

VILNIUS UNIVERSITY

FACULTY OF MATHEMATICS AND INFORMATICS

ARTIFICIAL NEURAL NETWORKS COURSE

**SATELLITE IMAGE RECOGNITION TASK**

3rd Report

Author: Edvardas Timoscenka 3rd year student of Mathematics and Mathematical Applications

Vilnius

2023

Contents

[1. Images and data. 2](#_Toc151995388)

[1.1. Original Dataset Images and Masks. 2](#_Toc151995389)

[1.2. Data preprocessing. 3](#_Toc151995390)

[1.3. Data preparation for the model. 4](#_Toc151995391)

[2. Neural Networks architecture. 6](#_Toc151995392)

[3. Comparative analysis of Neural Networks training. 6](#_Toc151995393)

[4. Computational resource usage. 10](#_Toc151995394)

[5. Plans for other reports. 10](#_Toc151995395)

[6. Project code. 10](#_Toc151995396)

[7. Appendix 11](#_Toc151995397)

[8. Bibliography 11](#_Toc151995398)

# Images and data.

## Original Dataset Images and Masks.

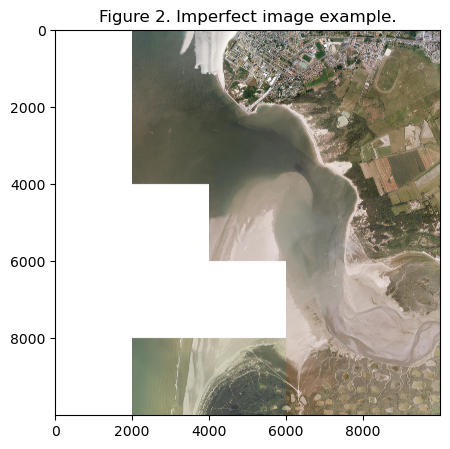
The original dataset comprised high-resolution images, each measuring 10,000 pixels by 10,000 pixels, with RGB color values. Notably, the dataset was predominantly centered around the Brest region, featuring a total of 377 images. In contrast, the Le Mans region had the lowest representation in the dataset with only 226 images.

Furthermore, the dataset included corresponding image masks, maintaining the same dimensions, but with pixel values indicating class labels. Figure 1 presents an illustrative image-mask pair.

A map of a city

Description automatically generated with medium confidence

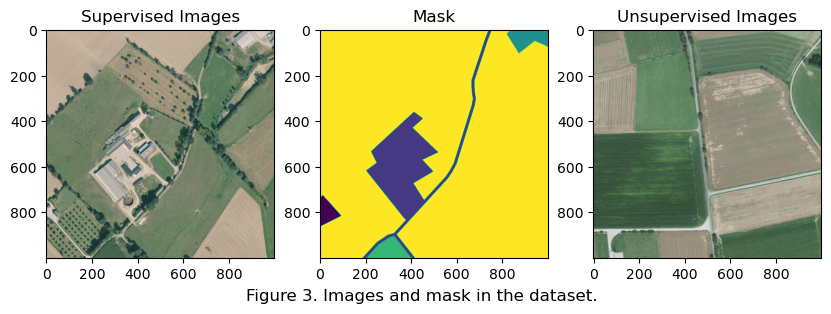
The miniFrance dataset was designed for semi-supervised learning, which implies that not every image had an associated mask. Approximately 53% of the images had corresponding masks, leaving the remainder unlabeled. The St. Brieuc region had the lowest mask coverage, with only 25% of its images labeled, while the Nice region had the highest proportion of labeled masks, encompassing approximately 77% of its images.

In summary, the dataset encompassed a total of 2,816 images and 1,490 corresponding masks. It's worth noting that some images exhibited imperfections in terms of quality, featuring missing data in certain areas, as visualized in Figure 2.

## Data preprocessing.

As outlined in the first report, the original dataset underwent modifications to enhance its computational feasibility and facilitate data manipulation. Each original image and its corresponding mask were partitioned into smaller sub-images, each measuring 1,000 pixels by 1,000 pixels, preserving the original values. Subsequently, a random selection process was employed to choose sub-images from each source image, which were then stored in the designated data repository for future use. To address variations in image quality, an additional filtering step was introduced to assess the percentage of white pixels within each sub-image.

The filter examined whether a sub-image was present in over 50%[[1]](#footnote-1) of the source images, and if not, the random selection process was repeated using the same source image. This procedure aimed to minimize the inclusion of low-quality images in the dataset for subsequent model training.

A visual representation of the results can be found in Figure 3.

A subsequent round of preprocessing was executed following the same methodology, expanding the image count to exceed 3,000 images. After the preprocessing phase, the dataset encompassed a total of X [[2]](#footnote-2) images and Y corresponding masks.

In the 2nd report the class distribution differed from the original paper, which was caused by the fact that the original paper did not include the ‘No Information’ class in their images. To solve this problem, a new filter was added, that would filter out the sub-images that contain the ‘No Information’ class. If no sub-images could be found in the original image, the image would be skipped over. This way a new dataset was created, consisting of X images and Y corresponding masks. The new dataset addresses the disparities between the created dataset and the original dataset. To ensure clarity and distinction between the datasets the initial dataset will be further called dataset\_v1. The revised dataset, which has been curated following the addition of the filter to exclude images containing the 'No Information' class, will be designated as dataset\_v2.

The distribution of classes (calculated from supervised training dataset) in both datasets can be found in Appendix Table 1.

## Data preparation for the model.

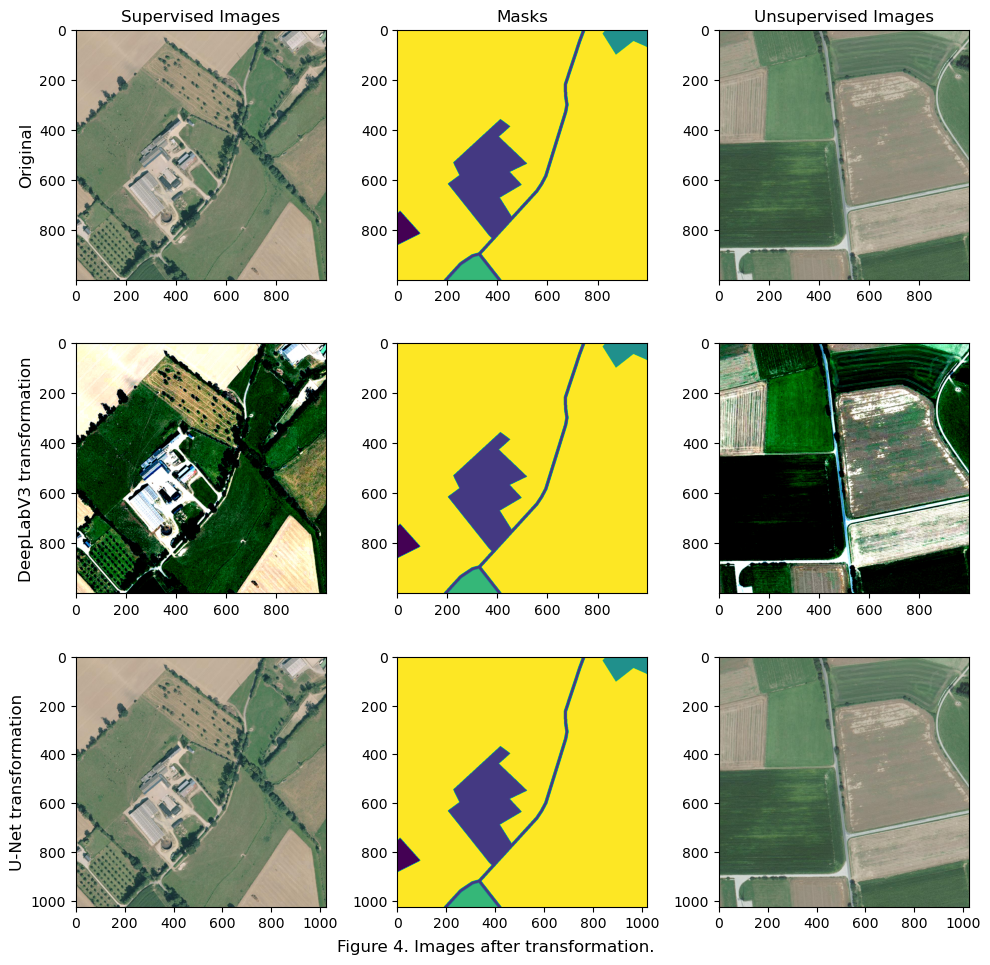
A dedicated Data Loader was created to manage both the supervised and unsupervised data streams intended for the model. This loader creates random samples from the dataset, ensuring a specified proportion of supervised and unsupervised data to be fed into the model. Moreover, the Data Loader performs transformations on the images, incorporating mean and standard deviation adjustments required for compatibility with the DeepLabV3 models and recognizing the model's expectation of input data in the form of minibatches and different shape. The mean and standard deviation for DeepLabV3 models is [0.485, 0.456, 0.406] and [0.229, 0.224, 0.225] respectively. The transformation was not performed on the mask images.

For the 3rd report Data Loader was updated with tailored image transformations to align with U-NET's architectural requirements. Because the original U-NET architecture typically relies on down-sampling and up-sampling steps that assume the image dimensions are powers of two (e.g., 512x512, 1024x1024), so the images were resized to the shape 1024x1024 pixels. All the images were transformed including masks.

In addition to the usual transformations, the dataset was split into train and validation sets, where 75% of the data would be in the train dataset and 25% in validation set. To improve accuracy and model performance, one of the ways to account for the class imbalance in the dataset, K-Means clustering was performed on whose results stratification could be performed to adjust the ratio of each class in train and validation set. The results can be seen in Figure 4.

A screenshot of a computer generated image

Description automatically generated

The results of both transformations can be seen in Figure 4.

# Neural Networks architecture.

For the 3rd report all the planned DeepLabV3 Pytorch implementations were implemented and tested. These implementations included the integration of various backbone architectures, namely ResNet-50, ResNet-101, and MobileNetV3 large. Each of these models was pretrained on a subset of the COCO train2017 dataset, which comprised 20 distinct categories.

Additionally, as planned U-NET architecture was implemented within Pytorch framework. The architecture mirrored the original papers CAFFE implementation[1].

# Comparative analysis of Neural Networks training.

The comparative analysis was performed on all the architectures that were implemented in this report using dataset\_v1. For the training process Adam optimizer was chosen over Stochastic Gradient Descent, AdaGrad and others. Because it has adaptive learning rates, is more efficient than others, and has little need for hyperparameter tuning. As a loss function Tversky Loss was chosen, because both the dataset\_v1 and dataset\_v2 have a big class imbalance as can be seen in Table 1. The parameter for the loss was 0.5. For the final evaluation in the validation set Jaccard index and Dice scores were calculated. All DeepLabV3 models were trained for 50 epochs. The training and validation graphs have an applied smoothing coefficient of 0.99 for better graph interpretability.

3.1. DeepLabV3 with ResNet-50 backbone.

The DeepLabV3 model with a ResNet-50 backbone processed dataset\_v1 in 3.23 hours, showing promising results, especially in comparison to BerundaNet implemented with SegNet and U-Net backbones[2]. The model's performance, as indicated by the Jaccard index and overall accuracy, was notably favorable, suggesting potential for further improvements with fine-tuning. On dataset\_v2, which is more aligned with the data in the referenced paper, the model excelled in overall accuracy but fell short in Jaccard index performance. Table 1 displays the comparative results on both datasets.

Table 1. Resuls of DeepLabV3 with ResNet-50 backbone

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Metric | Score | | Goal | |
| dataset\_v1 | dataset\_v2 | SegNet | U-Net |
| Jaccard Score | 16 | 20 | 24 | 25 |
| Dice Score | 50 | 65 | 59 | 58 |

Analysis of the training's learning curve suggests that extending the training epochs could enhance performance for both datasets. For dataset\_v1, decreasing the learning rate might also be beneficial to address the significant fluctuations observed during training. The learning curves for both datasets are illustrated in Figure 7.

A graph of a graph

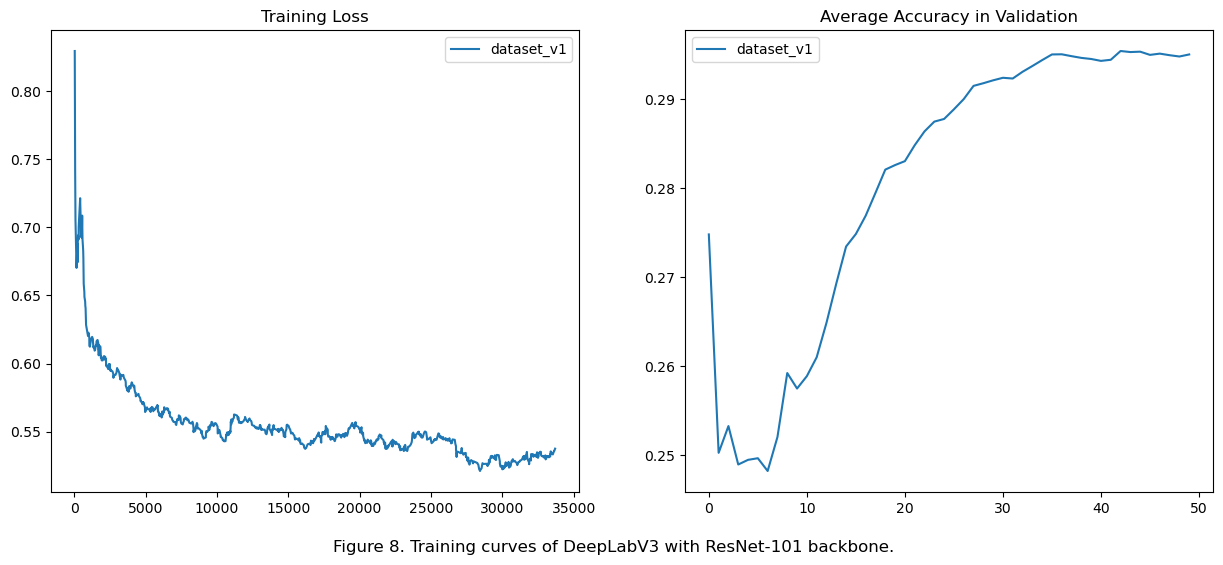
Description automatically generated with medium confidence

3.2. DeepLabV3 with ResNet-101 backbone.

The DeepLabV3 model with the ResNet-101 backbone took approximately 5 hours to run. This variant underperformed on dataset\_v1 compared to the ResNet-50 backbone version, as detailed in Table 2.

Table 2. Resuls of DeepLabV3 with ResNet-101 backbone

|  |  |  |  |
| --- | --- | --- | --- |
| Metric | Score | Goal | |
| dataset\_v1 | SegNet | U-Net |
| Jaccard Score | 4 | 24 | 25 |
| Dice Score | 29 | 59 | 58 |

The model's suboptimal performance could be due to a too-small loss learning rate and an insufficient number of training epochs. Figure 8 showcases the learning curve for this model. Currently, this model has only been tested on dataset\_v1, with no adjustments, due to time constraints.

3.3. DeepLabV3 with MobileNetV3 large backbone.

The DeepLabV3 model with a MobileNetV3 Large backbone had the shortest runtime of 2.08 hours, a characteristic of its design optimized for speed and reduced computational requirements. Despite its speed, this model performed the least effectively among the DeepLabV3 variants. The training results are available in Table 2.

Table 3. Resuls of DeepLabV3 with MobileNet V3 large backbone.

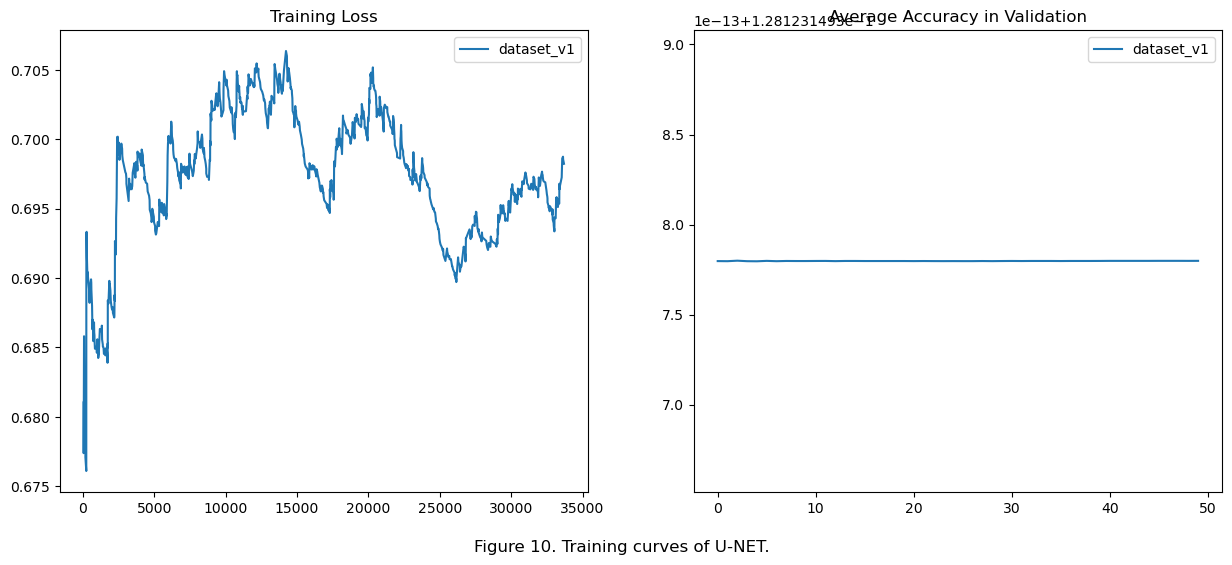
|  |  |  |  |
| --- | --- | --- | --- |
| Metric | Score | Goal | |
| dataset\_v1 | SegNet | U-Net |
| Jaccard Score | 12 | 24 | 25 |
| Dice Score | 42 | 59 | 58 |

The learning curves depicted in Figure 9, showed no irregularities, and the gradual decrease in training loss coupled with increasing average validation accuracy suggests an appropriate learning rate.

A graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of

Description automatically generated

3.4. U-NET.

 The U-NET model implemented did not meet expectations. The training loss failed to decrease over time, potentially due to an excessively high learning rate. Additionally, there were issues with the validation set calculations, as evidenced by the static validation loss and average accuracy throughout training. This could indicate problems when the model is tested on the validation dataset. Future efforts will focus on addressing these issues. The training curves are presented in Figure 10.

3.5. Conclusion.

The DeepLabV3 model with a ResNet-50 backbone demonstrated promising results on dataset\_v1 and dataset\_v2, outperforming other configurations like BerundaNet with SegNet and U-Net backbones in terms of the Jaccard index and overall accuracy. This suggests potential for further improvements through model calibration.

In contrast, the DeepLabV3 with the ResNet-101 backbone exhibited underperformance on dataset\_v1, indicating that a higher capacity model does not necessarily guarantee better results, particularly if the learning rate and epoch count are not optimally set.

The DeepLabV3 model with a MobileNetV3 Large backbone, while offering the fastest runtime, showed the least effective performance among the evaluated models. This underlines a trade-off between computational efficiency and model accuracy.

The U-NET model's inability to converge during training suggests issues with the learning rate setting and potential problems in the validation set computation. So, no conclusions as of right now can be made about this model architecture.

# Computational resource usage.

As planned before, my personal computer is used for carrying out most of the research and implementation. Through the testing model, a significant bottleneck has been identified in this approach - the GPU's VRAM. Presently, the GPU in use has 24GB of VRAM, which still faces challenges when handling the DeepLabV3 model, given its substantial size. The biggest issue arises when attempting to employ larger batch sizes as input into the model. For instance, with the batch parameter set to 5, the model demands approximately 35GB of VRAM, exceeding the available capacity. The most optimal batch size found now is 2, but there is potential for increasing this batch size through the implementation of efficient memory management.

# Plans for other reports.

The next phase of the project involves the following:

* The complete implementation of the DeepLabV3 model with Xception backbone.
* Performing unsupervised learning on the best performing model architecture.
* Fixing U-NET model.
* Fine-tuning DeepLabV3 model with ResNet-50 backbone.
* Experiment if the stratified split of train-validation would make improvements.

# Project code.

The project code can be found in the GitHub repository located here: <https://github.com/Efoks/sem_sat_img_segmentation>

# Appendix

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Class | dataset\_v1 | | |  | dataset\_v2 | | |
| % pixels | Goal | Difference |  | % pixels | Goal | Difference |
| No information | 30.3% | - | - |  | - | - | - |
| Urban fabric | 6.5% | 9.9% | -3.4 p.p. |  | 9.7% | 9.9% | -0.2 p.p. |
| Industrial, commercial, public, military, private and transport units | 4.3% | 6.5% | -2.2 p.p. |  | 6.4% | 6.5% | -0.1 p.p. |
| Mine, dump, and construction sites | 0.4% | 0.7% | -0.3 p.p. |  | 0.5% | 0.7% | -0.2 p.p. |
| Artificial non-agricultural vegetated areas | 0.6% | 1.2% | -0.6 p.p. |  | 1.1% | 1.2% | -0.1 p.p. |
| Arable land (annual crops) | 20.0% | 30.7% | -10.7 p.p. |  | 26.8% | 30.7% | -3.9 p.p. |
| Permanent crops | 0.4% | 1.3% | -0.9 p.p. |  | 0.5% | 1.3% | -0.8 p.p. |
| Pastures | 20.2% | 27.3% | -7.1 p.p. |  | 30.3% | 27.3% | 3.0 p.p. |
| Complex and mixed cultivation patterns | 0.0% | 0.0% | 0.0 p.p. |  | 0.0% | 0.0% | 0.0 p.p. |
| Orchards at the fringe of urban classes | 0.0% | 0.0% | 0.0 p.p. |  | 0.0% | 0.0% | 0.0 p.p. |
| Forests | 12.5% | 16.0% | -3.5 p.p. |  | 18.4% | 16.0% | 2.4 p.p. |
| Herbaceous vegetation associations | 1.9% | 4.5% | -2.6 p.p. |  | 3.1% | 4.5% | -1.4 p.p. |
| Open spaces with little or no vegetation | 0.8% | 0.1% | 0.7 p.p. |  | 1.0% | 0.1% | 0.9 p.p. |
| Wetlands | 1.1% | 0.7% | 0.4 p.p. |  | 1.2% | 0.7% | 0.5 p.p. |
| Water | 0.9% | 1.0% | -0.1 p.p. |  | 0.9% | 1.0% | -0.1 p.p. |
| Clouds and shadows | 0.1% | 0.1% | 0.0 p.p. |  | 0.1% | 0.1% | 0.0 p.p. |

# Bibliography

[1] O. Ronneberger, P. Fischer, and T. Brox, “U-Net: Convolutional Networks for Biomedical Image Segmentation.” arXiv, May 18, 2015. Accessed: Sep. 17, 2023. [Online]. Available: http://arxiv.org/abs/1505.04597

[2] J. Castillo-Navarro, B. Le Saux, A. Boulch, N. Audebert, and S. Lefèvre, “Semi-supervised semantic segmentation in Earth Observation: the MiniFrance suite, dataset analysis and multi-task network study,” *Mach. Learn.*, vol. 111, no. 9, pp. 3125–3160, Sep. 2022, doi: 10.1007/s10994-020-05943-y.

1. For the third report the percentage of white was changed from 90% (2nd report) to 50% for better results. [↑](#footnote-ref-1)
2. The image count has changed from previous report, see the footnote 1 on why. [↑](#footnote-ref-2)