Remote Sensing
Foundation models of computer vision for river detection

Emma Gauillard
Maya Janvier

March 2024

1 Introduction

River and water bodies detection is a crucial task in remote sensing, as it can help in the development of infrastructure projects or understanding the impacts of changes in hydrology on the environment. Monitoring floodings or hydrological droughts is of particular interest to the implementation of disaster risk management measures.

With the recent launches of Sentinel-1 (2014-2016) and Sentinel-2 (2015-2017), a lot of **high resolution data** is available and allow for more data-driven approaches. Wieland et al. [1] recently released a very large dataset of images from both satellite for this specific task.

This quantity of data authorizes the use of foundation models, that is to say large models trained on broad data such that it can be applied across a wide range of use cases. The objective of this project issue the foundation models learnt on natural images such as the recent Segment Anything Model (SAM) by Faceboook [2] and exploit them on the downstream task of **river detection** using the database provided in [1].
In this project, we implemented two methods using SAM : one **unsupervised method** exploiting prompts, while the other entailed **fine-tuning SAM** on our dataset.

Github You can find our implementation on Github: https://github.com/EmmaGau/projet_remote.

2 S1S2-Water Dataset

Automated water segmentation from satellite images is a famously challenging task because of the very structure of water bodies, which is very diverse in terms of reflected signal, size, and shape. The rivers' structure can be at fault as well: its **fractal banks** make resolution of highest importance when it comes to segment them.

The remote sensing nature of data also have its own challenges. Many vehicles present in the water can **modify the water bodies appearance** in the satellite images (sediment, ice, vegetation, boats). Another challenge is the presence of **clouds** and changes in **meteorological conditions** during the acquisition of images. Misclassifications are usually related to similar backscatter reflectance (sand, concrete), or increases in watersurface roughness due to strong winds that prevent specular reflection.

Wieland et al. [1] used the new amount data from Sentinel-1 (S1) and Sentinel-2 (S2) satellites to build a **robust dataset for automated water segmentation**. It follows recent guidelines for the construction of remote sensing benchmark datasets as much as possible. To get as much **diversity** as possible, a **stratified random sampling** was performed to get different climatic, atmospheric, and land cover conditions, spanning across all seasons as well.

This dataset is composed of **65 scenes** of resolution 100km × 100km,these scenes all include water areas. The class distribution in the training data is 1 (water) to 7.5 (background).

- The **Sentinel-1** satellites are equipped with a synthetic aperture radar (SAR) C-band sensor, which captures data in **VV and VH polarizations**. This sensor is adept at **penetrating through clouds and operates day and night**.
- The Sentinel-2 satellites are outfitted with a multispectral instrument that captures data across several spectral bands, including the visible (**RGB channels**), near-infrared (**NIR**), and short-wave infrared regions (**SWIR 1 and 2**). S1 images have a slightly higher resolution than S2 images.

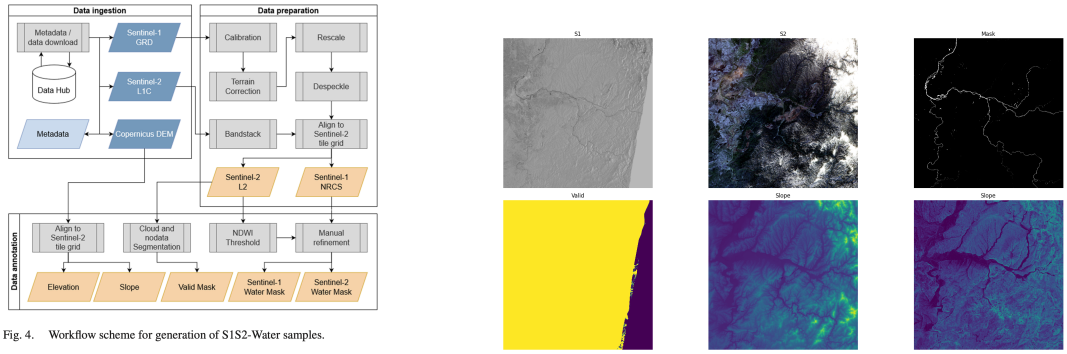


Fig. 4. Workflow scheme for generation of S1S2-Water samples.

(a) Generation of dataset

(b) Sample of dataset

Figure 1: Dataset

Figure 1 (a) shows the workflow used to generate the data samples (b).

- The ground truth is build based on a **Normalized Difference Water Index (NDWI) threshold and manual refinement**.
- A valid mask indicating the valid parts of the images is provided, excluding clouded areas. In addition, the elevation and the slope of the considered areas are given.
- The dataset comes along with a benchmark (IoU, recall precision), leveraging Convolutional Neural Networks architectures (**CNN**).

Wieland et al. also provided a Python package to tile the samples into smaller patches before feeding them to models. A tiling of S1S2-Water into **256 × 256 pixels patches** results in a total of 50 000+ patches for training and 25 000+ patches for validation and testing respectively.

3 SAM

The Segment Anything [2] paper introduces a **foundation model for segmentation**, composed of three main components: a promptable segmentation task, a segmentation model (SAM), and a data engine for collecting a large dataset named SA-1B.

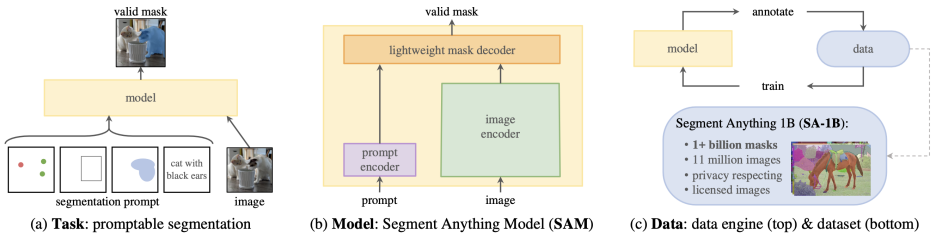


Figure 2: three main components of SAM paper

The paper introduces the concept of promptable segmentation, a task where the model generates segmentation masks based on prompts provided. This allows SAM to **output impressive zero shot performances**: it has the ability to solve a range of downstream segmentation problems on new data using **prompt engineering** (predictor mode) or non-supervised segmentation working directly on the images (generator mode).

SAM model is composed of three interconnected components: an image encoder, a prompt encoder, and a mask decoder.

- The image encoder uses a **pre-trained Vision Transformers** adapted to handle high-resolution inputs efficiently.
- The prompt encoder is adaptable to various types of prompts, be it sparse (points, boxes, text) or dense (masks). The model can handle ambiguity in prompts, ensuring that even when faced with multiple interpretations, it generates meaningful segmentation masks.
- The mask decoder is designed to map the prompt embedding, the image embedding to a mask. The decoder block inspired from a transformer block, combines **prompt self-attention and cross-attention mechanisms in 2 directions** (prompt-to-image and vice-versa), it efficiently translates input embeddings and prompts into precise segmentation foreground masks.

The predictor model is outputing multiple masks for a single prompt. The 3 mask outputs adress the issue of prompt ambiguity. The model predicts an estimated confidence score for each mask.

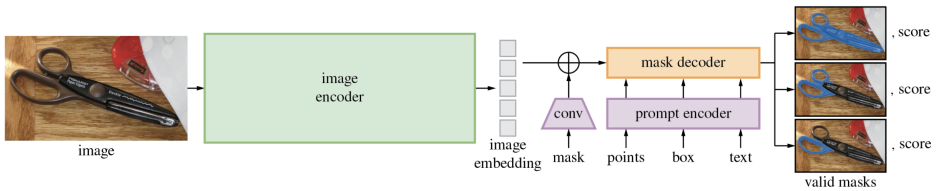


Figure 3: Sam architecture

4 Methods

4.1 Zero shot learning

As described in 3 , SAM can adapt to diverse downstream tasks by **engineering appropriate prompts**. Therefore, our first idea was to create an automated segmentation of river, using SAM in zero shot learning.

Preliminary experiments (see Figure 4) on Sentinel-1 **ruled out the generator mode** of SAM as it needed to much post-processing. It also showed us the importance of negative and positive prompts when using the predictor mode.

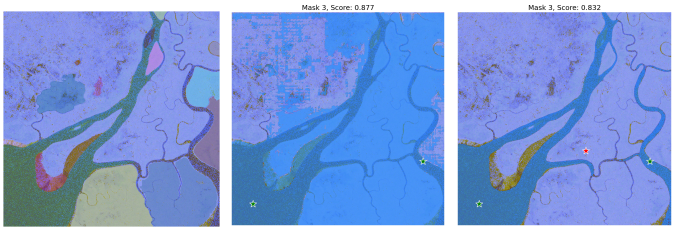


Figure 4: From left to right - Generator mode, Predictor with only positive prompts, Predictor with positive and negative prompts

The idea is then to find a suited **prompt model** for both S1 and S2 samples, as described in the method scheme (Figure 5). As for the baseline models provided in the S1S2-Water benchmark, we **worked on cropped images** for quicker inference, on patches of 2000x2000 for S1 images and 2000x2000 for S2 images.

For both data types, we proceed the same way:

- We build a first "naive" water mask to get **coarse distributions of water bodies** (section 4.1.1).
- From these masks, we can determine positive and negative points to give to SAM predictor, using the *majority class determination* algorithm (see section 4.1.2 and algorithm 1)).

4.1.1 Coarse segmentation with mathematical morphology

This section describes the specific procedures of the data types (S1, S2) leveraging **morphological operations** [3] to get these coarse segmentations. Indeed, rivers present shapes similar to eye vessels, for which many morphological filters have been developed such as the Frangi filter [4]. All morphological operations used come from skicit-image [5].

S1 S1 data has only 2 channels: VV and VH polarizations. We decided to work on VH (higher contrasts) to build the naive binary mask. After applying the Frangi filter, we do a local mean with a disk of diameter 5, and finally use the Li threshold [5] to get the final binary mask (see Figure 6).

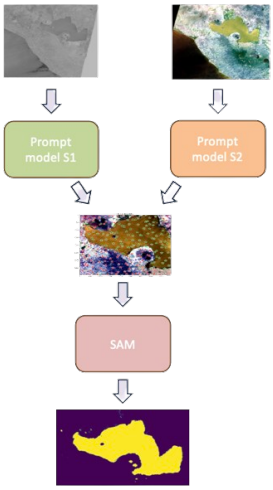


Figure 5: Method Scheme

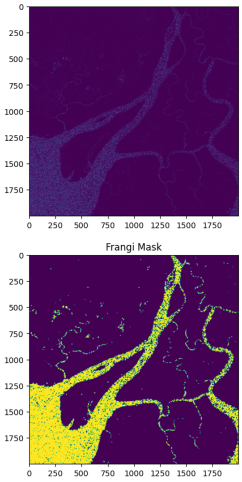


Figure 6: S1 coarse segmentation process VH channel and associated Frangi mask

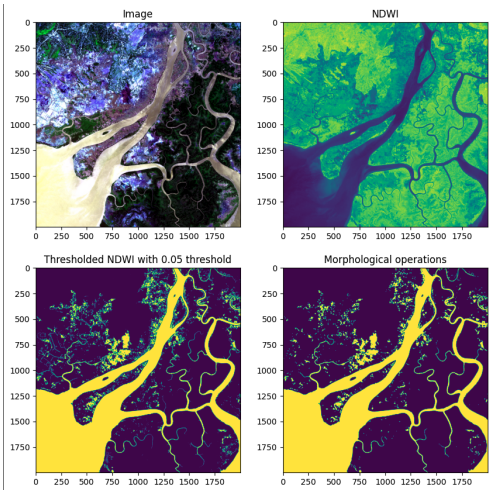


Figure 7: S2 coarse segmentation process Subsequent steps (left to right and up to bottom)

S2 For reminder, S2 is composed of 6 channels : Blue, Green, Red , NIR, SWIR1, SWIR2. This large set of channels is of great help in building a naive mask. An index is of particular interest for us, the Normalized Difference Water Index (NDWI). Using Green and NIR channels, the NDWI **is used to monitor changes related to water content in water bodies.** It is based on the principle that water absorbs and scatters light differently than other materials, such as soil and vegetation.

NDWI

$$NDWI = \frac{(X_{\text{green}} - X_{\text{nir}})}{(X_{\text{green}} + X_{\text{nir}})}$$

- Water bodies tend to absorb green light and reflect near-infrared light strongly.
- Non-water features, such as soil and vegetation, reflect green light and absorb or scatter

near-infrared light

Water bodies typically have low values due to high reflectance in the NIR band and lower reflectance in the green band.
This index is used to obtain a first mask with a manual thresholding. Morphological operations (opening, closing) are then applied to get rid of potential noise. Figure 7 illustrates the process of the coarse segmentation.

4.1.2 Prompt model

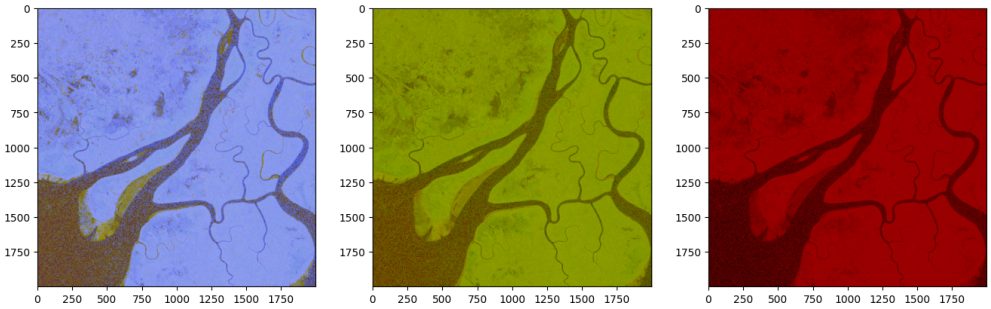
Using these coarse binary masks (4.1.1), we can finally determine prompts for SAM predictor. We used a simple approach with the *majority class determination* algorithm below (1).

Algorithm 1 Majority Class Determination

```
Get height and width of the image
Initialize empty lists for points and labels
for i = 0 to height step window_size do
    for j = 0 to width step window_size do
        Get crop of size window_size * size window_size at position [i,j]
        Determine the class that has the majority of pixels in the crop
        Randomly select a point from this majority class in the crop
        Update points and labels consequently
    end for
end for
return points, labels
End Function
```

4.1.3 Inputs

SAM needs a 3 channels image as input, preferably in RGB channels. This is possible for S2 data, however S1 data only has 2 channels. We tried many combinations to **create a third channel**, the best one being using VV, VH and the naive mask as RGB channels (Figure 8):



(a) VV, VH, VV - VH bands (b) VV, VH, Frangi-binary (c) VH, Frangi-binary x2

Figure 8: Channel combinations

4.2 Fine tuning SAM

4.2.1 Context

To leverage all of the potential of SAM, our second approach focused on fine tuning the model on our dataset. As done for the baselines of [1], it is best to fine-tune the model only on one type of data. We **worked on the S2** as it was more adapted to SAM with its RGB channels.

Fine-tuning the whole SAM is demanding **a lot of computational resources** as the model has **94.7 M parameters** and with models like this, we need a large amount of data to fine tune it.
Hence, we atempted fine-tuning solely **the decoder**. Given that the image encoder is capable of extracting robust features relevant to our specific task, we focused on tuning the decoder to generate precise masks tailored to our objectives.

4.2.2 Experiments Set Up

Data

We fine-tuned the model on the **'part 1'** of S1S2-Water. We created patches of 256x256 using the baseline code provided by github [**?**], as outlined in Section 2, to prepare our training data. Our training set is composed of 8 S2 scenes, amounting to 3290 samples, and our test dataset is composed of 4 scenes, amounting to 1529 samples. We used a batch size of 8.

We experimented with two different input configurations. Initially, after observing the impressive results of basic segmentation using the Normalized Difference Water Index (NDWI) (see Figure 7), we tried to improve performance by inputting either preprocessed **RGB channels** or the **R-NWDI-B channels** to our model. The latter consists in replacing the green channel with the NDWI index, which is able to capture reflections from water bodies.

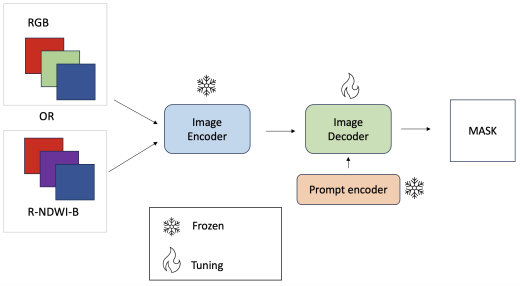


Figure 9: Model description

Model

The idea is to freeze the prompt encoder and image encoder, and solely tune the decoder (see Figure 9). Since the mask encoder produces binary masks, we optimize the criterion using Binary Cross-Entropy (BCE) Loss.

Even when training only the decoder, 10 epochs took 8 hours to train on an Nvidia A100. Fine tuning SAM involves significant computational costs.. We expect this approach to yield better results than zero-shot prediction.

5 Results

5.1 Qualitative zero-shot results on S1

Figure 10 shows the result of our pipeline on S1 data. The results seem satisfying when both river and land are present in the image. However, we observe that when there is only land with structure such as fields, our prompts are inefficient as SAM still relies heavily on the image structure. We also see that the third mask may not always be the best depending on the textures of the image.

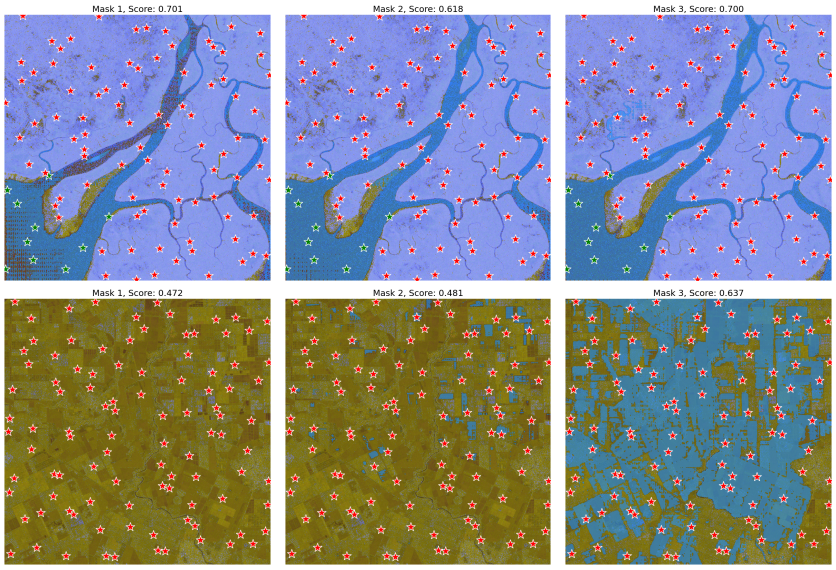


Figure 10: S1 results on river/land landscape (top row) and land (bottom row)

5.2 Qualitative zero-shot results on S2

We can observe qualitative results on patches measuring 2000×2000, SAM manages to recover the riverbed quite accurately (Figure 11). However, when using SAM on patch and then reconstructing the full image, we see that zero-shot prediction is not performing as great, the **naive NDWI mask being much more accurate** (Figure 12).

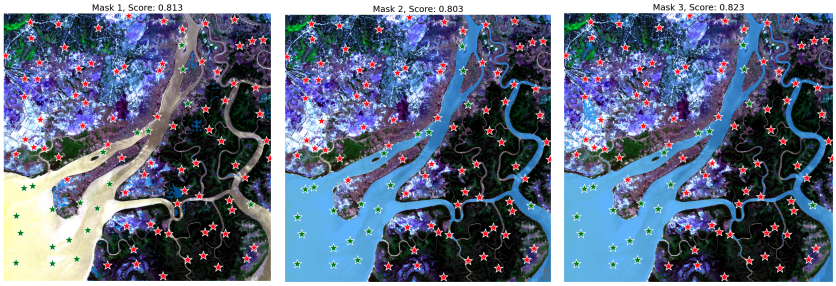


Figure 11: Masks outputs on S2 patch, using our prompt model

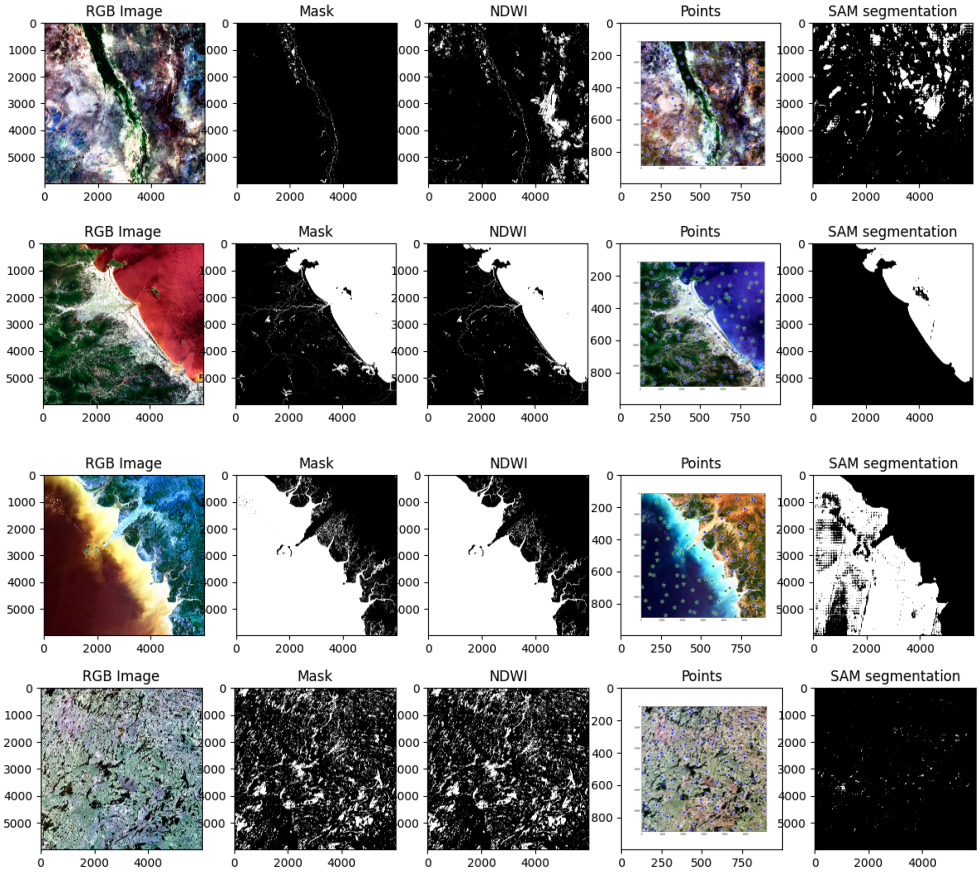


Figure 12: River Segmentation on the whole S2

5.3 Fine tuned SAM on S2

We assessed SAM performances (IoU) on 256 × 256 patches within the scenes measuring 10000 × 10000. Therefore, **we cannot compare directly** our results to others (precision, recall) as they were computed on the whole scenes.

SAM with NDWI instead of the green channel performs best, as this feature helps enhancing water bodies. We reach the same performances as the NDWI threshold. Further experimentation with tuning the image encoder to learn a **latent representation of R-NDWI-B** could be beneficial for improving performance.

We observe the training loss of SAM NDWI decreasing during the epochs in Figure 15, showing that our model is indeed learning. We only manage to train for 21 epochs, as it **already took 24h to train**. We believe more epochs will keep increasing the performances. We can see some of our best masks using SAM RGB in Figure 13 and using SAM NDWI in Figure 14.

We see that the masks lack smoothness, as SAM operates pixel-wise. SAM-RGB performs better on large water bodies whereas SAN-NDWI is better on thiner ones.

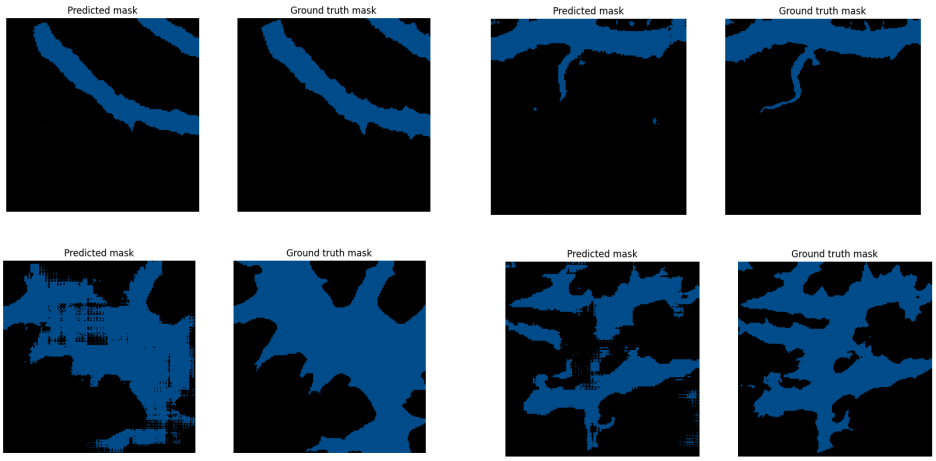


Figure 13: Masks output on some batches, SAM RGB

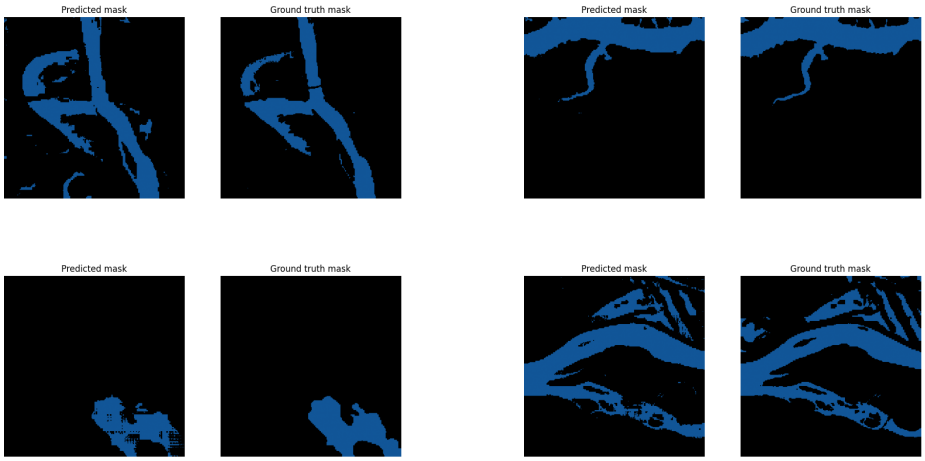


Figure 14: Masks outputs on some batches, SAM NDWI

Comparison As [1] provided a benchmark along with the dataset, we can compare our methods with the CNN baseline they developed.

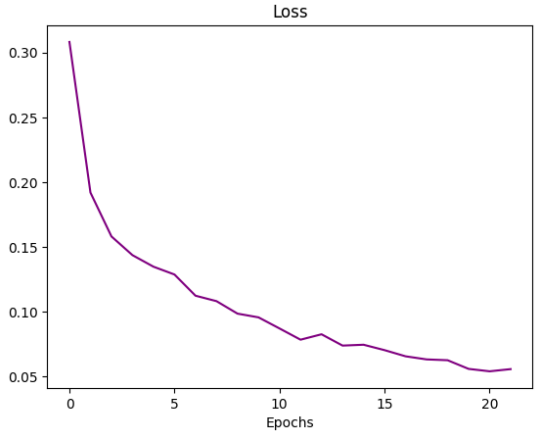


Figure 15: Training loss SAM (NDWI)

Figure 16: Quantitative Results

Method	**IOU**	**Precision**	**Recall**
NDWI	**0.80**	0.83	0.95
Zero shot SAM	0.40	0.53	0.41
Tuned SAM RGB (batch)	0.796	//	//
Tuned SAM NDWI (batch)	0.80	//	//
CNN	**0.94**	0.99	0.95

6 Conclusion

All in all, the **zero-shot approach is not precise enough**. We could improve it with **less but more accurate prompts**, as SAM actually only deals with a limited number of prompts. Another idea is to play with the different scales like CNNs do, and produce predictions at different scales that we unify using boolean operations on the masks, as SAM performs better on smaller images.

On the other hand, SAM performed better when fine-tuned but still less than the CNN baseline. It was **really hard to train because of its large number of parameters**. To tackle the issue of heavy model, we could have used the image encoder as a backbone for river segmentation task: exploiting the latent representation of our images generated by the image encoder, and pass them in a **lighter decoder** such as UNet one. We also thought about using **Lora to fine-tune SAM** [6] at lower cost.

In the end, the simple NDWI threshold was the quickest and best approach we implemented and represents a strong non-supervised baseline. We believe SAM could yield better performances with longer training. Indeed, due to the lack of resources, we couldn't train SAM decoder more than 22 epochs, but **tuned SAM seems to output very accurate results**.

Github You can find our implementation on Github: https://github.com/EmmaGau/projet_remote.

References

[1] Marc Wieland, Florian Fichtner, Sandro Martinis, Sandro Groth, Christian Krullikowski, Simon Plank, and Mahdi Motagh. S1S2-water: A global dataset for semantic segmentation of water bodies from sentinel- 1 and sentinel-2 satellite images. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 17:1084–1099, 2024.

[2] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything, 2023.

[3] Jean Paul Frédéric Serra. Image analysis and mathematical morphology. 1983.

[4] Koen L. Vinc Alejandro F. Frangi, Wiro J. Niessen and Max A. Viergever. Multiscale vessel enhancement filtering. *Medical Image Computing and Computer-Assisted Interventation — MICCAI'98 Lecture Notes in Computer Science 1496/1998: 130*, 1998.

[5] Scikit-image. https://scikit-image.org.

[6] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *CoRR*, abs/2106.09685, 2021.