IMAGE PROCESSING
FOR EARTH OBSERVATION

Historical Forest Mapping in the Swiss Alps

GOLAZ Cyril	cyril.golaz@epfl.ch
HE Shengyu	shengyu.he@epfl.ch
SUN Haoxin	haoxin.sun@epfl.ch
WU Yujie	y.wu@epfl.ch

Master Course ENV-540 Autumn 2023

Abstract

In this project, we explore deep learning methods for the semantic segmentation of forest cover in the Swiss Alps, a task critical for environmental monitoring and forest management. To address challenges such as diverse quality and class imbalance in initial aerial images, we employ data preprocessing techniques like colorization and resampling. Three processed datasets are generated based on the original images in different ways: *Gray*, *Deoldify_part*, and *Deoldify_all*. We also compare four deep learning models: U-Net, DeepLabv3, SegFormer, and SAM, analyzing their performance in this forest mapping task. Our results demonstrate that SegFormer achieves the highest accuracy among these models, largely due to its robustness in generalizing from imperfect data. Meanwhile, we investigate the varying effectiveness of different data preprocessing methods, providing insights for choosing optimal deep-learning approaches and data processing techniques in similar segmentation tasks.

1 Introduction

The Swiss forests have been intensively exploited in the past decades. The Confederation, through the "Forest Biodiversity" strategy supports the creation of natural forest reserves, which are meant to become old-growth forests over decades. Monitoring tools are required to observe the impact of such reserves and to put deforestation under surveillance. The observation of deforestation is particularly visible in the Swiss Alps. Indeed, in this region, major changes are driven by different factors such as climate change, land use, and natural disturbances [1].

Preserving forests is crucial to fight against climate change and to avoid massive biodiversity loss. A better partition of lands and forests exploitation are required and monitoring tools are of great help. In Switzerland, many aerial images have been captured over the whole country, offering the opportunity to map forests across time and space.

The field of image segmentation using deep learning has seen significant advancements, particularly in the context of environmental monitoring and disaster management. The relevance of this field to our project on forest segmentation is profound, given the shared emphasis on analyzing aerial images to extract meaningful insights.

In this project, we will explore deep learning methods to perform semantic segmentation of the forest cover, using the captured aerial images. For this, we will train different models and compare the produced outputs. The models we will use are: U-Net, DeepLabv3, SegFormer, and Segment Anything Model (SAM). In order to assess and compare their performances, we will use different metrics: training time per epoch, overall accuracy, user accuracy, producer accuracy, and intersection over union (IoU).

The report will be structured as follows: First, we will discuss the related work, more precisely some data processing that is required and the methods that we use. After this, we will present the dataset that we use and the data preprocessing that we apply to get better results. We will then present the different deep learning methods we apply and their respective architecture as well as their training details. Finally, we will present the results we obtain for each model and compare them.

The GitHub repository for our project can be accessed at the following link: `https://github.com/HaoxinSEU/IPEO-Project`. In this repository, you will find the data preprocessing code, processed dataset, and the implementations of our models.

2 Related work

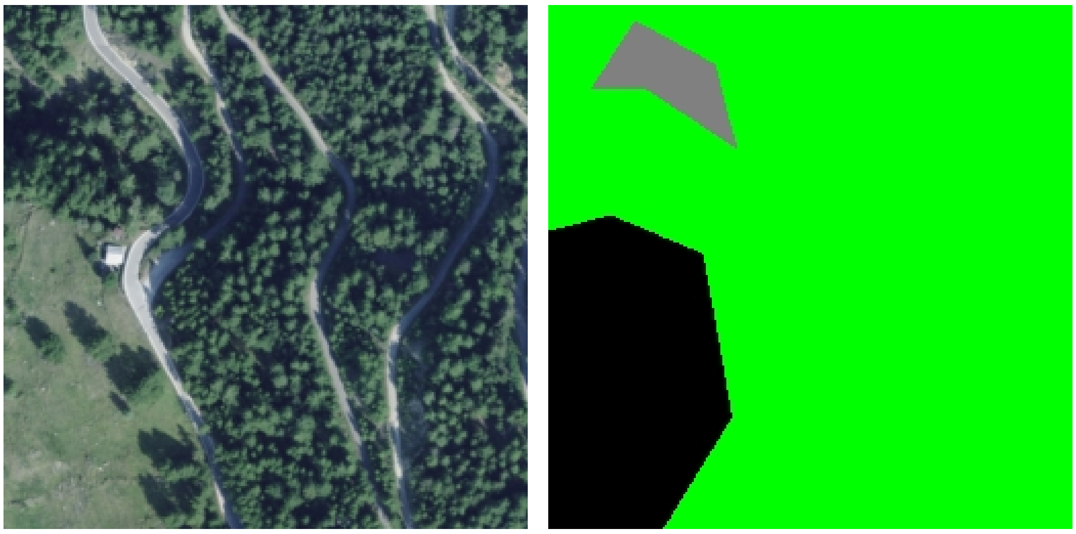
2.1 Related data processing approaches

Before training the images, we discovered the imbalance between the train and test data. The forest region is far fewer than the non-forest. Besides, many images have only one class, either forest or non. What's worse is that there is a third class with no data, increasing the task's difficulty.

To address this problem, Johnson et al. [2] reckon that there are three categories of methods to address class imbalance: data-level techniques, algorithm-level methods, and hybrid approaches combining both data and algorithmic methods. In terms of data-level techniques, Shorten et al. list the possible ways of combating imbalanced datasets such as transforming the geometry of images, altering color spaces, applying kernel filters, combining different images, transferring styles through neural networks, and employing meta-learning strategies [3].

2.2 Related methods

High resolution and a large number of earth images come out with the improvement of remote sensing techniques. Segmentation of different earth object classes has become a hot topic. In the past,



(a) Input image. (b) Ground truth (segmentation mask).

Figure 1: An example pair of an original image and the corresponding ground truth, where the forest is green, non-forest is black, and no-label is gray.

traditional machine learning methods like Random Forest are used. Recently, due to the success of deep learning models in computer vision fields, it's becoming more and more popular to apply deep learning in semantic segmentation in earth observation images.

Long et al. [4] build upon Fully Convolutional Networks (FCN) architecture for semantic segmentation. The model adapts classification networks with efficient inference and learning. U-Net is another popular image segmentation model, particularly effective for medical image segmentation, and has been adapted for high-resolution remote sensing imagery. Pan et al. [5] demonstrate a U-Net based segmentation and classification method for urban village. Adapted for remote sensing, SegNet has been used for applications like road network extraction. An encoder and decode network is embedded in the upgraded fully convolutional neural network by Kendall et al. [6]. DeepLab's capability with atrous convolutions is beneficial for analyzing diverse spatial contexts in remote sensing images. An example can be found in [7]. A simple and powerful semantic segmentation is called SegFormer [8]. It contains a hierarchical transformer encoder and multi-layer perception (MLP) decoder which could efficiently extract multi-scale features. SAM (Segment Anything Model) is a recent development in the field of computer vision, particularly focused on image segmentation. Osco et al. propose a versatile capability in segmenting remote sensing imagery, leveraging the model's zero-shot and one-shot learning abilities [9].

3 Dataset

In this project, we use the historical forest images in the Swiss Alps as the dataset. It contains 2882 aerial images and their semantic segmentation results (i.e., masks) at 1 m resolution. The dataset has already been split into train, validation and test sets, and each set has 1460, 820, and 602 images respectively.

The segmentation masks are per-pixel and thus in the same size as the original images. There are two classes in the ground truth, namely forest and non-forest. The class forest means that the pixel belongs to a forest, while the non-forest class stands for all objects that are not forest, for example, grassland, road, rock and so on. Besides these two classes, some pixels are not labeled in these images, so they are treated as non-labeled in the ground truth masks. An example pair of images and the corresponding mask is shown in Fig. 1.

The dataset is a historical recording of the forest from 1946 till 2020, thus aerial images are captured by different sensors. As a result, these images are quite different from each other. For example, some images are grayscale, while others are RGB. And they may have different sizes such as 256 $\times$ 256, 256 $\times$ 232, 232 $\times$ 232, etc. This makes our task more challenging, as deep learning models usually require unified inputs. Therefore, instead of inputting the original images to our models, we first do some preprocessing on this dataset, then feed the processed images to do the training and inference with models.

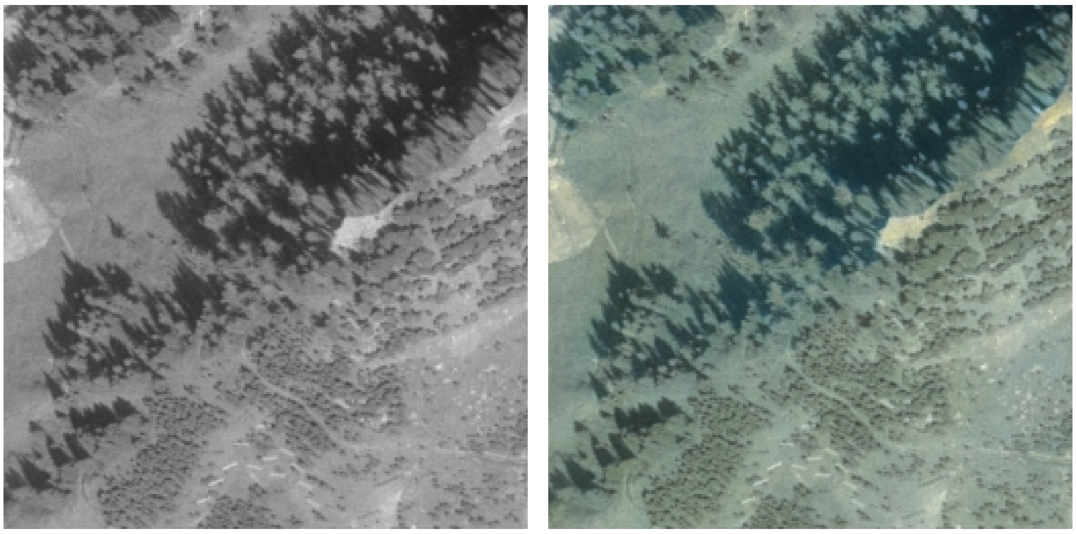

(a) Original grayscale image. (b) Colorized image.

Figure 2: An example pair of an original grayscale image and the colorization result by Deoldify.

4 Data preprocessing

4.1 Reduce non-labeled pixels

Since non-labeled pixels affect the training process and sometimes lead to unstable results, limiting the number of pixels with no ground truth is always beneficial. It turns out that in the training set, 63 images have more than 25% of their pixels non-labeled. Considering that 63 is less than 5% of the total size of the training set, not using them in the training would not affect us too much. So our first step is to directly drop these 63 images with too many non-labeled pixels.

4.2 Unify input images

The next step is to unify the number of channels for each image. Since most computer vision models expect 3-channel images as the input, we convert all grayscale images to 3-channel RGB images. There are multiple ways to do this. The simplest method is directly converting these grayscale images to 3-channel, but still gray. This traditional image processing method is fast and efficient. However, as forests are always green, while other objects may have different colors, meaning that the color can be an important feature for segmentation, it's intuitive to keep the images' color as much as possible. As a result, we would colorize the grayscale images. In this project, we use Deoldify [10] to do the colorization, which is a DL-based model and can extract the information and generate colors for gray images automatically. So our second method is to run Deoldify on all grayscale images, and replace the original images with the output RGB images. An example pair of colorized images is shown in Fig. 2. While in the first two methods, RGB images are not changed, the operation on gray images may introduce some bias [11], which makes the distribution of the resulting dataset deviate from the original dataset. Then in our third method, we decide to apply Deoldify on both RGB images and grayscale images, so that the original pattern is preserved as much as possible. For the sake of simplicity, in the rest part of this report, we refer to the preprocessed datasets by these three methods as *Gray*, *Deoldify_part*, and *Deoldify_all* respectively. An analysis about which method is the best will be given in the following sections.

For the different image sizes, there are two possible ways to deal with it. The first way is to zero-padding all images to the same size, i.e. 256 × 256. Then the ground truth of the padding area is set to be non-labeled. Another way is to directly resize all images and ground truth to 256 × 256 via interpolation. In this project, we implement and test both of these two methods, then select the one that performs better.

4.3 Data augmentation

Data augmentation helps to improve the robustness of the models and avoid overfitting. So besides the preprocessing mentioned above, we also implement random vertical and horizontal flipping when loading the images during training, to increase the diversity of the training dataset.

4.4 Fight with class imbalance

A big challenge we noticed when preprocessing this dataset is that the two classes are quite imbalanced. In the training set, the number of non-forest pixels is around 2.1 times that of forest pixels. This imbalance could lead to a large gap in the performance of forest and non-forest predictions [12].

In this project, we first resample the images with more forest pixels in the training set. Resampling is one of the most commonly used methods in deep learning, which aims to have the same number of samples in each class [13]. After resampling, the ratio between the two classes is 1.2 in the training set, which is more balanced. Using different weights for imbalanced classes is another common option. This method tells the model to pay more attention to the underrepresented class, preventing it from being overwhelmed by the majority class [14]. Therefore, during the training, we also set different weights for the two classes and the non-labeled pixels in the loss function, to make the model more aware of the forest class.

The class imbalance also happens in the validation set, which may prevent us from selecting the best model during training. We can remove images with no forest pixels at all in the validation set, so that the selected model will not just focus on the performance of the non-forest class and overwhelm the forest class.

It should also be noted that these techniques may cause overfitting and harm the model [15]. Therefore, in our implementation, these techniques are evaluated first and then we decide whether to use them or not based on the model performance.

4.5 Model-specific processing

For different models, we also have specific processing on the dataset based on the model characteristic.

As a large CV model, the fine-tuning of SAM requires more data than other models. Therefore, when training SAM, a more comprehensive set of data augmentation methods is employed. Multiple transformations like rotation, flipping, Gaussian blur, and cut-out, are applied. And each transformation has specific parameters and probabilities, ensuring a diverse range of alterations. Meanwhile, to further deal with class imbalance, we don't apply this augmentation on images that only have non-forest labels. In the later experiments, we found that without the above additional data augmentation, SAM has a high risk of developing a poor performance on the forest class. Therefore, it is imperative to do this processing.

Some pre-trained models we use require that the input images' values should be in the range [0,1], and normalized by the ImageNet mean [0.485, 0.456, 0.406] and standard deviation [0.229, 0.224, 0.225] [16]. So for these models, in the data preprocessing step we also normalize all the images according to the requirement.

5 Method

5.1 U-Net

U-Net is a deep learning architecture that became popular for its performance in image segmentation. It is an encoder-decoder convolutional neural network and is extensively used in satellite and aerial imagery analysis. It is very common to use U-Net to segment objects such as buildings, roads, and vegetation. This powerful architecture is also commonly used to support urban planning, disaster response, and environmental monitoring [17].

In deep learning, it is often required to have a large amount of data, with labels. These labeled data can be hard to acquire and it can be costly to label them correctly. An advantage of U-Net is that it performs well even with a limited amount of such data [18].

With this in mind, it was evident to consider this architecture and see how well it would perform on our task.

5.1.1 Architecture

The architecture of U-Net is shown in Fig. 3. As mentioned above, the U-Net architecture is composed of an encoder and a decoder.

The encoder path is composed of convolutions and max pooling layers. The convolutions are followed by a ReLU activation function, which helps to generalize the training data. The encoder captures contextual information and reduces the spatial resolution of the input. Moreover, the encoder acts as a feature extractor and learns an abstract representation of the image. The decoder is composed of convolution layers and upsamples the inputs. The role of the decoder is to generate a semantic mask, from the abstract representation produced by the encoder. At the end of the process, a sigmoid activation function is used and the output is the segmentation mask, representing the pixels classification [19].

The U-Net architecture also uses skip connections. They connect the encoder together with the decoder. They allow to get better segmentation results, as it consider spatial information from previous layers. Moreover, the encoder uses downsampling which causes information loss. Skip connections address this issue, as they integrate local features and global context [20].

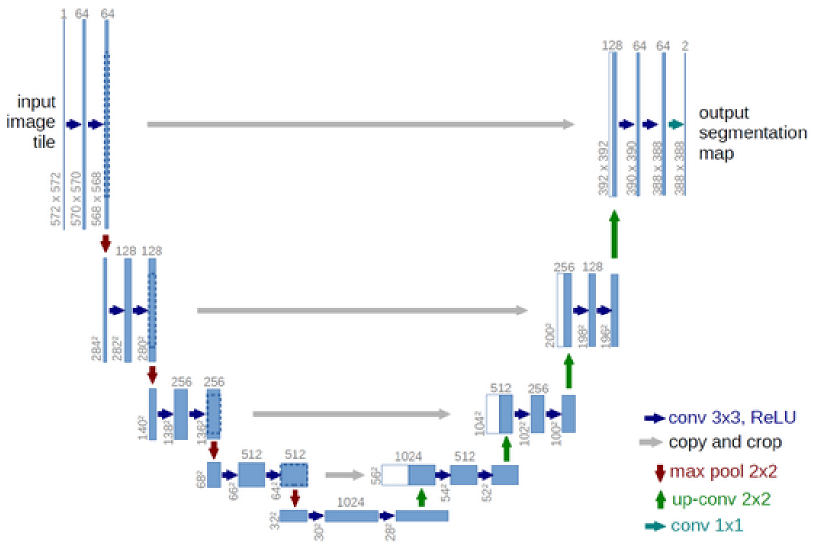


Figure 3: U-Net overview. Source: [18]

5.2 DeepLabv3

Conventional CNNs for semantic segmentation usually contain architectures like repeated pooling with a stride larger than 1, which reduces the resolution of feature maps. In our task, this loss in resolution may cause poor segmentation results as spatial information is important to us. To deal with this problem, DeepLabv3 [21] proposed a different feature extraction architecture, which is called Atrous Spatial Pyramid Pooling (ASPP). Unlike traditional convolutional layers, this ASPP module can control the feature responses' resolution without increasing the number of parameters significantly. Besides, it also enables us to detect segmenting objects with different scales by adopting various atrous rates in the convolution. Therefore, due to its better performance than traditional CNNs in semantic segmentation tasks, we implemented DeepLabv3 to solve our forest mapping task.

5.2.1 Architecture

As a common practice in computer vision models, DeepLabv3 also starts with a backbone CNN network, which is flexible and can be ResNet, MobileNet, etc. The responsibility of the backbone network is to extract hierarchical features from the input raw images. In our implementation, we choose ResNet-50 as the backbone, which is powerful and not so complex. Then the ASPP module is employed to capture information at multiple resolutions, which consists of parallel atrous convolutional layers with rates of 6, 12, and 18. The computed feature maps are concatenated together with a global average pooling result to form a multi-scale feature representation. After this, a 1x1 convolutional layer fuses these features and generates a segmentation map. Finally, a decoder module outputs high-resolution and pixel-level predictions for each class, which is our desired segmentation output. The architecture of DeepLabv3 is shown in Fig. 4.

5.3 SegFormer

Transformers is a class of deep neural network architectures, which has been increasingly applied in natural language processing and adapted for computer vision tasks, including semantic segmentation. SegFormer is a recently proposed Transformer-based framework which achieves an efficient but

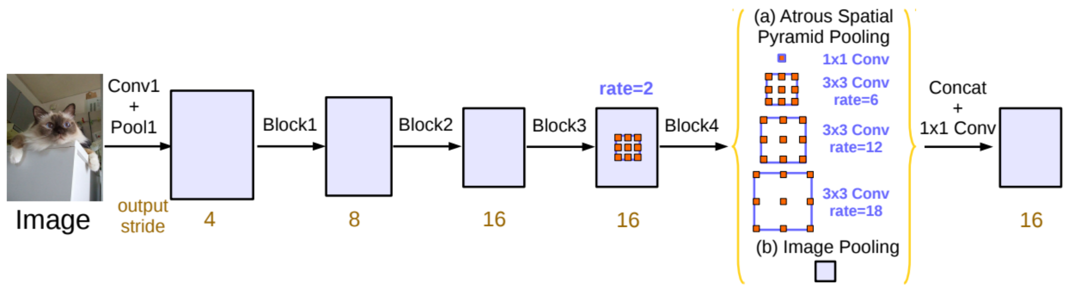
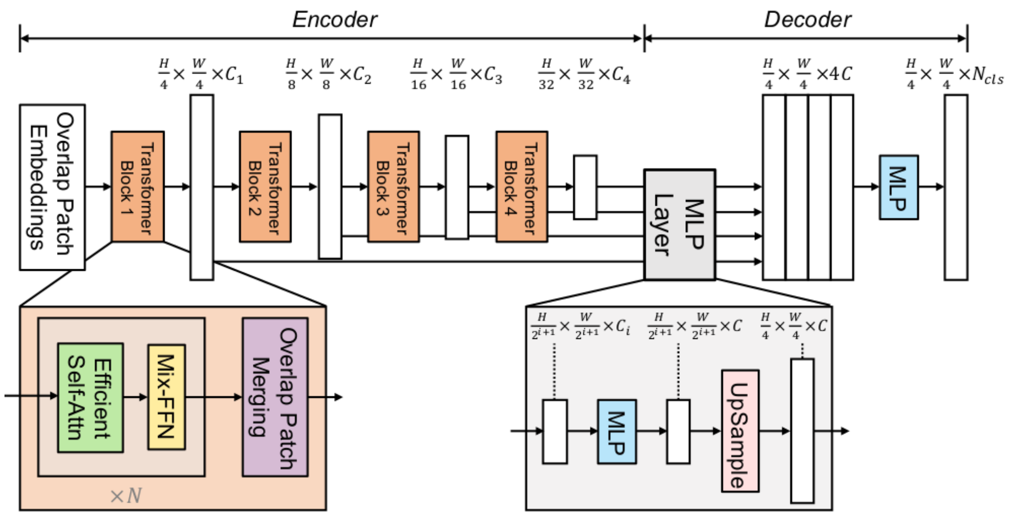


Figure 4: DeepLabv3 architecture. Source: [21]



Encoder
Decoder
$\frac{H}{4}\times\frac{W}{4}\times C_1$
$\frac{H}{8}\times\frac{W}{8}\times C_2$
$\frac{H}{16}\times\frac{W}{16}\times C_3$
$\frac{H}{32}\times\frac{W}{32}\times C_4$
$\frac{H}{4}\times\frac{W}{4}\times 4C$
$\frac{H}{4}\times\frac{W}{4}\times N_{cls}$
Overlap Patch Embeddings
Transformer Block 1
Transformer Block 2
Transformer Block 3
Transformer Block 4
MLP Layer
MLP
Efficient Self-Attn
Mix-FFN
Overlap Patch Merging
×N
$\frac{H}{2^{i+1}}\times\frac{W}{2^{i+1}}\times C_i$
$\frac{H}{2^{i+1}}\times\frac{W}{2^{i+1}}\times C$
$\frac{H}{4}\times\frac{W}{4}\times C$
MLP
UpSample

Figure 5: Segformer architecture. Source: [22]

also remarkably effective performance in semantic segmentation tasks [22]. The new model unifies Transformers has two notable features: it comprises a novel hierarchically structured Transformer encoder which outputs multi-scale features and it also avoids complex decoders. The proposed MLP decoder efficiently aggregates information from different layers, merging both local and global attention mechanisms to produce powerful representations.

5.3.1 Architecture

The SegFormer architecture, as shown in Fig. 5, is divided into two main components: the encoder and the decoder. The encoder employs the Mix-FFN (Mixed Feed Forward Network) to integrates local spatial information without positional encodings and also uses Overlap Patch Embedding to partition the input image into overlapping patches, enhancing local feature extraction. This is followed by a Transformer block to generate feature maps across multiple scales, from 1/4 to 1/32 resolution of the original image. On the decoder side, SegFormer uses an MLP that processes the fused features, which are then upsampled to generate a high-resolution segmentation mask.

The SegFormer encoder, alternatively known as the Mix Transform encoder (MiT), comes in a range of sizes, spanning from MiT-B0 to MiT-B5. Each variant maintains the same architectural design but varies in size. For our research, we utilized the MiT-B0, which has 3.8 million parameters. This model suffices to meet the requirements of our study and balances our computational resource considerations.

5.4 SAM

The last method we choose is Segment Anything Model (SAM). SAM is a model typically for object segmentation in images. It is a CNN model trained over one billion masks on eleven million images. SAM for promptable segmentation consists of three primary components: an image encoder, a prompt encoder, and a mask decoder. This model architecture is tailored for efficient, real-time performance [23]. The overall view of the SAM is shown in Fig. 6. In the next paragraph, we briefly introduce the structure of SAM.

5.4.1 Architecture

SAM employs a pre-trained Vision Transformer (ViT) minimally adapted for high-resolution inputs. This encoder is designed for scalability and powerful pre-training, processing each image once prior to prompting.

To facilitate the segmentation, SAM requires extra prompts in the training process. The prompt could be manual points on the want-to-segment place or mask which indicates different regions in the image. In this project, we have shifted our focus to detecting the boundary information of images, as opposed to working with masks. This strategic pivot is primarily driven by two key challenges we encountered. Firstly, we observed that the ground truth data is inaccurately labeled. These inaccuracies in the ground truth create substantial hurdles for effective model training, as they can lead to misguided learning and consequently, erroneous outputs. Secondly, we identified the presence of pixels in the images that has no data. When a model is trained on such incomplete or defective data, it is prone to generating false or misleading boundary information, which can significantly degrade the quality of the segmentation results. Given these challenges, we have decided to generate prompts directly from the original images. This approach allows us to circumvent the issues posed by inaccurate ground truth labels and missing pixel data. By focusing on the boundary information, which can be more reliably discerned from the original images, we aim to enhance the model's ability to perform segmentation tasks more accurately and effectively.

The mask decoder maps the image and prompts embeddings to a mask. It uses a modified Transformer decoder block for prompt self-attention and cross-attention. Following this, a dynamic mask prediction head computes the mask foreground probability for each image location.

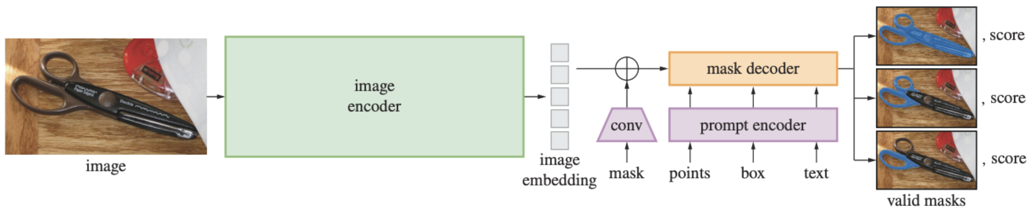


Figure 6: SAM overview. Source: [23]

The first reason for choosing this pre-trained model for this project is SAM's capability to handle various types of prompts, allowing it to accurately segment complex forest landscapes. Besides, SAM has high efficiency which could decrease the training time. Nevertheless, interactive Segmentation of the model helps to learn with an unbalanced dataset.

6 Training details

As stated before, our dataset can be considered as three categories: *Gray*, *Deoldify_part*, and *Deoldify_all*. These categories represent different methodologies in image processing as detailed in Section 4. For each one, we do training and inference independently, and then evaluate the results.

As detailed in Section 5, we strategically chose four distinct models for our image segmentation task: U-Net, DeepLabv3, SegFormer, and SAM. The configuration for each model, including optimizer, learning rates, batch sizes, and epochs, are presented in Table 1.

For U-Net, we trained the model from scratch. For DeepLabv3, we selected the pre-trained weights that were trained on a subset of COCO train2017, and then did the fine-tuning based on these. For SegFormer, we utilized the MiT-b0 version of the pre-trained model. following the guidelines provided by Hugging Face. For SAM, we used version 4.36.1 for transformers and latest version of ViT-H SAM model.

Table 1: Hyperparameters used in different models

Model	Learning rate	Optimizer	Batch size	Epochs
U-Net	1e-5	RMSprop	4	10
DeepLabv3	1e-4†	Adam	32	100
SegFormer	6e-5	Adam	8	20
SAM	8e-4	Adam	5	7

†: we use *ReduceLROnPlateau* as the scheduler, and this value here is the initial LR.

For the execution of our training and inference, we use the Kaggle Notebook environment. This platform provides us with access to the NVIDIA Tesla P100 GPU, which can enhance the computational efficiency of our segmentation tasks. The training duration for each model, measured per epoch, is displayed in Table 2.

Table 2: Training time per epoch in different models

Model	Training time per epoch (min)
U-Net	13.17
DeepLabv3	2.38
SegFormer	8.2
SAM	15.4

7 Results

In this section, we display the results of these four models. Firstly, Fig. 7 shows the predictions of each model visually. From this figure, we can have a general knowledge about the performance. U-Net appears to have a mixed performance across the different images. DeepLabv3 seems to provide a more refined segmentation with better edge definition compared to U-Net. It shows an improvement in capturing the shape of the objects, but there are still instances where the segmentation does not fully align with the ground truth. SegFormer's performance looks to be more consistent than U-Net and DeepLabv3, with a closer match to the ground truth in most cases. The edges and shapes of the segmented areas appear to be well-defined and more accurate. The SAM model shows varied results. It captures the object quite well, while for forest segmentation, there is noticeable over-segmentation.

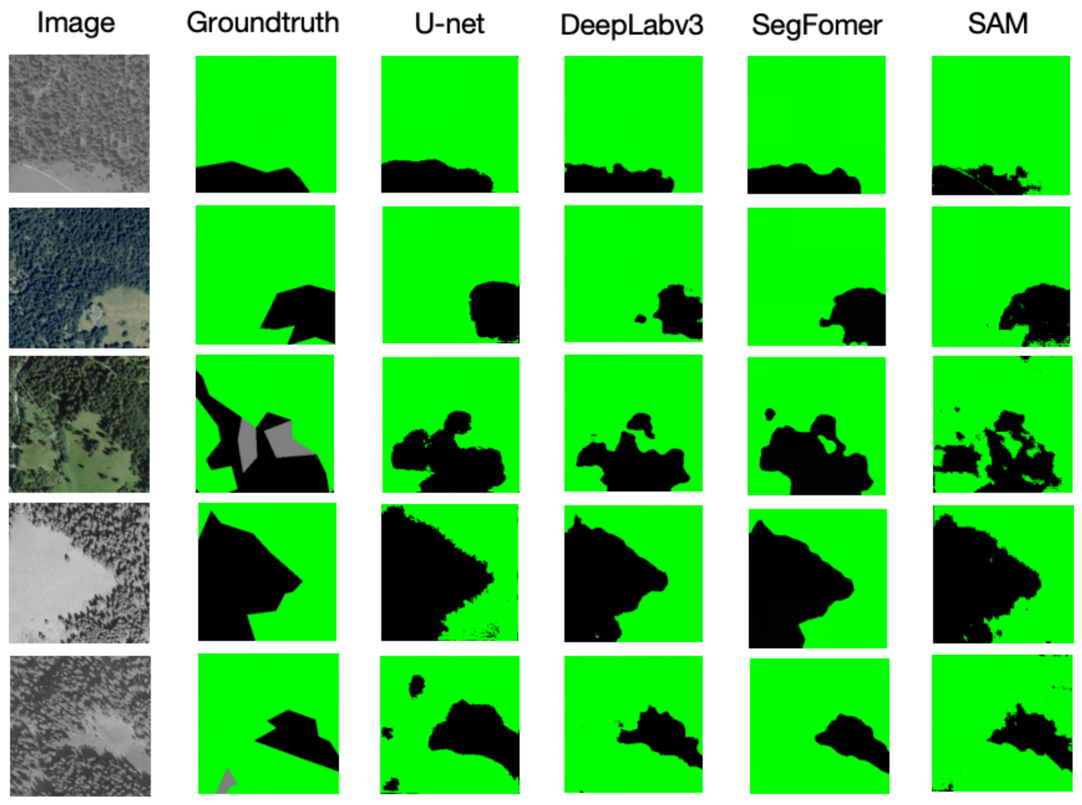


Figure 7: Comparison between image, ground truth, and models' predictions.

7.1 Evaluation metrics

Following the common practice in semantic segmentation, we use both the per-pixel accuracy and intersection over union (IoU) as the metrics to evaluate and compare the performance of different models. To be more specific, we calculate the following metrics.

- Overall accuracy (OA): It's the fraction of pixels that are correctly classified over all classes, which indicates the general performance.

- Producer accuracy (PA): It's the accuracy in capturing all the pixels of a specific class from the ground truth. And it's also known as the recall rate. For class i, it's defined as:
$$PA_i = \frac{TP_i}{TP_i + FN_i}. \tag{1}$$
- User accuracy (UA): For a specific class, this is the fraction of correct predictions among all the pixels predicted as that class. In many applications, it's also called precision. So for class i, it's defined as:
$$UA_i = \frac{TP_i}{TP_i + FP_i}. \tag{2}$$
- IoU: This metric quantifies the overlap between the predicted segmentation and the ground truth. It's one of the most widely used metrics in semantic segmentation. As the name suggests, for class i, we can calculate it as:
$$IoU_i = \frac{|Pred_i \bigcap GT_i|}{|Pred_i \bigcup GT_i|}, \tag{3}$$
where $Pred_i$ and GT_i are the regions of prediction and ground truth respectively.

The accuracy and IoU results on the test set are shown in Table 3 and Table 4 respectively. In general, they align with the visual results shown above. These different values provide insights into different aspects of the model's performance, so that they would help us to have a comprehensive understanding of the results.

Table 3: Test accuracy in different models

Model	Preprocess	OA	Forest UA	Non-forest UA	Forest PA	Non-forest PA
U-Net	*Gray*	90.52	80.29	96.59	93.32	89.21
	Deoldify_part	85.75	88.78	84.85	63.52	96.22
	Deoldify_all	89.76	87.27	90.79	79.65	94.52
DeepLabv3	*Gray*	92.94	90.20	94.17	87.44	95.53
	Deoldify_part	92.10	87.08	94.52	88.44	93.82
	Deoldify_all	92.73	86.50	95.93	91.59	93.27
SegFormer	*Gray*	**96.24**	92.43	98.15	96.14	96.29
	Deoldify_part	91.53	79.77	**99.23**	**98.54**	88.23
	Deoldify_all	96.02	93.41	97.27	94.22	96.87
SAM	*Gray*	85.63	86.08	93.65	87.26	93.02
	Deoldify_part	85.37	**94.32**	89.54	76.95	**97.70**
	Deoldify_all	86.64	92.99	69.11	86.43	97.42

Table 4: Test IoU in different models

Model	Preprocess	Forest IoU	Non-forest IoU	mIoU
U-Net	*Gray*	42.43	75.11	58.77
	Deoldify_part	40.00	74.24	57.1
	Deoldify_all	44.17	75.17	59.67
DeepLabv3	*Gray*	60.43	79.49	69.96
	Deoldify_part	59.95	77.94	68.95
	Deoldify_all	61.68	77.63	69.65
SegFormer	*Gray*	**73.13**	89.04	81.08
	Deoldify_part	59.61	76.92	68.27
	Deoldify_all	72.60	**90.97**	**81.79**
SAM	*Gray*	52.93	75.46	64.20
	Deoldify_part	51.98	76.63	64.31
	Deoldify_all	45.13	75.28	60.20

7.2 Analysis

Now we can get insights into the effectiveness of different data processing and models. For the forest class, SegFormer achieves the best IoU of 73.13, while DeepLabv3 reaches around 60, which is the second largest. SAM and U-Net get about 50 and 42 respectively. And the accuracy results show the same trend as the IoU. So we can conclude that SegFormer performs the best among these four models, while U-Net has the worst performance. Looking at the different three processing methods (i.e., *Gray*, *Deoldify_part*, and *Deoldify_all*), they show different effects on different models. For example, colorization on all images improves the performance of U-Net and DeepLabv3, but has an opposite impact on SegFormer and SAM. However, it's clear that colonizing only part of the dataset hurts all four models. A detailed analysis is given below.

7.2.1 Data processing analysis

Colorization can improve the performance of traditional models like U-Net and DeepLabv3, as the IoU when using *Deoldify_all* is better than using *Gray.* But it harms the IoU for advanced models like SegFormer and SAM. This is to be expected. First of all, the applied colorization is not perfect, we can see the color bias in the colorized images. In other words, using colorization introduces new information to the model compared to only inputting grayscale images, but this information may be inaccurate. For advanced models like SAM and SegFormer, their architectures already extract representative features very well from the input images. Thus, adding this additional colorization information provides little benefit and the inaccuracy may even hurt the model. But for smaller models like U-Net and DeepLabv3, their feature extraction can be complemented by colorization, and thus have a better performance during the inference.

Besides, if we only colorize part of the dataset, we change the original dataset distribution, and may introduce bias. In our case, when the colorization is only applied on grayscale images (i.e., *Deoldify_part*), forest pixels in colorized and non-colorized images may have a very different distribution. As a result, we see that it degrades all four model's performance.

7.2.2 Model analysis

U-Net has a simple architecture and the downsampling operations might lead to a loss of spatial information. Moreover, it takes a long time to train in the context of this project and using kaggle, we were limited in the number of epochs that we could train as kaggle only allows a limited number of hours of activity. The mentioned reasons can explain why it performed not as well as the other models.

For DeepLabv3, due to its atrous convolution structure, it can capture multi-scale contextual information, which helps the model effectively analyze both local and global contexts. Besides, its pre-trained weights make sure that it can always learn meaningful features from the input images. Thus, DeepLabv3 performs better in these metrics than U-Net. However, because of the class imbalance in the test set, we see that the performance of the forest class degrades, and the gap between the forest and non-forest classes is significant.

In our forest mapping and segmentation tasks, SegFormer has achieved the best score in prediction. Compared to traditional CNN models, it has a Transformer-based structure. This structure enables a better feature extraction and handling of long-range dependencies. Such capabilities are crucial for understanding the complex, varied textures and structures typically found in forest environments. Furthermore, given the potential inaccuracies in our ground truth, SegFormer also demonstrates an advantage. It shows a better ability to generalize from imperfect data, learning to identify patterns and features that are representative of the actual forest structure. This is particularly important in avoiding overfitting to potentially inaccurate labels, such as the precise classification of individual trees from forests.

While the SAM represents a state-of-the-art approach in image segmentation, leveraging a model trained on billions of data points, its performance in forest segmentation tasks has not reached optimal levels. This may be attributed to several factors:

- Robustness and adaptability: SAM's architecture is inherently robust, designed to handle a wide variety of scenarios and data types. Besides, SAM is the expert in detecting objects and separating them from the environment. This means it is not perfectly suitable for this project. For example, SAM would detect a green plant on the ground as forest while a forest is composed of plenty of trees.

- Ground truth accuracy: Another critical factor impacting SAM's performance in this task is the accuracy of the ground truth labeling. Inaccurate or inconsistent labeling in the training data can significantly affect the model's learning, leading to subpar performance. An example of inaccurate ground truth compared with SAM predict result could be seen in Fig. 8.

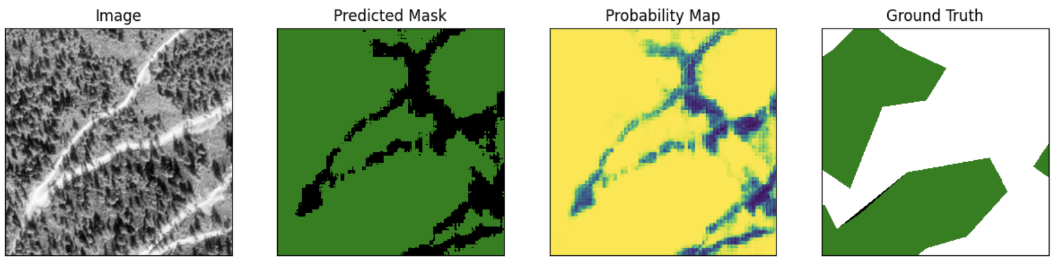


Figure 8: Original image, SAM predicted result, SAM predicted probability, ground truth.

8 Conclusion

In this report, we investigate how to use deep learning methods to do semantic segmentation on the aerial images of the Swiss Alps. To handle challenges such as diverse quality and class imbalance in the provided dataset, we employ various data processing techniques and develop three processed datasets: *Gray*, *Deoldify_part*, and *Deoldify_all*. Using these processed datasets, we train and do inference on four different deep learning models, namely U-Net, DeepLabv3, SegFormer, and SAM.

SegFormer achieves the best scores in accuracy, recording a forest class IoU of 73.13 and a mean IoU of 81.79. This performance is primarily attributed to its advanced architecture, which provides a better ability to generalize from imperfect data. U-Net and DeepLabv3 have worse performance than SegFormer, since CNN may not be as powerful as Transformers. Due to factors like its robustness and imperfect ground truth, SAM doesn't reach optimal performance, and has a relatively small accuracy in our task.

Besides, we also see that when dealing with grayscale and RGB images simultaneously, if the model is not so advanced, colorization on the whole image dataset is helpful. But when the model itself extracts features very well, there is no need to colorize, and directly using raw images can be a better choice.

References

[1] *Monitoring natural forest reserves in Switzerland — wsl.ch.* `https://www.wsl.ch/en/forest/biodiversity-conservation-and-primeval-forests/natural-forest-reserves/`. [Accessed 05-01-2024].

[2] Justin M Johnson and Taghi M Khoshgoftaar. "Survey on deep learning with class imbalance". In: *Journal of Big Data* 6.1 (2019), pp. 1–54.

[3] Connor Shorten and Taghi M Khoshgoftaar. "A survey on image data augmentation for deep learning". In: *Journal of big data* 6.1 (2019), pp. 1–48.

[4] Jonathan Long, Evan Shelhamer, and Trevor Darrell. "Fully Convolutional Networks for Semantic Segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* June 2015.

[5] Zhuokun Pan et al. "Deep Learning Segmentation and Classification for Urban Village Using a Worldview Satellite Image Based on U-Net". In: *Remote Sensing* 12.10 (2020). ISSN: 2072-4292. DOI: `10.3390/rs12101574`. URL: `https://www.mdpi.com/2072-4292/12/10/1574`.

[6] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation". In: *CoRR* abs/1511.00561 (2015). arXiv: `1511.00561`. URL: `http://arxiv.org/abs/1511.00561`.

[7] Liang-Chieh Chen et al. *DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs.* 2017. arXiv: `1606.00915 [cs.CV]`.

[8] Enze Xie et al. "SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers". In: *CoRR* abs/2105.15203 (2021). arXiv: `2105.15203`. URL: `https://arxiv.org/abs/2105.15203`.

[9] Lucas Prado Osco et al. *The Segment Anything Model (SAM) for Remote Sensing Applications: From Zero to One Shot.* 2023. arXiv: `2306.16623 [cs.CV]`.

[10] Jason Antic. *DeOldify.* `https://github.com/jantic/DeOldify`. Accessed Dec 2023.

[11] Jheng-Wei Su, Hung-Kuo Chu, and Jia-Bin Huang. *Instance-aware Image Colorization*. 2020. arXiv: `2005.10825 [cs.CV]`.

[12] Justin Johnson and Taghi Khoshgoftaar. "Survey on deep learning with class imbalance". In: *Journal of Big Data* 6 (Mar. 2019), p. 27.

[13] Guo Haixiang et al. "Learning from class-imbalanced data: Review of methods and applications". In: *Expert systems with applications* 73 (2017), pp. 220–239.

[14] Haibo He and Edwardo A. Garcia. "Learning from Imbalanced Data". In: *IEEE Transactions on Knowledge and Data Engineering* 21.9 (2009), pp. 1263–1284.

[15] Mateusz Buda, Atsuto Maki, and Maciej A Mazurowski. "A systematic study of the class imbalance problem in convolutional neural networks". In: *Neural networks* 106 (2018), pp. 249–259.

[16] Jia Deng et al. "ImageNet: A large-scale hierarchical image database". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009, pp. 248–255.

[17] Alexquesada. *U-Net: A versatile deep learning architecture for image segmentation*. Aug. 2023. URL: `https://medium.com/@alexquesada22/u-net-a-versatile-deep-learning-architecture-for-image-segmentation-2a85b52d71f6`.

[18] Raphael Kassel. *U-NET: Le Réseau de Neurones de Computer Vision*. Nov. 2023. URL: `https://datascientest.com/u-net`.

[19] Nikhil Tomar. *What is UNET?* Jan. 2021. URL: `https://medium.com/analytics-vidhya/what-is-unet-157314c87634`.

[20] S Premanand. *A Comprehensive Guide to UNET Architecture | Mastering Image Segmentation — analyticsvidhya.com*. Accessed 05-01-2024. 2023. URL: `https://www.analyticsvidhya.com/blog/2023/08/unet-architecture-mastering-image-segmentation/#h-skip-connections`.

[21] Liang-Chieh Chen et al. *Rethinking Atrous Convolution for Semantic Image Segmentation*. 2017. arXiv: `1706.05587 [cs.CV]`.

[22] Enze Xie et al. *SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers*. 2021. arXiv: `2105.15203 [cs.CV]`.

[23] Alexander Kirillov et al. *Segment Anything*. 2023. arXiv: `2304.02643 [cs.CV]`.