was the $\text{DeepLab}_{(\text{ResNet50andEfficentNet})}$ model with 76 and 73 excellent images, respectively (Fig. 17a). Other quality levels of ''Good, Moderate, Poor, and Failed'' were thoroughly similar, assigned to the three models with low numbers (1 to 5) of poor and failed images. Nevertheless, the SAM frequently had better semantic segmentation results than the YOLO when they were trained with 50 epochs, and with the Sentinel-1 dataset, this scene has remained for the Pytorch group (Fig.17b). That is demonstrated by the statistics of the dominant number of 85 and 25 of ''Excellent'' and ''Good'', the lower number of ''Poor'' and ''Failed'' of zero and 3 for SAM compared to 48 and 12 of ''Excellent'' and ''Good'', 7 and 6 of ''Poor'' and ''Failed'' for the YOLO model.

4) ON THE COMBINATION OF ALL DATASETS

a: MODEL ACCURACY METRIC COMPARISON

There was an interesting change in model accuracy metrics of the DL models where the YOLOv8x accuracy is improved compared to applying it to individual datasets (Table 8). Even though YOLOv8x ranked in third place, its precision was enhanced to the first (0.962) meaning that the YOLO model handled the large dataset well. $\text{DeepLab}_{(\text{ResNet50})}$ was the best at this experiment with its recall (0.948), Fs (0.938), and K (0.908) were the highest values. The SAM(vit-h) was in second place with its OA of 0.971 and recall, precision, Fs, and K values ranged at the second-best. Although the $\text{U-Net}_{(\text{ResNet50})}$ metrics were the lowest, the differences in their accuracy between the models were minor and no model completely prevailed over all others.

b: SHOWCASED EXAMPLES OF LAKE SEGMENTATIONS

Agreed well with the accuracy metrics in Table 8, the segmentation results showed most of the model's segmented lake areas fitted well with the ground-truth masks where the $\text{DeepLab}_{(\text{ResNet50})}$ predicted masks approached perfection. The Kompsat-5 image of Lake #7 consisted of many radar-scattering speckles (salt and pepper) resulting in large areas of false positive (FP) segmentation of the lake by the $\text{U-Net}_{(\text{ResNet50})}$, reversely not a challenge for other models. However, the Sentinel-1 images of Lake #8 also had a large number of speckles, the speckle values were still lower than most pixel values of the land (background), thus all the models segmented this lake accurately. An interesting point is that the two small water areas at the end of the small part of the lake (could be small water bodies) in the Kompsat-5 image Lake #8 were not masked as lakes in the ground-truth data, on the other hand, segmented as lake areas by the only $\text{DeepLab}_{(\text{ResNet50})}$ model. If it is claimed as a missed ground-truth mask, the result of the $\text{DeepLab}_{(\text{ResNet50})}$ model will not be assigned as a failed negative, however, other models produced false positives. Comparing the YOLO and SAM models, minor differences could be found in more green pixels (FN) of the SAM compared to the YOLO's FN. In addition, the two models had mostly identical FP pixels (red) in all images. In general, all models performed well with the combination dataset except for a large FP of U-net in Lake #7.

c: MODEL QUALITY ASSESSMENT

The model performances were more uniform in quality (Fig. 19a&b) where the differences between models were narrowed when we trained and tested with a larger dataset (train/validation = 1110/279). In Fig. 19a, the $\text{DeepLab}_{(\text{ResNet50})}$ model continuously led in producing more excellent lake segmentations in the Keras-TensorFlow group with (163 images, occupied 58.4%) while the U-net and $\text{DeepLab}_{(\text{EfficientNet})}$ have achieved 119 (42.6%) and 123 (44.1%) of excellent cases, respectively. The number of images classified into other quality rankings of ''Good'', ''Moderate'', ''Poor'', and ''Failed'' decreased gradually from 60 to 20 images and the variations of differences were lower except for the ''Moderate''. This trend was also depicted by the YOLO and SAM results in Fig. 19b. However, both YOLO and SAM had the largest number of excellent images 179 (64.2%) and 192 (68.8%), respectively.

IV. DISCUSSIONS

These study experiment results demonstrated improvement in the lake segmentations of the SAM model compared to

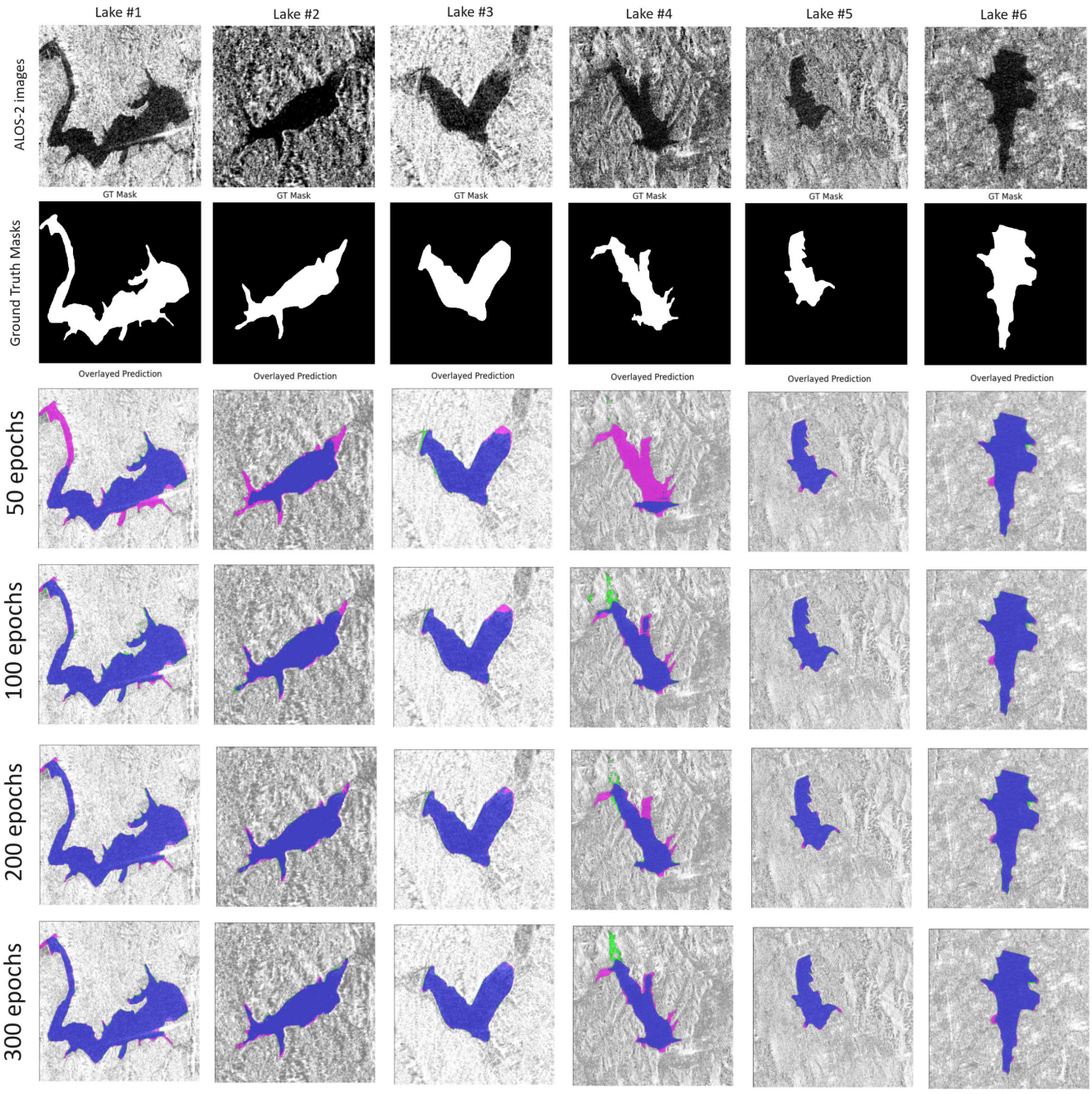

FIGURE 21. Testing training depth affects the segmentation results of the YOLOv8x model on the ALOS-2 dataset.

the SAM pre-test presented in the introduction section. That could assert that our initial hypothesis of a need to fine-tune SAM was right. Although the SAM pre-trained was trained on an unprecedentedly large segmentation dataset (more than 11 million images and one billion masks (SA-1)) [19], their performance for remote sensing data was also tested. It came up with limitations like SAM segmented masks without class information, supported by [36], and a fine-tuning SAM technique for remote sensing datasets recommended by [33] to enhance model proficiency. In addition, we assume that most of the SA-1 images were optical, and a small proportion were from space-bone sensors. Therefore, when dealing with SAR images, with different characteristics with optical data, the SAM model produces uncertainty and errors in its results. In addition, shorter wavelength radar sensors (band-x, 3.2 cm) like Kompsat-5 are more sensitive to the smaller surface roughness [62] including water surfaces affected by winds and floating vegetation. This is a great source of noise

Algorithm 1 Train for lake surface extraction Pytorch platform

Model configuration# setting model parent parameters for the training task
  Device=Torch.cuda.device connection()
  Bath_size = {a} #number of Baths = I_n/a
  Model.to(device)
  Epoch_number = {e}
  Model Initializing: Parameter Φ_M, Val loss $L_{val} = \infty$
Data pipe # Prepare training, validating, and testing input images
$\begin{cases} D_{image}D_{train} = \{(I_{n\_train}, W_x, H_y)_{i=0}^{n\_train-1}, \\ D_{val} = \{(I_{n\_val}, W_x, H_y)_{i=0}^{n\_val-1}\}, \end{cases}$
$\begin{cases} D_{test} = \{(I_{n\_test}, W_x, H_y)_{i=0}^{n\_test-1}\} \\ D_{test\_mask} = \{(I_{n\_test\_mask}, W_x, H_y)_{i=0}^{n\_test-1}\} \end{cases}$
D_dict = {
  "image": [Image.fromarray (D_{image}) for image in D_{train}],
  "label": [Image.fromarray (D_{mask}) for mask in D_{train_mask}],}
D_{input} = Dataset.from_dict(D_dict)
Initialization: model = pretrained(sam-vit-base)
  optimizer = Adam()
  loss = monai.losses.DiceCEloss()
  dataset = dataloader(D_{input})
Training SAM: for epoch ∈ {e} do
  Foreach batch (D_{train}, D_{train_mask}) ∈ D_{input} do # training data loader
  Model_output = model(pixcelvalue = batch [pixel_values].to(device)), # @GPU
  $\hat{L}_{T:i:i+B-1} \leftarrow f_M\left(D_{T:i:i+B-1}, \theta_M\right) L_{DCEi:i+B-1}$
  DiceCEloss($D_{T:i:i+B-1}$, $L_{DCEi:i+B-1}$)
  $\theta_M \leftarrow$
  AdamOp(θ_M, $\nabla_{\theta_M} \frac{1}{B} \sum_{j=i}^{i+B-1} L_{DCEj}, \alpha$)
  $L_T \leftarrow \frac{1}{N_{train}/B} \sum_{i=1}^{N_{train}/B} \sum_{j=1}^{i+B-1} L_{DCEj}$
  Print(L_T);
Model Evaluation:
  Foreach batch ($D^F_{val\_B}$, $D^F_{val\_mask\_B}$) ∈ D_{input} do # validating dataloader
  Model_output = model(pixcelvalue = batch [pixel_values].to(device)), # @GPU
  $\hat{L}^F_{T:i:i+B-1} \leftarrow f_M\left(D^F_{T:i:i+B-1}, \theta_M\right) L_{DCEi:i+B-1}$
  DiceCEloss($D^F_{T:i:i+B-1}$, $L_{DCEi:i+B-1}$)
  $\theta_M \leftarrow$
  AdamOp(θ_M, $\nabla_{\theta_M} \frac{1}{B} \sum_{j=i}^{i+B-1} L_{DCEj}, \alpha$)
  $L_V \leftarrow \frac{1}{N_{val}/B} \sum_{i=1}^{N_{val}/B} \sum_{j=1}^{i+B-1} L_{DCEj}$
  Print(L_V);
Save model checkpoint: while $L_V < L_{V-Best}$ save model f_M,

Algorithm 2 Train for lake surface extraction Tensorflow-Keras platform

1. Seeding for Reproducibility: system_config(seed_values)
  Random.seed(s)
  Os.enviroment(o)
  system_config ←
5. Data pipe
  $D_{input} = \left\{(I_n, W_x, H_y)_{i=0}^{n}, (M_n, W_x, H_y)_{i=0}^{n}\right\}$, $D_{test} = (I_t, W_x, H_y)_{i=0}^{t}$
  $D_{train} = slpit(D_{input}, ratio = 0.8)$, $D_{val} = slpit(D_{input}, ratio = 0.2)$,
  $Readimage, mask\ (D_{train,} D_{val}) = x, y \leftarrow$
  Tensorflow sparse (x,y) = x,y ←
10. Building the model
  Conv_block (x)
    x= (Conv2D(f=2,p=same), BatchNor(x), Aciva tion(relu))
  Encoder Block (x, p)
    x= Conv_block (x, p)
    P=MaxPool2D(2,2)
    x ←
  Decoder_block (x)
    x= (Conv2DT(), Concatenate(), conv_block())
    x ←
  Build_model(input_shape)→ input=keras.Input(I nput_shape)
    build: Encoder, Bridge, Decorder, output
    model ←
23. Initialization: model = pretrained(model)
  Opt=(Adam(lr), loss("binary"), metric("acc")
  Callback = (Checkpoint(ver=1), Reduce $LROP$(m on=v_loss, f, p), EarlyStopping(p=30))
26. Model train: for epoch ∈ {e} do
  Foreach batch (D_{train}, D_{train_mask}) ∈ D_{input} do #t raining data loader
  Model_output = model.fit(data = ...{x,y}, callback(), opt(θ_M))
  $\hat{L}_{T:i:i+B-1} \leftarrow f_M\left(D_{T:i:i+B-1}, \theta_M\right) L_{LROPi:i+B-1}$
  ReduceLROnPlateau($D_{T:i:i+B-1}$, $L_{LROPi:i+B-1}$)
  $\theta_M \leftarrow$
  AdamOp(θ_M, $\nabla_{\theta_M} \frac{1}{B} \sum_{j=i}^{i+B-1} L_{LROPj}, \alpha$)
  $L_T \leftarrow \frac{1}{N_{train}/B} \sum_{i=1}^{N_{train}/B} \sum_{j=1}^{i+B-1} L_{LROPj}$
  Print(L_T);
36. Model validation:
  Foreach batch (D_{val}, D_{val_mask}) ∈ D_{input} do #vali dating dataloader
  Model_output = model.fit(data = {x,y}, callback(), opt(θ_M))
  $\hat{L}_{V:i:i+B-1} \leftarrow$
  $f_M\left(D_{V:i:i+B-1}, \theta_M\right) L_{LROPi:i+B-1}$
  ReduceLROnPlateau($D_{T:i:i+B-1}$, $L_{LROPi:i+B-1}$)
  $\theta_M \leftarrow$
  AdamOp(θ_M, $\nabla_{\theta_M} \frac{1}{B} \sum_{j=i}^{i+B-1} L_{LROPj}, \alpha$)
  $L_V \leftarrow \frac{1}{N_{val}/B} \sum_{i=1}^{N_{val}/B} \sum_{j=1}^{i+B-1} L_{LROPj}$
  Print(L_V);
46. Save model checkpoint: while $L_V < L_{V-Best}$ save model f_M,

in SAR remote-sensing data creating big challenges for not only traditional methods of water extractions but also modern deep learning models. These study's results showed significant improvement in water surface segmentations of example lakes in the Kompsat-5 images reaching up to around 90%. Even though we have not done pre-tests for the ALOS-2, and Sentinel-1 data, we believe that after the DL models are trained they can be employed to extract the lake water

Algorithm 3 Model test for tested dataset

1. **Input**: $D_{test} = (I_t, W_x, H_y)_{i=0}^{t} \rightarrow$ trained f_M
2. Model parameter: θ_M
3. **Output**: Predictions on test images:
4. output $= f_M(D_{test:i)}, Input_{prompt}$ $(optional)$, $tensor$ (pt)),
   `#calculate Accuracy metrics`
5. masks$\leftarrow$
6. Overlaid predicted masks with corresponding images
7. Plot ()

surfaces on these data correctly (more than 80%), not only for regions of South Korea but for any lakes and seservoirs.

For individual datasets (Kompsat-5, ALOS-2, and Sentinel-1), the YOLO presented poorer results compared to the outputs of other models with 50-epoch trains. However, in the results of loss function (LF) analysis, we found that the YOLO's LFs could be improved (decreased) more after the 50th epoch, and that is a good sign of a potential accuracy amelioration for this model. Therefore, the 50-epoch train could be an overly shallow fine-tune but sufficient for the U-net$_{(ResNet50)}$ and DeepLab$_{(ResNet50,EfficientNet)}$, previous studies fine-tunned the YOLO for 120 [63], 250 [64], and 300 epochs [65], [66]. That is a reason for deeper training tests for the YOLO on the poorest results of ALOS-2 (Fig. 20) and, interestingly, the YOLO LF was continuously decreased to 100 epoch (best validation at 89th epoch), when the number epoch set for 200 the FL improved to 200th epoch and (val. of 149). While the deeper train was set for 300 epochs, the FL of training continued to improve to the end of the training phase but the validation LF did not improve at epoch 50. That could be an overfitting of the model after the epoch 149 might be due to our small dataset's limitation (553 images). In addition, the test of YOLO lake segmentation also demonstrated the higher accuracy of its results (less red and green pixels) when the number of training epochs increased (Fig. 21). It is additionally a good showcase of when the model overfitting, the segmentation accuracy is not improved as we have not seen an improvement of the 300-epoch compared to the 200-epoch trains. As DL model trains are time and resource-consuming (Table 4), it is worth discussing the trade-offs between computational time and model accuracy. A longer training time may ensure a better model result, however, this is only the case for model convergence. Another considered aspect is the model's sizes related to its number of parameters (size n to x of YOLO, ViT-tiny, small, base, large, and huge of the SAM model e.g.), meaning that setting up a more complex model will cost more time and computational memory and, reversely, can gain more accurate model outputs. It is recommended to examine available resources, input datasets, and expected model accuracy to decide on chose appropriate model configuration.

The results of DL lake/reservoir segmentations with different models and datasets demonstrated the supervisor of this modeling technique with a strong focus on the interested targets (lakes). That is unlike most pixel-based and object-based classification methods the post-classifications often require extra work for cleaning the model outputs (model result refinement) to produce good final results [67], sometimes we must use other tools to refine the results [68]. Nonetheless, the DL model's outcomes in this study are well clean except for some small false negatives (small ponds that were out of our masking and annotating task target) that require some elimination but it is a negligible task. Additionally, the level of errors in the results depends on the quality of the inputted SAR images and aforementionedly regarding the types of SAR sensors, and the surface roughness and types. Although open water surfaces are most inherently detectable, particularly with a longer wavelength such as Sentinel-1 (band-C, 5.6cm) and ALOS-2 (band-L, 22.9 cm) which appear dark and strongly related to SAR power function compared to other land types (background) [69], common thresholding (Otsu, KI) and Chan-Vese segmentation had some limitations dealing with vegetation intervention in water bodies and recommended further deep learning techniques to obtain higher accuracy [70]. The fine-tuned SAM can be applied to analogous C, L, and X band SAR images, however, it may reveal potential limitations for the other SAR sensors as they have distinctive characteristics with the used data in this study.

Besides model accuracy, the quality of overall performances is valuable information for choosing a suitable DL model for a particular task and at a specific resource. Additionally, it often requires an intensive review of present topics and an understanding of DL aspects such as DL concepts, research gaps, computational infrastructure or platform, and applications [71]. Every DL model has their pros and cons like the SAM had the most robustness and generality but it is highly complex and time-consuming in the training process (Table 9). The YOLO is also complex however, it is fast, particularly in object detection (suitable for real-time operation) [72]. The model selection is also based on the priority of the modeling purposes, for example when we have a good computational infrastructure (CPU, RAM, GPU; e.g.) and model accuracy is the main focus, the SAM, YOLO, and DeepLab$_{(ResNet50)}$ are more relevant. However, the responses of each DL model to a dataset will be different, for example, the YOLOv8x dealt better with larger datasets rather than smaller ones, and the slower model convergence (more training epochs required), (we set a learning rate of all model was 10^{-5}). It is noted that larger training datasets might warranty higher model outputs [73] and reduce the risk of overfitting and spurious correlations [74], however, we need to recall the current resources. This could be a future research direction searching for optimal algorithms to deal with large datasets and lower the barriers of required computational resources. Hence, the limitations of time and scalability of DL models can be solved.

V. CONCLUSION

Through our intensive experiments of the DL models for semantic water bodies (lake and reservoirs) segmentations, DL models are promising avenues for this task with high accuracy and model implementation efficiency. The SAM presented an excellent tool for its most accurate results, however, the model runs are time-consuming. Although, for the individual datasets, the YOLO demonstrated poor performances, for the largest dataset of the combination of all images, the YOLOv8x indicated its robustness with highly accurate results and computational time effectiveness. The DeepLab$_{(ResNet50)}$ is also highly recommended with the second most precision results and followed by the U-net$_{(ResNet50)}$, and the DeepLab$_{(ResNet50)}$ models. Despite the DL models are possibly more popular and applied for optical images, this work proves that they can work effectively with SAR remote sensing data for specific aims (surface water extraction e.g.). Also, this research collected more than 1300 SAR images of lakes, when it comes to DL modeling, larger datasets may produce higher accuracy results and prevent model overfitting. In general, DL models are recommended in the remote sensing domains, particularly for SAR data.

APPENDIX A

See Algorithm 1.

APPENDIX B

See Algorithm 2.

APPENDIX C

See Algorithm 3.

REFERENCES

[1] W. Feng, H. Sui, W. Huang, C. Xu, and K. An, "Water body extraction from very high-resolution remote sensing imagery using deep U-Net and a superpixel-based conditional random field model," *IEEE Geosci. Remote Sens. Lett.*, vol. 16, no. 4, pp. 618–622, Apr. 2019.

[2] T. S. Akiyama, J. Marcato Jr., W. N. Gonçalves, P. O. Bressan, A. Eltner, F. Binder, and T. Singer, "Deep learning applied to water segmentation," *Int. Arch. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. 43, pp. 1189–1193, Aug. 2020.

[3] P. Freitas, G. Vieira, J. Canário, W. F. Vincent, P. Pina, and C. Mora, "A trained mask R-CNN model over PlanetScope imagery for very-high resolution surface water mapping in boreal forest-tundra," *Remote Sens. Environ.*, vol. 304, Apr. 2024, Art. no. 114047.

[4] L. Huang, B. Jiang, S. Lv, Y. Liu, and Y. Fu, "Deep learning-based semantic segmentation of remote sensing images: A survey," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 17, pp. 8370–8396, 2023.

[5] S. Dong, L. Pang, Y. Zhuang, W. Liu, Z. Yang, and T. Long, "Optical remote sensing water-land segmentation representation based on proposed SNS-CNN network," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, Jul. 2019, pp. 3895–3898.

[6] T. Mayer, A. Poortinga, B. Bhandari, A. P. Nicolau, K. Markert, N. S. Thwal, A. Markert, A. Haag, J. Kilbride, F. Chishtie, A. Wadhwa, N. Clinton, and D. Saah, "Deep learning approach for Sentinel-1 surface water mapping leveraging Google Earth Engine," *ISPRS Open J. Photogramm. Remote Sens.*, vol. 2, Dec. 2021, Art. no. 100005.

[7] Y. Yang, Z. Miao, H. Zhang, B. Wang, and L. Wu, "Lightweight attention-guided Yolo with level set layer for landslide detection from optical satellite images," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 17, pp. 3543–3559, 2024.

[8] G. H. Aly, M. Marey, S. A. El-Sayed, and M. F. Tolba, "YOLO based breast masses detection and classification in full-field digital mammograms," *Comput. Methods Programs Biomed.*, vol. 200, Mar. 2021, Art. no. 105823.

[9] M. A. Al-Masni, M. A. Al-Antari, J.-M. Park, G. Gi, T.-Y. Kim, P. Rivera, E. Valarezo, M.-T. Choi, S.-M. Han, and T.-S. Kim, "Simultaneous detection and classification of breast masses in digital mammograms via a deep learning YOLO-based CAD system," *Comput. Methods Programs Biomed.*, vol. 157, pp. 85–94, Apr. 2018.

[10] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.

[11] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[12] K. Yuan, X. Zhuang, G. Schaefer, J. Feng, L. Guan, and H. Fang, "Deep-learning-based multispectral satellite image segmentation for water body detection," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 7422–7434, 2021.

[13] F. Isikdogan, A. C. Bovik, and P. Passalacqua, "Surface water mapping by deep learning," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 10, no. 11, pp. 4909–4918, Nov. 2017.

[14] B. Pradhan, M. I. Sameen, and B. Kalantar, "Optimized rule-based flood mapping technique using multitemporal RADARSAT-2 images in the tropical region," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 10, no. 7, pp. 3190–3199, Jul. 2017.

[15] J. Li, R. Ma, Z. Cao, K. Xue, J. Xiong, M. Hu, and X. Feng, "Satellite detection of surface water extent: A review of methodology," *Water*, vol. 14, no. 7, p. 1148, Apr. 2022.

[16] Z. Guo, L. Wu, Y. Huang, Z. Guo, J. Zhao, and N. Li, "Water-body segmentation for SAR images: Past, current, and future," *Remote Sens.*, vol. 14, no. 7, p. 1752, Apr. 2022.

[17] Q. Wu, May 18, 2023, "Segment-anything-py: An unofficial Python package for meta AI's segment anything model," [Online]. Available: https://zenodo.org/records

[18] S. Shankar, L. A. Stearns, and C. J. van der Veen, "Semantic segmentation of glaciological features across multiple remote sensing platforms with the segment anything model (SAM)," *J. Glaciol.*, vol. 2023, pp. 1–10, Nov. 2023.

[19] W. Ji, J. Li, Q. Bi, T. Liu, W. Li, and L. Cheng, "Segment anything is not always perfect: An investigation of SAM on different real-world applications," 2023, *arXiv:2304.05750*.

[20] Z. Yan, J. Li, X. Li, R. Zhou, W. Zhang, Y. Feng, W. Diao, K. Fu, and X. Sun, "RingMo-SAM: A foundation model for segment anything in multimodal remote-sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 61, 2023, Art. no. 5625716.

[21] E. O. Yilmaz and T. Kavzoglu, "Quality assessment for multi-resolution segmentation and segment-anything model using worldview-3 imagery," *Int. Arch. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. 48, pp. 383–390, Mar. 2024.

[22] R. Yang, G. He, R. Yin, G. Wang, Z. Zhang, T. Long, and Y. Peng, "Weakly-semi supervised extraction of rooftop photovoltaics from high-resolution images based on segment anything model and class activation map," *Appl. Energy*, vol. 361, May 2024, Art. no. 122964.

[23] K. Chen, C. Liu, H. Chen, H. Zhang, W. Li, Z. Zou, and Z. Shi, "RSPrompter: Learning to prompt for remote sensing instance segmentation based on visual foundation model," *IEEE Trans. Geosci. Remote Sens.*, vol. 62, 2024, Art. no. 4701117.

[24] J. E. Gallagher, A. Gogia, and E. J. Oughton, "A multispectral automated transfer technique (MATT) for machine-driven image labeling utilizing the segment anything model (SAM)," 2024, *arXiv:2402.11413*.

[25] B. Baziak, M. Bodziony, and R. Szczepanek, "Mountain streambed roughness and flood extent estimation from imagery using the segment anything model (SAM)," *Hydrology*, vol. 11, no. 2, p. 17, Jan. 2024.

[26] A. Moghimi, M. Welzel, T. Celik, and T. Schlurmann, "A comparative performance analysis of popular deep learning models and segment anything model (SAM) for river water segmentation in close-range remote sensing imagery," *IEEE Access*, vol. 12, pp. 52067–52085, 2024.

[27] D. Zhao, B. Yuan, Z. Chen, T. Li, Z. Liu, W. Li, and Y. Gao, "Panoptic perception: A novel task and fine-grained dataset for universal remote sensing image interpretation," *IEEE Trans. Geosci. Remote Sens.*, vol. 12, 2024, Art. no. 5620714.

[28] G. Kasmi, Y.-M. Saint-Drenan, D. Trebosc, R. Jolivet, J. Leloux, B. Sarr, and L. Dubus, "A crowdsourced dataset of aerial images with annotated solar photovoltaic arrays and installation metadata," *Sci. Data*, vol. 10, no. 1, p. 59, Jan. 2023.

[29] B. Xue, H. Cheng, Q. Yang, Y. Wang, and X. He, "Adapting segment anything model to aerial land cover classification with low rank adaptation," *IEEE Geosci. Remote Sens. Lett.*, vol. 21, pp. 1–5, 2024.

[30] L. Ding, K. Zhu, D. Peng, H. Tang, K. Yang, and L. Bruzzone, "Adapting segment anything model for change detection in VHR remote sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 62, 2024, Art. no. 5611711.

[31] L. P. Osco, Q. Wu, E. L. de Lemos, W. N. Gonçalves, A. P. M. Ramos, J. Li, and J. Marcato, "The segment anything model (SAM) for remote sensing applications: From zero to one shot," *Int. J. Appl. Earth Observ. Geoinf.*, vol. 124, Nov. 2023, Art. no. 103540.

[32] H. Zheng, C. Zhang, K. Guan, Y. Deng, S. Wang, B. L. Rhoads, A. J. Margenot, S. Zhou, and S. Wang, "Segment any stream: Scalable water extent detection with the segment anything model," in *Proc. NeurIPS*, 2023, pp. 1–7.

[33] T. Chen et al., "SAM fails to segment anything?—SAM-adapter: Adapting SAM in underperformed scenes: Camouflage, shadow, and more," 2023, *arXiv:2304.09148*.

[34] L. Ke, M. Ye, M. Danelljan, Y. Liu, Y.-W. Tai, C.-K. Tang, and F. Yu, "Segment anything in high quality," 2023, *arXiv:2306.01567*.

[35] L. Ding, K. Zhu, D. Peng, H. Tang, K. Yang, and L. Bruzzone, "Adapting segment anything model for change detection in HR remote sensing images," 2023, *arXiv:2309.01429*.

[36] X. Ma, Q. Wu, X. Zhao, X. Zhang, M.-O. Pun, and B. Huang, "SAM-assisted remote sensing imagery semantic segmentation with object and boundary constraints," 2023, *arXiv:2312.02464*.

[37] Q. Wu, "Leafmap: A Python package for interactive mapping and geospatial analysis with minimal coding in a jupyter environment," *J. Open Source Softw.*, vol. 6, no. 63, p. 3414, Jul. 2021.

[38] S. Gillies, B. Ward, and A. Petersen. (2013). *Rasterio: Geospatial Raster I/O for Python Programmers*. [Online]. Available: https://github.com/mapbox/rasterio

[39] K. Jordahl, J. Van den Bossche, J. Wasserman, J. McBride, J. Gerard, J. Tratner, M. Perry, and C. Farmer, "geopandas/geopandas: v0.5.0," Apr. 2019. Accessed: Feb. 15, 2024, doi: 10.5281/zenodo.2705946.

[40] S. Ren, F. Luzi, S. Lahrichi, K. Kassaw, L. M. Collins, K. Bradbury, and J. M. Malof, "Segment anything, from space?" in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2024, pp. 8340–8350.

[41] S. J. Park, "Generality and specificity of landforms of the Korean peninsula, and its sustainability," *J. Korean geographical Soc.*, vol. 49, no. 5, pp. 656–674, Jan. 2014.

[42] B.-J. So, J.-Y. Kim, H.-H. Kwon, and C. H. R. Lima, "Stochastic extreme downscaling model for an assessment of changes in rainfall intensity-duration-frequency curves over South Korea using multiple regional climate models," *J. Hydrol.*, vol. 553, pp. 321–337, Oct. 2017.

[43] B. Kim, J.-H. Park, G. Hwang, M.-S. Jun, and K. Choi, "Eutrophication of reservoirs in South Korea," *Limnology*, vol. 2, no. 3, pp. 223–229, Dec. 2001.

[44] J. Terven, D.-M. Córdova-Esparza, and J.-A. Romero-González, "A comprehensive review of Yolo architectures in computer vision: From YOLOv1 to YOLOv8 and YOLO-NAS," *Mach. Learn. Knowl. Extraction*, vol. 5, no. 4, pp. 1680–1716, Nov. 2023.

[45] T. Ahamed, "Big data scheme from remote sensing applications: Concluding notes for agriculture and forestry applications," in *Remote Sensing Application: Regional Perspectives in Agriculture and Forestry*, T. Ahamed, Ed., Singapore: Springer, 2022, pp. 351–361.

[46] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 8748–8763.

[47] C. Dewi, R.-C. Chen, Y.-T. Liu, X. Jiang, and K. D. Hartomo, "YOLO V4 for advanced traffic sign recognition with synthetic training data generated by various GAN," *IEEE Access*, vol. 9, pp. 97228–97242, 2021.

[48] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017, *arXiv:1706.03762*.

[49] A. Younis, Q. Li, Z. Afzal, M. Jajere Adamu, H. Bello Kawuwa, F. Hussain, and H. Hussain, "Abnormal brain tumors classification using ResNet50 and its comprehensive evaluation," *IEEE Access*, vol. 12, pp. 78843–78853, 2024.

[50] K. Singh Gill, V. Anand, S. Malhotra, and S. Devliyal, "Sports game classification and detection using ResNet50 model through machine learning techniques using artificial intelligence," in *Proc. 3rd Int. Conf. Innov. Technol. (INOCON)*, Mar. 2024, pp. 1–5.

[51] J. Laitala and L. Ruotsalainen, "Computer vision based planogram compliance evaluation," *Appl. Sci.*, vol. 13, no. 18, p. 10145, Sep. 2023.

[52] A. Corovic, V. Ilic, S. Đuric, M. Marijan, and B. Pavkovic, "The real-time detection of traffic participants using YOLO algorithm," in *Proc. 26th Telecommun. Forum (TELFOR)*, Nov. 2018, pp. 1–4.

[53] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. 18th Int. Conf. Med. Image Comput. Comput.-Assist. Intervent. (MICCAI)*, Munich, Germany. Cham, Switzerland: Springer, Jan. 2015, pp. 234–241.

[54] P. Zhang, Y. Ban, and A. Nascetti, "Learning U-Net without forgetting for near real-time wildfire monitoring by the fusion of SAR and optical time series," *Remote Sens. Environ.*, vol. 261, Aug. 2021, Art. no. 112467.

[55] R. Huang, J. Pedoeem, and C. Chen, "YOLO-LITE: A real-time object detection algorithm optimized for non-GPU computers," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2018, pp. 2503–2510.

[56] Y. Huang, Q. Yan, Y. Li, Y. Chen, X. Wang, L. Gao, and Z. Tang, "A YOLO-based table detection method," in *Proc. Int. Conf. Document Anal. Recognit. (ICDAR)*, Sep. 2019, pp. 813–818.

[57] C. Sunil, C. Jaidhar, and N. Patil, "Cardamom plant disease detection approach using EfficientNetV2," *IEEE Access*, vol. 10, pp. 789–804, 2022.

[58] D. Pestana, P. R. Miranda, J. D. Lopes, R. P. Duarte, M. P. Véstias, H. C. Neto, and J. T. De Sousa, "A full featured configurable accelerator for object detection with YOLO," *IEEE Access*, vol. 9, pp. 75864–75877, 2021.

[59] Open-MMLab. *YOLOv8*. Accessed: Feb. 3, 2024. [Online]. Available: https://github.com/open-mmlab/mmyolo/tree/main/configs/yolo

[60] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment anything," 2023, *arXiv:2304.02643*.

[61] B.-S. Kim and H.-S. Kim, "Estimation of the flash flood severity using runoff hydrograph and flash flood index," *J. Korea Water Resour. Assoc.*, vol. 41, no. 2, pp. 185–196, Feb. 2008.

[62] U. Soergel, *Review of Radar Remote Sensing on Urban Areas*. Dordrecht, The Netherlands: Springer, 2010, pp. 1–47.

[63] Y. Hui, J. Wang, and B. Li, "WSA-YOLO: Weak-supervised and adaptive object detection in the low-light environment for YOLOV7," *IEEE Trans. Instrum. Meas.*, vol. 73, pp. 1–12, 2024.

[64] C. Hou, Z. Li, X. Shen, and G. Li, "Real-time defect detection method based on YOLO-GSS at the edge end of a transmission line," *IET Image Process.*, vol. 18, no. 5, pp. 1315–1327, Apr. 2024.

[65] Z. Wang, Z. Hua, Y. Wen, S. Zhang, X. Xu, and H. Song, "E-YOLO: Recognition of estrus cow based on improved YOLOv8n model," *Expert Syst. Appl.*, vol. 238, Mar. 2024, Art. no. 122212.

[66] L. Kang, Z. Lu, L. Meng, and Z. Gao, "YOLO-FA: Type-1 fuzzy attention based Yolo detector for vehicle detection," *Expert Syst. Appl.*, vol. 237, Mar. 2024, Art. no. 121209.

[67] B. Johnson and Z. Xie, "Unsupervised image segmentation evaluation and refinement using a multi-scale approach," *ISPRS J. Photogramm. Remote Sens.*, vol. 66, no. 4, pp. 473–483, Jul. 2011.

[68] X. Long, K. Deng, G. Wang, Y. Zhang, Q. Dang, Y. Gao, H. Shen, J. Ren, S. Han, E. Ding, and S. Wen, "PP-YOLO: An effective and efficient implementation of object detector," 2020, *arXiv:2007.12099*.

[69] X. Cai, L. Feng, X. Hou, and X. Chen, "Remote sensing of the water storage dynamics of large lakes and reservoirs in the Yangtze river basin from 2000 to 2014," *Sci. Rep.*, vol. 6, no. 1, p. 36405, Nov. 2016.

[70] W.-J. Wang, D. Kim, G. Kim, K. T. Kim, S. Kim, and H. S. Kim, "Flood risk assessment of the naeseongcheon stream basin, Korea using the grid-based flood risk index," *J. Hydrol., Regional Stud.*, vol. 51, Feb. 2024, Art. no. 101619.

[71] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, "Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions," *J. Big Data*, vol. 8, no. 1, pp. 1–74, Mar. 2021.

[72] N. H. Quang, H. Lee, N. Kim, and G. Kim, "Real-time flash flood detection employing the YOLOv8 model," *Earth Sci. Informat.*, vol. 17, no. 5, pp. 4809–4829, Oct. 2024.

[73] M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic, "Deep learning applications and challenges in big data analytics," *J. Big Data*, vol. 2, no. 1, p. 1, Dec. 2015.

[74] W. Ye, G. Zheng, X. Cao, Y. Ma, and A. Zhang, "Spurious correlations in machine learning: A survey," 2024, *arXiv:2402.12715*.

NGUYEN HONG QUANG was born in Thái Binh, Vietnam, in 1977. He received the B.Sc. degree in land survey and the M.Sc. degree in technical geodesy from Hanoi University of Mining and Geology, in 1999 and 2007, respectively, and the Ph.D. degree (magna cum laude) in geography from Georg-August-Universität Göttingen, Göttingen, Germany, in 2016.

From 2000 to 2015, he was a Geodesy Engineer with Vietnam Natural Resource and Environment Corporation under the Ministry of Natural Resource and Environment. Since 2016, he has been a Senior Researcher with Vietnam National Space Center, Vietnam. Currently, he is a Postdoctoral Fellow at the Institute for Smart Infrastructure, Gangneung–Wonju National University, South Korea. Since 2015, he has had 45 publications, including research articles, books, and conference proceedings in hydrological modeling and remote sensing. His current research interests include deep learning, computer vision, change detection, machine learning, optical and SAR remote sensing, and GIS for applications of disaster prediction, monitoring, and analysis.

HANNA LEE was born in Seoul, South Korea, in 1977. She received the B.S. and M.S. degrees in civil engineering from Yonsei University, Seoul, in 2001, and the Ph.D. degree in geoinformatic engineering from Gangneung–Wonju National University, Gangneung, South Korea, in 2024.

From 2001 to 2004, she was a Researcher at Korea Research Institute for Human Settlements. Since 2024, she has been a Researcher at the Institute for Smart Infrastructure, Gangneung–Wonju National University. Her research interests include GIS data processing, wildfire damage analysis using satellite image data, and landslide prediction and analysis using GIS.

Dr. Lee was a recipient of Korean Society of Civil Engineers Best Paper Award, in 2023.

EUI-MYOUNG KIM was born in Jinju, South Korea, in 1970. He received the B.S. and M.S. degrees in urban engineering from Gyeongsang National University, Jinju, in 1994 and 1996, respectively, and the Ph.D. degree in civil engineering from Yonsei University, Seoul, South Korea, in 2000. From 2000 to 2002, he was a Senior Researcher with the GIS Division, Korea Institute of Civil Engineering and Building Technology (KICT), South Korea. From 2003 to 2005, he was a Postdoctoral Fellow with the Department of Geomatics, University of Calgary, Canada. From 2005 to 2006, he was the Director at Korea Geospatial Information and Communication (KSIC), South Korea. Since 2007, he has been a Professor with the Department of Drone and GIS Engineering, Namseoul University, Cheonan, South Korea. He is the author of three books and more than 106 articles. His research interests include photogrammetry, computer vision, and GIS.

GIHONG KIM was born in Incheon, South Korea, in 1973. He received the B.S., M.S., and Ph.D. degrees in civil engineering from Yonsei University, Seoul, in 2004.

From 2004 to 2005, he was a Researcher at Korea Institute of Civil Engineering and Building Technology. Since 2005, he has been a Professor with the Department of Civil and Environmental Engineering, Gangneung–Wonju National University, South Korea. He is the author of more than 90 articles. His research interests include GIS data processing, satellite image processing and applications, and disaster prediction and analysis. He is a Senior Editor of the *KSCE Journal of Civil and Environmental Engineering Research*.