Best practices for producers and providers of image sets and image analysis applications in aquatic sciences

iMagine Deliverable D3.4

01/10/2024

Funded by the European Union

iMagine receives funding from the European Union's Horizon Europe research and innovation program under grant agreement No. 101058625.
Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union, which cannot be held responsible for them.

Abstract

iMagine is a European project to serve aquatic researchers with a suite of high-performance image analysis tools equipped with artificial intelligence. To effectively achieve the objectives of the project, eight use cases in different areas of aquatic science are collaborating with the providers of the iMagine AI platform. This collaboration has yielded valuable insights and practical knowledge.

In this deliverable, we delve into the details of the best AI-based solutions for image processing in aquatic science, drawing on the extensive experience and knowledge we have gained over the course of the iMagine project. We thoroughly review the methods and tools used in the initial phase of data labelling, in the subsequent phases of model training and in the final deployment of the model as a service.

Document Description

Document title			
Work Package number			
Due date	30/09/2024	Actual delivery date:	01/10/2024
Nature of document	[Report]	Version	1.1
Dissemination level	Public		
Lead Partner	KIT		
Authors	Khadijeh Alibabaei (KIT), Elnaz Azmi (KIT), Valentin Kozlov (KIT)		
Reviewers	Tjerk Krijger (MARIS)		
Public link	https://zenodo.org/records/13866394		
Keywords	Best practices, AI, Image sets, Analysis		

Revision History

Issue	Item	Comments	Author/Reviewer
V 0.1	Draft version	Early draft	Khadijeh Alibabaei, Marco Francescangeli, Alberto Gaya, Elnaz Azmi, Jesús Soriano González, Jean-Olivier Irisson, Amanda Calatrava
V 0.2	Draft version	Contribution from the use cases	Carolin Leluschko, Jean-Olivier Irisson, Enoc Martinez, Antoine Lebeaud, Damian Smyth, Igor Atake, Sandro Luigi Fiore, Marco Mariano De Carlo, Wout Decrop, Rune Lagaisse, Jesús Soriano González, Martin Laviale, Sakina-Dorothée Ayata

V 0.3	First draft	First version for review	Khadijeh Alibabaei, Elnaz Azmi, Valentin Kozlov
V 0.4	Revision	Feedback from the reviewers	Tjerk Krijger
V 0.5	Revised version based on comments	Consolidated Comments	Khadijeh Alibabaei, Elnaz Azmi, Valentin Kozlov
V 1.0	FINAL		
V 1.1	FINAL	Executive summary updated	

Copyright and license info

This material by parties of the iMagine Consortium is licensed under a **Creative Commons Attribution 4.0 International License**.

Table of content

Executive summary

The iMagine project includes 8 scientific use cases, each empowering researchers in aquatic sciences with thematic AI analysis applications that are combined from generic AI models, scientific image datasets and various AI/ML and distributed computing solutions. Besides delivering AI-powered services, the project also facilitates the sharing of knowledge, challenges, and solutions among the 8 use cases through the "iMagine Competence Center", a virtual team that meets regularly to bring together AI-experts from aquatic and computational sciences. This document summarizes, and makes available the experiences and know-how that was gained by the iMagine Competence Centre during the past 2 years. The captured good practices are primarily aimed to help aquatic science communities that need to cope with image analysis challenges, but they can be relevant to other scientific domains where AI-based image analysis is in scope.

The first part of the document is structured by the main areas of work related to AI-powered image analysis: neural networks for image and video analysis; annotation of images for constructing training datasets; open publication of training datasets; data preprocessing methods prior to AI model training; evaluation metrics and experiment tracking tools; FAIR-ness aspects; recognising and preventing data bias; sharing trained AI models for interference. The second part of the document provides insight into the 8 iMagine use cases and shows how they applied the aforementioned techniques to real scientific scenarios.

Based on the experiences of the iMagine use cases, CVAT was the most commonly used image annotation tool due to its user-friendly interface, robust annotation features, and flexibility. The second most used annotation tool was LabelStudio, however, it was found to be more complex to use and to install in desktop environments. In terms of deep learning models YOLO was the most commonly used for object detection and it demonstrated strong performance across all our use cases. For segmentation we experienced that the panoptic segmentation approach with the Mask2Former tool achieved better performance in aquatic environments than instance segmentation using Mask R-CNN. For data sharing we found that Zenodo offers a robust solution, providing an open-access platform for researchers to share, store, and manage datasets. Prior to sharing the FAIR EVA tool has been used to improve the FAIRness of published training datasets concerning the metadata published in Zenodo along the training dataset. In terms of experiment tracking several tools were examined, and MLflow was found to be the most practical solution. Consequently, an MLflow server is provided centrally by the project to all the iMagine use cases for the efficient management and tracking of their machine learning experiments. The iMagine models were published as Docker images on the iMagine AI platform to make them available for other scientists and communities. The OSCAR platform provides a common and flexible inference layer for our use cases, delivering efficient and scalable deployment capabilities for running the AI models on end-user images either from the iMagine Platform, or from custom-made Web portals.

List of figures

List of tables

Acronyms

Please, see https://confluence.egi.eu/display/IMPAIP/Glossary

Introduction

Recently, machine learning, especially deep learning, has revolutionized the field of image analysis. Advances in neural network architectures such as Convolutional Neural Networks (CNNs) and Generative Adversarial Networks (GANs) have enabled computers to achieve high accuracy in tasks such as object recognition, image segmentation and image generation. These developments have transformed numerous industries, including healthcare, autonomous driving, and environmental monitoring, by providing powerful tools for extracting meaningful information from image data.
iMagine, an EC-funded project, contributes to the overarching mission of the EU of 'Healthy Oceans, Seas, Coastal and Inland Waters'. It does so by working towards the following project objectives:

- Enhancing aquatic research utilizing AI applications;
- Leveraging EOSC for developing, training, and deploying AI models;
- Online data stream analysis in distributed environments;
- Facilitate collaboration among research infrastructures to share images and AI applications;
- Developing best practices for delivering image processing services.

To effectively achieve the objectives of the project, currently eleven use cases (WP3) in different areas of aquatic science collaboratively engage with the iMagine AI Platform providers (WP4). These use cases are eight cases started with the project:

- UC1 Marine litter assessment
- UC2 Zooscan - EcoTaxa pipeline
- UC3 Marine ecosystem monitoring at EMSO sites (OBSEA, Azores, SmartBay)
- UC4 Oil spill detection
- UC5 Flowcam plankton identification
- UC6 Underwater noise identification
- UC7 Beach monitoring
- UC8 Freshwater diatoms identification

And three cases accepted via open calls:

- UC9 Cold water coral reefs
- UC10 Satellite derived bathymetry
- UC11 Fish otoliths

During the course of iMagine, the UCs gained practical experience in developing AI projects and performing image analysis: from collecting images, preparing training datasets, and training AI models, to deploying models for inference, specifically in the field of aquatic sciences. In this deliverable, we go through a general overview of these processes and highlight key learnings.

Purpose of the document

This document summarizes learned lessons and gained experiences within the iMagine Competence Center activities, starting from training data preparation, annotation, reviewing various preprocessing techniques, choosing AI model, to FAIR data evaluation, publishing datasets, and serving trained AI model for inference. The provided summary is based on the development of image sets and image analysis applications of iMagine use cases.

Scope of the document

This D3.4 document covers the best practices for producers and providers of image sets and image analysis applications leveraging AI-based solutions. This is the forth deliverable after the five mature use cases (UC1-5) progressed towards service delivery. The document covers our best know-how of today, but our knowledge and experience continue to expand by improving currently developed applications and assisting prototype services (UC6-8) to deliver their solutions.

Structure of the document

The document is structured as follows:

- An Overview of Deep Learning
    - We first review various neural networks relevant for image and video analysis, making an emphasis on those used by iMagine use cases.
- Annotation tools
    - A crucial step for AI model training is the preparation of a training dataset. Therefore, this section assesses annotation tools which proved their usefulness in application for aquatic science.
- Data repositories & Open-source datasets for Aquatic Applications
    - This section focuses on data repositories and training datasets. Once a training dataset is ready, it can be shared and published in a data repository.
- Preprocessing Techniques
    - Here, data preprocessing methods to improve data before feeding them to the AI model are described.
- Performance Metrics and evaluation methods
    - This section discusses AI model evaluation metrics and experiment tracking tools.
- Tools for monitoring model performance
    - Here, tools are discussed for monitoring the model performance and reliability during training (experiment tracking tools).
- Data biases in aquatic science models and data

    - This section goes into more detail on the data biases that arise in aquatic science when modelling or analysis data fail to accurately reflect reality.
- Model delivery
    - When AI models are trained and evaluated, they can be shared with the aquatic community and served for inference, which is covered in this section.
- AI model Drift Tools
    - In this section, we discuss the tool that was designed to detect, analyze, and mitigate model drift in AI systems in production.
- An overview of Use Case experiences
    - Finally, iMagine use cases provide insightful information and their experience on the used AI techniques.

An Overview of Deep Learning

Deep learning (DL), a branch of artificial intelligence (AI), utilizes multi-layered neural networks to model and interpret complex patterns in data. It has transformed image processing by empowering machines to learn and identify features from raw images, eliminating the need for manual feature extraction. Techniques like convolutional neural networks have driven significant advancements in tasks such as image classification, object detection, and image generation. This progress is crucial for developing sophisticated applications across various fields, especially in aquatic science. Over time, different CNN architectures have been developed, introducing unique structures to enhance performance, efficiency, and accuracy in tasks such as image classification, object detection, and segmentation. These networks are highly effective at automatically detecting and classifying patterns in images, making them indispensable for tasks such as monitoring marine biodiversity, mapping underwater habitats, and tracking changes in water quality or coral reef health. By automating these processes, CNNs improve the accuracy and efficiency of environmental assessments, facilitate more informed decision-making for conservation efforts, and enable more effective monitoring of aquatic ecosystems in response to climate change and human impacts.

Convolutional Neural Networks

Convolutional Neural Networks are a class of deep neural networks that are most commonly used to analyze visual images. They have proven to be extremely successful in various computer vision tasks, we briefly review the most well-known CNNs for:

- Image classification
    - Classifying images into predefined categories
- Object detection
    - Locating and identifying objects in an image
- Image segmentation
    - Segmenting images into different classes

Classification Methods

Classification is a fundamental task in the field of computer vision, which aims to classify input data into predefined classes or categories. In the field of computer vision, CNNs have redefined image classification by automatically learning hierarchical features from raw pixel data. Over the years, various CNN architectures have emerged, all characterized by different features and performance metrics. For instance, AlexNet, the winner of the ImageNet Large Scale Visual Recognition Challenge in 2012, was a pioneer in the use of deep convolutional networks. VGGNet increased the network depth with 19 layers, while ResNet introduced skip connections to address the vanishing gradient problem. MobileNet, on the other hand, represents a class of lightweight CNNs designed for optimized inference on mobile and embedded devices with limited computational resources. Table A shows some of the best-known models that have been developed for classification tasks:

Table A - List of the best-known CNNs for classification

Model name	Description
AlexNet[1]	One of the first deep convolutional neural networks, introduced techniques such as ReLU and dropout.
VGG[2]	Very deep convolutional networks, known for their uniform architecture with small (3×3) filters.
ResNet[3]	Deep residual networks, known for their ability to train very deep networks effectively.
Mobilenet[4]	Lightweight convolutional neural networks designed for mobile and embedded vision applications.
InceptionNet[5]	Designed to improve computational efficiency and accuracy through the use of inception modules.
Xception[6]	It is based on an improvement to the Inception model

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. (2017). ImageNet classification with deep convolutional neural networks. Commun. ACM 60, 6 (June 2017), 84–90. https://doi.org/10.1145/3065386

[2] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

[3] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 770-778). IEEE. https://doi.org/10.1109/CVPR.2016.90

[4] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., & Adam, H. (2017). *MobileNets: Efficient convolutional neural networks for mobile vision applications*. arXiv. https://arxiv.org/abs/1704.04861

[5] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 2818-2826). IEEE. https://doi.org/10.1109/CVPR.2016.308

[6] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, pp. 1800-1807, doi: 10.1109/CVPR.2017.195.

	architecture, specifically utilizing depthwise separable convolutions to make the network more efficient in terms of computational cost and parameter count.
DenseNet[7]	Densely connected convolutional networks where each layer receives feature maps from all preceding layers.
ResNeXt[8]	Extension of ResNet architecture with a cardinality parameter to improve representational power.
EfficientNet[9]	Compound scaling method that uniformly scales dimensions of depth/width/resolution
CLAP[10]	The CLAP model is a deep learning framework designed to align audio signals and textual descriptions in a joint embedding space.

Object Detection and Localization Methods

These methods are not only aimed at classifying objects in the image, but also at detecting and localizing objects within an image or a video and the identification of their class labels.

For the classification task, a labelled dataset consists of images with a single class label assigned to each individual image. However, for object detection and localization, the dataset requires labelled images where each object of interest is assigned a bounding box and the corresponding class label.

Current object detection algorithms can be divided into two main categories: two-stage detectors, such as RCNN[11], Fast-RCNN[12], and Faster-RCNN[13], and one-stage detectors,

[7] Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 2261-2269). IEEE. https://doi.org/10.1109/CVPR.2017.243

[8] Xie, S., Girshick, R., Dollár, P., Tu, Z., & He, K. (2017). Aggregated residual transformations for deep neural networks. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 5987-5995). IEEE. https://doi.org/10.1109/CVPR.2017.634

[9] Tan, M., & Le, Q. V. (2020). *EfficientNet: Rethinking model scaling for convolutional neural networks*. arXiv. https://arxiv.org/abs/1905.11946

[10] Wu, Y., Chen, K., Zhang, T., Hui, Y., Berg-Kirkpatrick, T., & Dubnov, S. (2023). Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation. In *Proceedings of the 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 1-5). IEEE. https://doi.org/10.1109/ICASSP49357.2023.10095969

[11] Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 580-587). IEEE. https://doi.org/10.1109/CVPR.2014.81

[12] Girshick, R. (2015). Fast R-CNN. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)* (pp. 1440-1448). IEEE. https://doi.org/10.1109/ICCV.2015.169

[13] Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 39*(6), 1137-1149. https://doi.org/10.1109/TPAMI.2016.2577031

such as Single Shot MultiBox Detector (SSD)[14] and You Only Look Once (YOLO)[15], see Table B.

Various software implementations were developed for these algorithms. For example, an external repository fasterrcnn-pytorch-training-pipeline[16] provides a pipeline for training PyTorch FasterRCNN models on custom datasets. With this pipeline, users have the flexibility to choose between official PyTorch models trained on the COCO dataset, use any backbone from Torchvision classification models, or even define their own custom backbones. The trained models can be used for object detection tasks on specific datasets. We have integrated the DEEPaaS API into this existing codebase, and this model is accessible as a general-purpose model on the marketplace of the iMagine platform.

Recently, the YOLO model has evolved considerably with the development of new versions designed to improve both the accuracy and efficiency of the system. YOLOv8[17] has significantly improved both the speed and accuracy of object detection compared to the previous versions. This model introduces various variants, including nano (n), small (s), medium (m), large (l), and xlarge (x) with a different number of variables that can be trained depending on the model variant.

The Ultralytics community[18] is an active contributor to the open-source community, providing accessible and cutting-edge solutions for various artificial intelligence tasks such as detection, segmentation, classification, tracking, and pose estimation. YOLOv8 is provided by the Ultralytics organization using the Pytorch framework. This model can be used flexibly for classification, detection, oriented object detection, and segmentation tasks. We have integrated a DEEPaaS API into the Ultralytics YOLOv8 and made it available as a general-purpose model on the marketplace of the iMagine platform. Depending on the application, users can select one of the model variants and train it. The Docker images are provided together with these models. The users have the option of either starting a deployment on the iMagine platform from this Docker image and training the models on the custom data without further effort. Alternatively, they can run these Docker containers on both Clouds and High-Performance Computing (HPC) infrastructures.

[14] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). SSD: Single shot multibox detector. In B. Leibe, J. Matas, N. Sebe, & M. Welling (Eds.), *Computer vision – ECCV 2016. Lecture Notes in Computer Science* (Vol. 9905). Springer, Cham. https://doi.org/10.1007/978-3-319-46448-0_2

[15] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 779-788). IEEE. https://doi.org/10.1109/CVPR.2016.91

[16] https://dashboard.cloud.ai4eosc.eu/marketplace/modules/ai4os-fasterrcnn-torch

[17] Jocher, G., Chaurasia, A., & Qiu, J. (2023). Ultralytics YOLO (Version 8.0.0) [Computer software]. https://github.com/ultralytics/ultralytics

[18] https://docs.ultralytics.com/

Table B - List of the best-known CNNs for object detection

Model name	Description
RCNN[11]	Region-based Convolutional Neural Network for object detection. It uses selective search to propose regions and applies a convolutional network to classify each region.
Fast RCNN[12]	Evolution of RCNN that improves speed and accuracy by introducing the Region Of Interest (ROI) pooling layer to efficiently extract region features.
Faster RCNN[13]	Introduces Region Proposal Network (RPN) to generate region proposals instead of using external methods like selective search, making it faster and more accurate. Integrates RPN with Fast RCNN architecture.
SSD[14]	Single Shot MultiBox Detector designed for real-time object detection. Utilizes a single neural network to predict bounding boxes and class probabilities for multiple object instances simultaneously at different scales.
YOLO[15] (v1,...,v10)	You Only Look Once models for real-time object detection. Divided into multiple versions (v1 to v10) each with improvements in accuracy and speed, focusing on predicting bounding boxes and class probabilities directly from full images in a single evaluation.
EfficientDet[19]	A family of object detection models that achieve better accuracy and efficiency by using a compound scaling method to balance the depth, width, and resolution of the network.

Segmentation Methods

The segmentation task in computer vision refers to the process of dividing an image into multiple segments or regions to simplify its representation and facilitate the analysis of its content. The goal is to group pixels that belong to the same object or have similar visual characteristics such as colour, texture, or intensity. A dataset used for segmentation tasks such as semantic segmentation or instance segmentation provides pixel-level annotations for the entire image. Instead of bounding boxes in the object detection task, each pixel in the image is labelled with a class label indicating the category of the object or region to which it belongs.
Segmentation tasks can be roughly divided into the following types:

[19] Tan, M., Pang, R., & Le, Q. V. (2020). EfficientDet: Scalable and efficient object detection. arXiv. https://arxiv.org/abs/1911.09070

- **Semantic segmentation**[20]: In semantic segmentation, each pixel in the image is assigned a class label that represents the category of the object or region to which it belongs. The main objective is to classify each pixel into predefined categories without distinguishing between the individual object instances.
- **Instance segmentation**[21]: Instance segmentation is an extension of semantic segmentation, where the goal is not only to classify each pixel into categories, but also to distinguish between different instances of the same object class. This means that each individual object instance in the image is assigned a unique identifier.
- **Panoptic segmentation**[22]: Panoptic segmentation combines both semantic and instance segmentation in a unified framework. It aims to segment and detect all objects in the scene, both things (e.g., background elements such as sky, road, grass) and objects (e.g., cars, people, animals), while also distinguishing between individual instances of objects.

Table C - List of the best-known CNNs for segmentation

Model name	Description
Fully Convolutional Networks (FCNs)[23]	FCNs are designed for semantic segmentation and replace the fully connected layers of traditional CNNs with convolutional layers to produce dense predictions. The model is effective for generating pixel-wise segmentation maps.
Segment Anything Model (SAM)[24]	SAM is a versatile foundation model for image segmentation that excels in zero-shot learning, allowing it to segment unseen objects and perform edge detection, object proposal generation and instance segmentation. It improves speed and accuracy in annotation tasks and is available under a permissive open license.

[20] Guo, Y., Liu, Y., Georgiou, T., et al. (2018). A review of semantic segmentation using deep neural networks. International Journal of Multimedia Information Retrieval, 7, 87–93. https://doi.org/10.1007/s13735-017-0141-z.

[21] Sharma, R., Saqib, M., Lin, C.T. et al. A Survey on Object Instance Segmentation. SN COMPUT. SCI. 3, 499 (2022). https://doi.org/10.1007/s42979-022-01407-3

[22] Kirillov, A., He, K., Girshick, R., Rother, C., & Dollár, P. (2019). Panoptic Segmentation. In 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 9396-9405). Long Beach, CA, USA: IEEE/CVF. DOI: 10.1109/CVPR.2019.00963

[23] Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 3431-3440). IEEE. https://doi.org/10.1109/CVPR.2015.7298965

[24] Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W.-Y., Dollár, P., & Girshick, R. (2023). *Segment Anything*. arXiv. https://arxiv.org/abs/2304.02643

Mask2Former[25]	Features are extracted from the image (at several scale) by a CNN. A pixel decoder scales those features back up to the original image size. A transformer decoder receives object queries and information from the pixel decoder to predict a mask and a class.
UNet[26]	U-Net is a convolutional neural network architecture designed specifically for biomedical image segmentation. It has a unique architecture that consists of a contracting path to capture context and a symmetric expanding path that enables precise localization.

Fully Convolutional Networks (FCNs) represent an approach in the field of semantic segmentation, which transforms traditional convolutional neural networks into fully convolutional structures. This enables training and prediction for pixel-wise segmentation tasks without relying on fully connected layers.

One of the features of FCNs is their ability to perform dense predictions for image segmentation by replacing the fully connected layers typically used for classification with convolutional layers. This structural modification allows the network to maintain spatial hierarchies and directly output segmentation maps that align with the input image dimensions.

FCNs facilitate end-to-end learning and prediction, which means that the network can be trained and inferred in a single step, directly mapping input images to segmented output images. By using only convolutional layers, FCNs preserve spatial information throughout the network, which is important for accurate pixel-wise segmentation. FCNs have been extensively used in various domains, such as medical imaging, autonomous driving, and scene understanding. For instance, in biomedical image segmentation, FCNs have demonstrated a good performance in tasks like cell segmentation and lesion detection.

Segment Anything model (SAM) [seg-anything] is a breakthrough foundation model for image segmentation that improves both the speed and quality of computer vision annotations. One of the most compelling features of SAM is its ability as a promotable segmentation system, demonstrating zero-shot generalization to unfamiliar objects and images, eliminating the necessity for additional training. Zero-shot learning [zero-shot] is a machine learning paradigm in which a model is trained to recognize classes or objects that it has never seen during the training phase. This is achieved by adding some form of

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

[26] Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional networks for biomedical image segmentation. In N. Navab, J. Hornegger, W. Wells, & A. Frangi (Eds.), *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. MICCAI 2015. Lecture Notes in Computer Science* (Vol. 9351). Springer, Cham. https://doi.org/10.1007/978-3-319-24574-4_28

additional information or knowledge about these unseen classes, enabling the model to generalize and make predictions even for unfamiliar instances. SAM uses zero-shot learning to segment unseen objects during inference. SAM can be asked to:

- perform edge detection, segment anything, including object proposal generation.
- segment detected objects, essentially performing instance segmentation.
- as a proof-of-concept, to segment objects from free-form text.

SAM is available under a permissive open license (Apache 2.0) 33. We are investigating adding the fine-tuning of the SAM to the marketplace.
Mask2Former is a deep learning model that combines different segmentation tasks, such as instance, semantic and panoptic segmentation, in a single architecture. Mask2Former utilizes transformers and a novel masked attention mechanism and efficiently processes image data by focusing on specific regions to generate accurate segmentation masks. The model is able to capture detailed relationships between pixels, resulting in improved segmentation quality and computational efficiency.

U-nets distinctive U-shaped structure consists of the main parts of an encoder and a decoder, connected by skip connections. The encoder or the contracting path follows the typical architecture of a convolutional network. The input images spatial dimensions are progressively reduced while increasing the number of feature channels with each step in the contracting path. Each downsampling step doubles the number of feature channels. Within the decoder part, or the expansive path, the goal is to to recover the spatial information and generate a segmentation map. Each step is made up with an upsampling of the feature map.

Foundation Models Next-Generation AI Models

Foundation models are large neural networks that have been trained with large amounts of diverse data and can be fine-tuned for various downstream tasks. They serve as a foundational base upon which more specific models can be built. These models are considered a significant next step in AI due to the following reasons:

- Generalisation across tasks and Transfer Learning: A significant advantage of foundation models is their ability to transfer knowledge from one domain to another. For example, a model trained on a large body of text data can be adapted to understand new languages, scientific texts, or legal documents without extensive retraining.
    - GPT (Generative Pre-trained Transformer)[27]: The GPT models developed by OpenAI are trained using large text corpora and can generate human-like text, answer questions, translate languages and much more.

[27] Brown, T. B., Mann, B., Ryder, N., Subbiah, et. al. (2020). Language models are few-shot learners. *arXiv*. https://arxiv.org/abs/2005.14165

    - BERT (Bidirectional Encoder Representations from Transformers)[28]: Developed by Google, BERT is optimized for understanding the context of words in a sentence, making it powerful for tasks such as question answering and language understanding.
- Multimodality Foundation models like CLIP[29] (Contrastive Language-Image Pretraining) integrate multiple data modalities (e.g., text and images).
- Efficiency in AI Development: By reducing the need to train models for every specific task from scratch, foundation models accelerate AI development and deployment.

Leveraging Foundation models can be considered as a next step beyond CNNs for AI-based aquatic services wherever applicable. So far iMagine use cases did not need to apply them.

The detailed application of the aforementioned CNN models for aquatic science cases of iMagine is described in the section "**An overview of Use Case Experiences**".

Annotation Tools

The rapid development of deep learning techniques, in particular convolutional neural networks, has revolutionized image analysis. However, the effectiveness of these models depends heavily on the quality and quantity of the annotated training data. Annotation tools allow users to annotate images with semantic labels, bounding boxes, polygons, and key points to create annotated datasets tailored to specific research objectives and model requirements.
This section summarizes our experience with several annotation tools used in aquatic sciences and image processing applications of the iMagine project. We examine the features, functionality, and suitability of each tool for annotating underwater images and videos, considering factors such as annotation complexity, scalability, and integration with AI frameworks.

BIIGLE

BIIGLE[30] is a web-based open-source tool for fast and effective labelling of images and videos. It was originally developed for monitoring and researching the marine environment, but it can also be used to annotate various image and video processing applications.
Advantages:

[28] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv*. https://arxiv.org/abs/1810.04805
[29] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., & Sutskever, I. (2021). Learning transferable visual models from natural language supervision. *arXiv*. https://arxiv.org/abs/2103.00020
[30] https://www.frontiersin.org/articles/10.3389/fmars.2017.00083/full ; https://biigle.de/

- It is open-source, GPL-3.0 license.
- A label tree is a collection of labels that can either be organized in a flat structure or arranged in a hierarchical, tree-like format. BIIGLE has a label tree that makes labeling easier.
- BIIGLE is capable of managing collections containing thousands, or even tens of thousands, of images or videos.
- All images and videos can be downloaded to a folder.
- The files can be loaded from a public web server, cloud storage, or by directly uploading volume files to BIIGLE.
- It allows you to view annotations with the same label in a regular grid.
- With an annotation assistance request, you can ask any person for help with annotating a particular image.
- It provides a Machine learning Assisted Image Annotation (MAIA) method.

One can download the labelling information for free in formats easy to read (e.g., CSV)
Disadvantages (mainly for videos):

- The video must be started and stopped and restarted each time to label each object.
- Lacks a tracking option for objects in the video.
- Slow process for uploading files.
- It does not offer rotated bounding box annotation.

RoboFlow

RoboFlow[31] is a platform that helps developers and businesses manage and deploy computer vision models more efficiently. It provides tools for data annotation, model training, and deployment that facilitate the development and deployment of custom computer vision applications. RoboFlow offers features such as data augmentation, model versioning, and integration with popular cloud services.

Advantages:

- It is primarily a cloud-based service.
- It has free tier operates under terms of service and usage agreements provided by RoboFlow for their cloud-based platform.
- In the video annotation, the labels of the previous frame can be regenerated to track objects.
- Permit users to partition the data into the train, validation, and test datasets.
- It is possible to version the dataset
- Enables users to annotate videos frame by frame, with the flexibility to choose the frequency of frames.
- The annotations can be exported in various formats such as JSON, XML, TXT and more.
- The platform can be used to train a model, such as YOLO models.

Disadvantages:

- Proprietary service based on closed source software.

[31] https://roboflow.com/

- All images are public on the public plan (free plan).
- The paid plan is too expensive.
- Feature limitations in the free tier. While RoboFlow offers free, many advanced features require paid subscription.
- Online requirement: RoboFlow is a cloud base tool, so a stable internet connection is needed.
- Does not provide an official on-premises installation.
- Uploading sensitive or proprietary data to the cloud can raise privacy and security concerns for some organizations.

Labelstudio

Labelstudio[32] is an open-source data labelling tool designed for a wide range of data types, including text, images, audio, video and time series. It is highly customizable, supports various annotation tasks and can be integrated into machine learning pipelines.

Advantages:

- The Community Edition of Label Studio is fully open-source under Apache License 2.0 and can be installed on-premises on user infrastructure.
- It also offers **Enterprise Cloud Service** under a commercial license.
- Supports multiple data types (text, images, audio, video, time series).
- Handles various annotation tasks, such as object detection, classification, and more.
- Supports multi-user collaboration for large teams.
- Active community support and comprehensive documentation help users get started and troubleshoot issues.

Disadvantages:

- Running the tool locally can be resource-intensive, particularly for large datasets or complex annotation tasks.
- Performance can vary based on the underlying hardware and setup.
- Requires installation and setup, which can be challenging for users without technical expertise due to sometimes missing details in the installation and configuration documentation.
- Some advanced features available in commercial tools might be missing, such as enhanced security, advanced user management, and priority support.
- Collaboration features require network connectivity, which might be a limitation in environments with poor internet access.

CVAT

Computer Vision Annotation Tool (CVAT[33]) is another popular open-source tool for annotating digital images and videos that was primarily developed to support computer vision tasks. It is open-source and there are two main free options to use it: Self-hosted and CVAT Cloud (limited).

[32] https://labelstud.io/
[33] https://www.cvat.ai/

Advantage:

- CVAT is an open-source tool under the Apache License 2.0.
- Supports various annotation types, including bounding boxes, polygons, polylines, points, cuboids, and 3D annotations.
- Designed for different tasks such as object detection, image segmentation, classification, and more.
- Provides an intuitive and easy-to-use interface for annotators.
- CVAT provides serverless functions for deploying containerized applications on GPU or CPU in a local system.
- It provides automatic annotation with pre-trained models, including segmentation, detection, and tracker models such as the Segment Anything Model (SAM), YOLOv7, text detection v4 or TransT.
- Users have the possibility to integrate their models for specific tasks.
- The installation supports multiple platforms, including Docker-based deployments, which makes it easy to set up on various types of infrastructure.

Disadvantage

- Cannot have more than 4 annotators in the free Cloud plan.
- Initial setup can be complex, especially for users without technical expertise in using Docker images and containers.
- Running CVAT, especially for large datasets or complex tasks involving automatic annotation, can be resource-intensive.
- Performance can vary based on the underlying hardware and system configuration.
- Some containers, such as SAM, are 'ALWAYS RESTART', which consumes the resources; the user should change the policy of the docker to stop these docker images.
- Collaborative features require network connectivity.

LabelBox

Labelbox[34] is an easy-to-use data labeling platform that simplifies the creation of training datasets for machine learning models. It provides an intuitive user interface that supports various annotation tasks, including object recognition, image segmentation and text classification, making it suitable for various projects.

Advantages:

- offers a free tier known as the Community plan under specific terms of service when signing up and using the platform.
- offers a cloud-based platform for data labeling and annotation, which is available as a service with various pricing tiers.
- Supports large-scale projects with the ability to manage multiple projects and large datasets efficiently.

[34] https://labelbox.com/

- One can add and remove labels as needed.
- Enables tagging frames with the option to adjust bounding boxes for tracking.
- Provides the ability to go frame by frame or play the video during annotation.
- Enables collaboration among team members with role-based access control, making it ideal for team-based annotation projects.
- Provides AI-assisted labeling features to automate and accelerate the annotation process.
- Comprehensive data management features, including dataset versioning and quality control.
- Multi domain input such as image, video, and text.

Disadvantages:

- Propietary service based on closed source software.
- Lacks offline capabilities, which can be a disadvantage for users who need to work without internet access.
- Although it offers a free tier, many advanced features and higher usage limits require a paid subscription, which can be expensive for some users.
- Being a cloud-based tool, it requires a stable internet connection.
- Uploading sensitive or proprietary data to the cloud can raise privacy and security concerns for some organizations.
- No on-premises installation option.

VIAME

The Video and Image Analytics for the Marine Environment (VIAME[35]) annotation tools are a suite of tools and utilities designed for the annotation, processing, and analysis of video and image data, particularly related to the marine environment. VIAME was developed by Kitware and its partners as part of the American National Oceanic and Atmospheric Administration's (NOAA) Automated Image Analysis Strategic Initiative (AIASI): "VIAME is an open-source computer vision software platform designed for do-it-yourself artificial intelligence (AI). It is an evolving toolkit that contains many workflows used to generate different object detectors, full-frame classifiers, image mosaics, rapid model generation, image and video search, and methods for stereo measurement." [32]

Advantages:

- Specifically tailored to underwater and marine data, making it ideal for researchers and professionals in marine biology.
- VIAME is fully open-source under the Apache License 2.0
- Offers a wide range of tools for video and image analysis, including object detection, classification, tracking, and more.
- Supports both manual and automated annotation processes.
- Completely open-source.
- Web and Desktop Annotator.

[35] https://www.viametoolkit.org/

- supports polygons, lines, points, or boxes, and can train models over multiple videos or image sequences using standard models.
- Ability to run image enhancement under the hood.
- GPU and CPU Installations.
- It can be configured to run in a cloud environment if users set it up on cloud infrastructure like AWS, Google Cloud, or Azure.
- It is a relatively self-contained environment that may suit some researchers.

Disadvantages:

- As an open-source project, VIAME may not have as large or active a community as some commercial tools.
- The software requires significant computational resources, including a powerful CPU and GPU, especially when processing large datasets or running complex algorithms.
- VIAME can be complex to set up and use, especially for beginners or users who are not familiar with command-line interfaces or programming.
- VIAME itself does not provide an official cloud-hosted instance. It is designed to be installed and run locally on user machines or servers.
- The documentation and tutorials for VIAME are currently quite limited as compared to the CVAT.
- VIAME supports just import/export of its own VIAME CSV, DIVE JSON Annotation formats and COCO Annotation format.

VGG Image Annotator

VGG Image Annotator (VIA[36]) is a popular web-based tool used for annotating images. It's primarily designed to facilitate the creation of ground truth data for training computer vision algorithms, such as object detection, image segmentation, and classification models. Here are its key features, advantages, and disadvantages:

Advantages:

- VIA is completely free to use under BSD-2-Clause License.
- VIA runs entirely in a web browser, making it accessible from any platform without the need for installation.
- It supports multiple annotation types, such as bounding boxes, polygons, points, and image classification tags.
- It allows exporting annotations in different formats, such as JSON, CSV, or plain text
- VIA is open-source software that allows individual annotators to collaborate as a team.
- It is designed as a client-side tool that runs directly in the user's web browser.
- It does not require installation or server hosting.

[36] https://www.robots.ox.ac.uk/~vgg/software/via/via_demo.html

- It can be used offline.

Disadvantages:

- While the interface is intuitive, it may lack some advanced features found in more specialised annotation tools.
- While VIA is suitable for small to medium-sized datasets, it may face scalability issues with large datasets due to performance limitations inherent in web-based applications.

Supervisely

Supervisely[37] is a powerful annotation platform that allows users to annotate images, videos, LiDAR 3D sensor fusion or even DICOM volumes, manage datasets, collaborate and train neural networks. Unlike other products, Supervisely creates a platform that integrates countless open-source tools and customized solutions into a single ecosystem with Supervisely Apps — interactive web apps that run in your browser yet are based on Python.

Advantages:

- Export to different models: The platform allows you to export annotations to different models, such as YOLO, making it compatible with a wide range of machine-learning tools.
- Integration with open-source apps: Supervisely integrates with numerous open-source apps available on GitHub. These apps allow you to run models, generate synthetic images, and perform other tasks related to computer vision.
- Fast and efficient annotation: The platform offers shortcuts and tools for annotating data quickly and efficiently, saving time and effort.

Disadvantage:

- Propietary service based on closed source software.
- Unfriendly interface: The user interface of Supervisely is not the most intuitive and attractive on the market. While it is functional, it could be improved in terms of design and usability.
- GPU dependency for certain apps: Some apps integrated into the platform require a GPU. This can limit access to certain functionalities for some users.
- A paid model with limitations: Supervisely has become a paid platform, which has limited the storage space available for free users. This can be a problem for projects that handle large amounts of data.

[37] https://supervisely.com/

Hasty.ai

Hasty.ai[38] promises to be an all-in-one solution for faster AI model creation and deployment. The company claims to solve three major pain points: slow manual annotations, lack of developer feedback and the need to manage multiple tools. Their platform improves annotation speed by 12x, reduces data quality control costs by 90% and streamlines the entire development process.

Advantages:

- Faster annotation and development: Hasty.ai boasts a significant increase in speed in both data annotation and model development, which can lead to considerable time savings.
- Real-time training model: One of the most attractive features of Hasty.ai is the ability to train the model as the annotations are made. This means that the model continuously learns and improves as the annotation process progresses, which can lead to greater accuracy and efficiency.
- Agile feedback loop: The platform is designed to learn from your actions and provide feedback throughout the development process so that you can identify opportunities to improve your model.

Disadvantages:

- Propietary service based on closed source software.
- Limited free plan: While Hasty offers a free plan to explore the platform, its functionality is restricted after you use up your virtual credits.
- Limited export formats: Hasty.ai may not export annotations in all desired formats, such as YOLO, which could limit compatibility with some tools.
- Lack of advanced shortcuts: Compared to other platforms, hasty.ai might have fewer keyboard shortcuts for faster annotation.

Summary of features per annotation tool

In **Table D**, we take a closer look at the comparison of the annotation tools and examine them based on several key features:

- Active learning: Some annotation tools employ active learning techniques that aim to reduce the annotation effort by strategically selecting the most informative data samples for annotation, thus optimizing the annotation process.
- Video support: Tools equipped with video annotation capabilities that allow users to efficiently annotate frames, segments, or entire videos.
- Data augmentation services: Certain annotation platforms provide data augmentation services that allow users to generate augmented versions of their annotated data.

[38] https://hasty.cloudfactory.com/

- Workflow management: Users can define sequential or parallel annotation tasks, assign them to annotators and monitor the progress of annotation projects in real-time to ensure efficient collaboration and project management.
- Document version control: Tools that offer robust version control capabilities allow users to effectively manage different versions of their annotated datasets.
- Automation and AI assistance: Platforms that use artificial intelligence (AI) technologies to automate annotation tasks and provide intelligent assistance to annotators can significantly accelerate the annotation process, improve the minimize human error.
- Collaboration and review features: Annotation tools that facilitate collaboration and review between annotators and project stakeholders promote effective communication, feedback sharing, and quality assurance.

The CVAT was the tool most used among the iMagine project UCs due to its features, such as automatic labeling, which significantly streamlined the annotation process.

Table D – Overview of the annotation tools and their features

Feature	BIIGLE	ROBO FLOW	Label studio	VIAME	VGG	hasty.ai	CVAT	LabelBox	LabelImg	Super-visely
Active learning	No	Yes	Yes	Yes	No	Yes	No	Yes	No	Yes
Video Support				Yes	No	Yes	Yes			
Data augmentation service	No	Yes	No	Yes	Yes	Yes	No	Yes	No	Yes
Workflow management	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes
Document version control	Yes	Yes	Yes	No	No	Yes	Yes	Yes	No	Yes
Automation and AI assistance	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes
Collaboration and review	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes
Locally hosted	No	Yes	Yes	Yes	Yes	No	Yes	No	Yes	Yes

Open-source	Yes	Paid and open-source	Yes	Yes	Yes	No	Yes	No	Yes	No
Input format type	Images	Images, videos, datasets	Various	Various (e.g., images, video)	Images	Images, Annotations	Images, videos	Various	Images	Images, videos, datasets
Export format	JSON, XML, CSV, TXT etc	JSON, CSV, XML	JSON, CSV, XML	Various (e.g., CSV, JSON)	CSV, JSON	Various (e.g., JSON, CSV)	JSON, XML, COCO, YOLO,, PNG, etc	JSON, CSV	XML, TXT	JSON, CSV, XML
Extra feature	No	Model training & deployment	Customisable interfaces	Specialised for marine life	Pre-trained models for images	Automated labelling	Customisable workflows, Automated labelling	Model training & deployment	Simple UI	Model training & deployment
Supported tasks	Detection, Segmentation, Tracking	Detection, Segmentation, Classification	Detection, Segmentation, Classification, NLP	Object detection, tracking	Classification, object detection	Object detection, segmentation	Classification, Detection, Segmentation, Tracking	Detection, Segmentation, Classification	Detection	Detection, Segmentation, Classification

Data Repositories and Open-source Datasets for Aquatic Applications

Open-source datasets are a valuable contribution to expanding scientific knowledge, supporting policy decisions, and promoting collaboration in marine research and conservation.
Each UC of iMagine has created and used labelled data sets to train their applications. One goal of iMagine was to make these image training datasets available to external users in order to (D3.3[39]):

- Show the images used to train the AI models to increase the user's confidence in and understanding of these models.
- Enable the reuse of the images for further use cases, including retraining the models or training further AI models outside the consortium.

To this end, it was decided that the labelled images used for training all iMagine use cases (UC1-8) will be catalogued for download via the **Zenodo**[40] system.
The reasons why Zenodo is a compelling choice for storing and sharing data are (**D3.3**):

- Long-term storage: Zenodo is supported by the EU and offers guaranteed long-term storage and dissemination of research results.
- Persistent identifiers: Each publication is assigned a Digital Object Identifier (DOI) so that the work can be easily found and cited.
- Version control: Zenodo enables clear versioning of the datasets so that researchers can track changes and access specific versions when needed.
- Generous storage: With 50 GB of free storage per publication and the ability to request more, Zenodo can meet most research data needs.
- Usage insights: Zenodo provides valuable download and access statistics to help researchers understand the data's reach and impact.

One drawback is that Zenodo serves as a general repository for a wide range of EU results from different areas. Its publication template is primarily tailored to reports and papers and is, therefore, less suitable for describing datasets underpinned by domain-specific vocabularies. However, iMagine has discussed and agreed with Zenodo as part of the EU-funded Zenodo-ZEN project to work on a more domain-specific approach. This means that iMagine formulated a specific template for iMagine training image datasets as a DCAT profile supported by aquatic vocabularies (**D3.3**).

In this section, we provide an overview of the available open-source data for aquatic scientists used or produced by UC of the iMagine project.

[39] https://doi.org/10.5281/zenodo.11520846
[40] https://zenodo.org/communities/imagine-project

Table E - Overview of open-source datasets for aquatic science

Dataset Name	Description	Appli-cation
FathomNet[41]	FathomNet, an open-source image database, serves as a valuable resource for training, testing, and validating cutting-edge artificial intelligence algorithms aimed at exploring our ocean and its diverse inhabitants. Drawing inspiration from well-known annotated image databases like ImageNet and COCO, FathomNet strives to create a comprehensive reference dataset specifically focused on images depicting marine life.	Detection
MoNKA [42]	To download pic from MoNKA **https://github.com/obsea-upc/minka-downloader?tab=readme-ov-file**	
UC1 Marine litter assessment[43]	Plastic litter detection, classification, and quantification (APLASTIC-Q)' by Wolf et al. 2020. It consists of a Plastic Litter Detector (PLD) dataset which contains classes with litter present (two classes) and classes with litter-free areas (four classes). The second Plastic Litter Quantifier (PLQ) dataset contains classes of litter types (14 classes) and classes of litter-free areas (4 classes). Each data set is split into a train (80%) and a test dataset (20%).	Classifi-cation
ZooScanNet (UC2-iMagine)[44]	1.4M plankton images taken with a ZooScan and classified into 93 taxa	Classifi-cation
Segmentation masks for ZooScan[45] (UC2-iMagine)	14k images with manual segmentation marks between touching objects + 5k images of single objects over a white background, to serve as negative segmentation examples	Segmen-tation, object detection

[41] https://fathomnet.org/fathomnet/#/
[42] https://minka-sdg.org/
[43] https://zenodo./records/4552389
[44] https://doi.org/10.17882/55741
[45] https://doi.org/10.17882/99

UC3 Marine ecosystem monitoring	UC3o EMSO OBSEA	• Training data to be published on Zenodo • Live stream to be on YouTube	
	UC3a EMSO Azores	To be published in SeaDataNet SEANOE (SEA scieNtific Open Data Edition)	
	UC3s Smart-Bay	To be published in Zenodo (Underwater Marine species dataset and Nephrops Burrow dataset)	
UC4 Oil spill detection		Stored in a Data Management Platform at the University of Trento, with public access via THREDDS and already published in Zenodo[46]. The THREDDS catalog includes both oil spill images (observations) and NetCDF data (simulation outputs) available also in JSON format.	
UC5 Flowcam plankton identification[47]		Training dataset with over 300,000 images is already in Zenodo. To grow this up to 2.2 million during the project.	Classifi-cation
UC6		Not yet published	Classifi-cation
UC7	Shoreline Extraction [48]	The SCLabels dataset is intended to be used in the exploring and development of Artificial Intelligence (AI) applications aimed at the automation of the shoreline extraction process from rectified images.	Segmen-tation
	Beach seagrass Wrack Identification Labelled Dataset[49]	BWILD is a dataset tailored to train Artificial Intelligence applications to automate beach seagrass wrack detection in RGB images. It includes oblique RGB images captured by SIRENA beach video-monitoring systems, along with corresponding annotations, auxiliary data and a README file.	Detection and segmen-tation

[46] https://zenodo.org/records/11354663
[47] https://zenodo.org/records/10554845
[48] https://zenodo.org/records/10159978
[49] https://zenodo.org/records/12698764

UC8[50]	The dataset consists of two components: individual diatom images extracted from publicly available diatom atlases and individual debris images. Currently, the first repository (individual diatom images) consists of 166 diatom species, totaling 9230 images. The subfolders within each diatom species indicate the origin of the images. The second repository (individual debris images) contains 600 debris objects extracted from real microscopy images.	Detection

Preprocessing Techniques

Raw data often include noise, corruption, missing values, and inconsistencies, making preprocessing essential. Poor data quality can affect accuracy and lead to false predictions. For example, Gaussian noise, which adds random variations in pixel values, can blur the objects' edges and fine details. Salt-and-pepper noise introduces random black and white pixels, which can obscure important parts of the image. An AI model trained on noisy images might misclassify images because the noise alters the texture and outline, which are crucial for the model's recognition process.

Therefore, preprocessing through cleaning, integration, transformation, and reduction is vital for facilitating and more accurate knowledge extraction. Techniques such as classification, clustering, and association, among others, enhance data quality and improve the performance of AI models[51]. These techniques are frequently used in combination, depending on the specific requirements of the AI model, the characteristics of the imaging data and their domain. Here are some common preprocessing techniques for imaging data:

Data Cleaning

Raw data is prepared for analysis through data cleaning techniques that address issues which could negatively impact the model's performance. Data cleaning is an essential part of data preprocessing, focused on fixing or removing errors, inconsistencies, and irrelevant information. It ensures the dataset is accurate, consistent, and complete, helping the model perform better and generalize effectively to new data.

[50] https://dorel.univ-lorraine.fr/dataset.xhtml?persistentId=doi:10.12763/UADENQ

[51] Maharana, K., Mondal, S., Nemade, B. (2022). A review: Data pre-processing and data augmentation techniques. In Global Transitions Proceedings. Volume 3, Issue 1, Pages 91-99. https://doi.org/10.1016/j.gltp.2022.04.020

Images might come in different sizes. Resizing them to a uniform size reduces computational complexity and ensures consistency. Furthermore, already trained neural networks require a certain input image size (e.g., ResNet50 requires images of 224×224), that should be fulfilled to use them for prediction of the new images. Additionally, rescaling pixel values to a specific range (e.g., [0, 1]) helps in numerical stability during training.

Normalization and Standardization

Scaling techniques such as normalization and standardization improve model performance, reduce the impact of outliers, and ensure that the data is on the same scale contributing equally to the model. Generally, there is no method other than trial and error to know which technique is the best for a specific dataset.

Normalization scales the pixel values to a range, usually [0, 1] or [-1, 1]. This ensures that all the input data is consistent and within a range that the neural network can process effectively. It reduces computational complexity and helps in faster convergence during training.

Standardization adjusts the pixel values so that they have a mean of zero and a standard deviation of one. This is particularly useful when the dataset has varying lighting conditions and contrasts. It helps in faster and more stable training by ensuring the data distribution is consistent. Standardization can lead to faster and more stable convergence during training.

Traditional vs. Just-in-Time Scaling

The pixel values in images must be scaled before providing the images as input to a deep learning neural network model during the training or evaluation of the model.

Traditionally, the images would have to be scaled before the development of the model and stored in memory or on disk in the scaled format. The latter approach requires preprocessing the entire dataset before training, which can be time-consuming and resource-intensive, especially with large datasets, but has the advantage that scaling does need to be repeated for every experiment

An alternative approach is to scale the images using a preferred scaling technique just-in-time during the training or model evaluation process. Keras supports this type of data preparation for image data via the ImageDataGenerator class and API. PyTorch supports this type of data preparation for image data via the "`torchvision.transforms`" module and its various transformation functions. This method allows for on-the-fly data augmentation and scaling, reducing the need for extensive preprocessing and storage.

Handling Imbalanced Data

Class imbalance occurs when the number of instances of one class is significantly higher or lower than the instances of other classes. This can cause the model to be biased towards the majority class and perform poorly on the minority class.

To handle imbalanced classes, several techniques are used, such as resampling methods:

- Resampling (Oversampling and Undersampling)
    - Oversampling increases the number of instances in the minority class by duplicating existing instances, augmentation or creating new instances synthetically.
    - Undersampling reduces the number of instances in the majority class by randomly removing instances.
- Class weight adjustment
    - Modifying the cost function to give importance to the minority class.
- Synthetic data generation
    - Creating synthetic samples for the minority class using techniques like SMOTE (Synthetic Minority Oversampling Technique).

Data Augmentation

Augmenting data by applying transformations like rotation, flipping, scaling, and cropping helps in increasing the diversity of the dataset, thereby improving the generalization capability of the model and reducing overfitting (when the model memorizes the specific features of the training data set, rather than the underlying patterns of the broader data set).

It is important to note that the augmentation technique should transform the training dataset by reflecting realistic and relevant variations of the original dataset. This can then help the model to better generalize without deviating far away from the real scenarios that might be encountered.

Noise Reduction

Noise in images can adversely affect model performance. Techniques such as Gaussian blurring, median filtering, or denoising autoencoders can be used to reduce noise as described in the following.

Gaussian blur[52] is an image processing algorithm used to smooth the image. The blurring effect reduces sharp edges and creates smooth color transitions at the edges. Gaussian blur, achieved by applying the Gaussian function to an image, creates a normal

[52] https://scholarworks.calstate.edu/downloads/m326m425n

distribution of pixel values, smoothing out some randomness, reduces noise and minimizes details. It creates an effect similar to viewing the image through a translucent lens. It is often used in preprocessing to enhance image structure and involves convolving the image with a Gaussian kernel matrix.

The median filter[53] is a non-linear digital filtering technique commonly used to remove noise from images or signals. It is widely used in digital image processing because it preserves edges while effectively reducing noise, making it a valuable preprocessing step for tasks like edge detection. Unlike a linear filter that averages values, the median filter sorts the pixel values within a specified neighbourhood around each pixel and replaces the central pixel with the median value of that group. This technique is especially effective at removing "salt-and-pepper" noise, which manifests as random black and white specks, while preserving the sharpness of edges in the image.

Denoising Autoencoders[54] (DAEs) are a variation of the traditional autoencoder, where the input to the encoder is a noisy or corrupted version of the original data. The encoder, a neural network with hidden layers, processes this noisy input to generate a low-dimensional encoding. The decoder, another neural network, then reconstructs the original data from this encoding. The loss is calculated by comparing the reconstructed output with the original, uncorrupted input. Training with noisy data as input and clean data as the target helps the model to focus on learning meaningful features in the latent space, effectively ignoring the noise, which allows it to reconstruct a clean version of the input.

Contrast Enhancement

Enhancing contrast can improve the visibility of features in the image. Techniques like histogram equalization or adaptive histogram equalization can be employed for this purpose[55].

Feature Extraction

Depending on the task, extracting relevant features from images can be beneficial. Techniques like edge detection (e.g., Sobel, Canny), texture analysis (e.g., Gabor filters), or deep feature extraction using pre-trained convolutional neural networks (CNNs) can be applied.

---

[53] https://www.researchgate.net/publication/271637291_An_Improved_Median_Filtering_Algorithm_for_Image_Noise_Reduction

[54] https://www.geeksforgeeks.org/denoising-autoencoders-in-machine-learning/

[55] Raveendran, S., Patil, M.D. & Birajdar, G.K. Underwater image enhancement: a comprehensive review, recent trends, challenges and applications. *Artif Intell Rev* **54**, 5413–5467 (2021). https://doi.org/10.1007/s10462-021-10025-z

Dimensionality Reduction

For high-dimensional imaging data, techniques like Principal Component Analysis (PCA) or t-distributed Stochastic Neighbor Embedding (t-SNE) can be used to reduce dimensionality while preserving important information.

Image Segmentation

Segmenting images into meaningful regions can facilitate object detection, recognition, or tracking tasks. Techniques like thresholding, region-growing, or deep learning-based segmentation models can be employed.

Artifact Removal

In aquatic imagery, artifacts like motion artifacts, scanner or underwater camera artifacts can degrade image quality. Data collected during the study of aquatic environments often contains unwanted elements like noise, distortions, or errors, referred to as artifacts. These artifacts can originate from various sources, including measurement equipment, environmental conditions, or data processing techniques, potentially obscuring or distorting the true characteristics of the aquatic system under investigation. Removing these artifacts improves the quality and accuracy of the data, enabling more precise analysis, interpretation, and decision-making. Techniques like interpolation, artifact detection, and correction algorithms can be applied.

Performance Metrics and Evaluation Methods

Performance metrics and evaluation methods are essential components of any machine learning project, as they provide insights into the effectiveness and accuracy of the trained models. These metrics quantify the performance of the model and help to assess its suitability for the intended task. Common performance metrics include:

- **Accuracy**: The ratio of correctly predicted instances to the total instances. $Accuracy = \frac{Number\ of\ Correct\ Predictions}{Total\ Number\ of\ Predictions}$
- **Precision**: The ratio of correctly predicted positive observations to the total predicted positives. $Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$
- **Recall**: The ratio of correctly predicted positive observations to all observations in the actual class. $Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$
- **F1 score**: The harmonic mean of precision and recall, useful for imbalanced datasets. $F1\ score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$
- **ROC-AUC**:
    - ROC Curve (Receiver Operating Characteristic Curve): A graphical plot that illustrates the diagnostic ability of a binary classifier system.

  - AUC (Area Under the ROC Curve): Measures the entire two-dimensional area underneath the entire ROC curve.
- **Mean Square Error (MSE)**: The average of the squared differences between predicted and actual values. A lower value indicates better performance. $MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y}_i)^2$
- **mean Average Precision (mAP)**: It measures the accuracy of a model in detecting objects and localizing them within an image. A higher mAP score signifies better performance. $mAP = \frac{1}{n}\sum_{i=1}^{n} AP_i$, where AP_i is the average precision for the i-th class. AP is the area under the precision-recall curve for a single class. It summarizes the precision-recall curve into a single value representing the average precision at different recall levels.
- **Intersection over Union (IoU)**: Intersection over Union (IoU) measures the overlap between two boxes, with greater overlap indicating a higher IoU. It is primarily used in object detection to train models to accurately predict bounding boxes around objects. For instance, a green box (correct) and a blue box (predicted) should ideally overlap perfectly, achieving an IoU of 1. IoU is also employed in non-max suppression to eliminate redundant boxes around the same object. In this method, the bounding box with higher confidence is selected and all remaining bounding boxes that have an IoU less than a predefined IoU threshold with the selected box are removed.
- **Minimal Intersection over Union (Min IoU)**: Minimal Intersection over Union (Min IoU) is the area of overlap of two bounding boxes divided by the minimum area of the two bounding boxes. This measure is helpful while trying to understand how bounding boxes are overlapping each other, and execute a cleaning algorithm similar to Non-Maximum Suppression.
- **Cross-Entropy (CE)**: It quantifies the difference between the predicted probability distribution generated by the model and the true probability distribution, which is usually represented by a one-hot encoded vector indicating the correct class. CE is often used as a loss function during model training. For a single sample, the CE loss can be expressed as:

  $H(p,q) = -\sum_{i=1}^{c} p_i log(q_i)$, where p_i is the true distribution, (q_i) is the predicted probability for class i, and c is the total number of classes. For a dataset with multiple samples, the CE loss is averaged across all samples.
- **The Fraction Skill Score (FSS)** was originally introduced in the context of probabilistic weather forecasting analysis. It evaluates forecast accuracy by accounting for the uncertainty and variability of weather events. Specifically, the FSS measures the skill of probabilistic predictions by comparing them to a ground truth (e.g., satellite observations). Unlike a traditional overlay method, this metric considers not only the common area between simulation and observation but also the spatial distribution within that area.

The FSS is calculated as:

$$FSS = 1 - \frac{\sum_{i=1}^{n} (f_i - o_i)^2}{\sum_{i=1}^{n} (f_i^2 + o_i^2)}$$

where f_i is the forecast fraction, o_i is the observed fraction, and the summation runs over all grid points i. The FSS ranges from 0 to 1, where 1 indicates a perfect match between the forecast and observation, and 0 indicates no skill.

Validation model

A confusion matrix is a table layout that illustrates the performance of an algorithm, typically for a supervised learning classification task. It compares the actual target values with those predicted by the machine learning model. The table shows the number of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). The table shows the representation of the actual classes and columns of the predicted classes.

In a binary classification, if you have 100 test samples and your confusion matrix shows 80 true positives, 10 true negatives, 5 false positives, and 5 false negatives, it helps to evaluate how well the model distinguishes between the two classes.

- TP: These are the cases in which the model correctly predicts the positive class. If the actual class is positive (e.g., the presence of a disease), and the model also predicts it as positive, it is a true positive.
- FP: These are the cases in which the model incorrectly predicts the positive class. If the actual class is negative, but the model predicts it as positive, it is a false positive.
- TN: These are the cases in which the model correctly predicts the negative class.
- FN: These are the cases in which the model incorrectly predicts the negative class.

Tools for monitoring model performance

While assessing various preprocessing techniques, AI models, choosing right metrics etc., it is essential to keep track of these experiments and compare them. Experiment tracking tools are software platforms or frameworks designed to help researchers and data scientists efficiently manage and monitor their machine learning experiments. These tools provide features for organizing, recording, visualizing, and analyzing experimental data, making it easier to track the progress of experiments, compare results, and reproduce findings. Here are some popular tools for tracking experiments:

Tensorboard

TensorBoard[56] is a powerful tool that provides the visualization and tooling needed for machine learning experimentation. It was originally developed for use with TensorFlow, but it has since been adapted to work with other deep learning frameworks, including **PyTorch**. Tensorboard allows you to visualize various metrics, graphs, and other details about your model.

- Visualization: Provides detailed visualizations of metrics such as loss and accuracy to help understand model performance.
- Graph Analysis: Allows you to view the calculation graph and helps with troubleshooting and optimization.
- Embeddings: Projects embeddings into low-dimensional spaces for easier interpretation.
- Flexibility: Supports visualizations of histograms, images, text and more.
- Can be self-hosted solutions.

MLflow

MLflow[57] simplifies machine learning development by integrating experiment tracking, packaging code into reproducible runs, and facilitating model sharing and deployment. It provides lightweight APIs that are compatible with any ML application or library (such as TensorFlow, PyTorch, XGBoost) and can be used in various environments such as Jupyter notebooks, standalone applications, and the cloud.
Key features of the tool are:

- Model Registry: MLflow includes a centralized model registry for managing and versioning models.
- Experiment Tracking: With MLflow, you can log parameters, metrics, artifacts (e.g., model files) and other metadata from your machine-learning experiments.
- MLflow Projects: packaging ML code in a reusable and reproducible form to share with other data scientists or transfer to production;
- MLflow Models: managing and deploying models from various ML libraries to multiple model serving and inference platforms;
- Can be self-hosted solutions.

In the context of iMagine, we have deployed an MLflow tracking server primarily for experiment tracking[58]. We added another layer on top of the MLflow server that enables users to self-register in MLflow via the project common AAI (EGI Check-In) and to manage permissions to their experiments and models with other users. We implemented automatic backup of the in-use databases and enabled manual restore operations. The code related to the dockerized solution is available.

[56] https://www.tensorflow.org/tensorboard
[57] https://mlflow.org/
[58] https://mlflow.cloud.imagine-ai.eu/

Weight and Biases

Weights & Biases[59] (W&B) provides a comprehensive platform that facilitates experiment tracking, visualization and collaboration on machine learning and artificial intelligence projects.

- Experiment Tracking: W&B enables users to comprehensively track machine learning experiments. It logs metrics, hyperparameters, and other relevant information during model training, providing a clear record of experiment configurations and results.
- Visualization: The platform provides interactive visualizations that help users identify trends in model performance, compare experiments and identify areas for improvement.
- Collaboration: Teams can collaborate effectively via W&B by sharing experimental results, visualizations, and insights. This promotes transparency and knowledge sharing within research and development teams.
- Integration: W&B integrates seamlessly with popular machine learning frameworks such as TensorFlow, PyTorch and scikit-learn. This integration simplifies the logging and visualization of experiments directly from these frameworks.
- Dashboard and project management: Users can manage their machine learning projects and experiments via a user-friendly dashboard. It provides tools to efficiently organize, search, and analyze experiment data.
- Hyperparameter optimization: it provides capabilities for hyperparameter optimization (HPO) as part of its suite of experiment tracking and management tools.
- Cannot be self-hosted solutions.
- It has various payment plans, the Free one is limited, but there is one for academia.

Tensorboard and MLflow are flexible tools that allow for self-hosting, offering users enhanced control over their deployment and data management. In contrast, W&B does not offer a self-hosting option, as it operates as a cloud-based service. W&B offers a range of payment plans tailored to different user requirements, including a limited free tier. Additionally, W&B provides a dedicated plan for academia, specifically designed to support researchers and students working on machine learning projects.

Data Biases and Fairness in Aquatic Science Models and Data

Aquatic science models, like many other scientific models, rely heavily on data to make predictions, draw conclusions and make decisions. However, these models are not immune to biases inherent in the data on which they are based, which can lead to unfair

[59] https://wandb.ai/

predictions. Understanding data bias and ensuring fairness in aquatic scientific models is important to the integrity and reliability of the predictions and recommendations they provide.

Data biases in aquatic science arise when modelling or analysis data fail to accurately reflect reality. Phenotypic variability of natural systems, for instance a species may have different morphologies depending on its age and location, can introduce substantial biases into aquatic datasets. These biases affect population estimates, community analyses, and the evaluation of ecosystem health. Data biases can stem from:

- Sampling Bias: Certain regions or species may be over- or under-represented, leading to inaccurate depictions of less-studied or remote areas. In other words, data collection does not incorporate adequate randomization that results in class imbalance. This imbalance biases the model toward the majority class, as it encounters and learns from it more frequently. This leads to underperformance when predicting the minority class, resulting in poor generalization and skewed predictions.
- Measurement Bias: Errors in data collection methods, such as sensor inaccuracies or inconsistent sampling, can distort results.
- Temporal Bias: Data collected at specific times may overlook seasonal variations or long-term trends, impacting models that do not account for these changes.
- Geographical Bias: Some areas may be over-represented due to ease of access or historical research focus, leading to an incomplete understanding of broader or regional conditions.
- Group attribution bias: The tendency to generalize observations or characteristics from individual aquatic organisms or specific events to an entire species or ecological group, that can result in oversimplified or inaccurate conclusions.
- Morphological Bias: Ontogenetic alterations, can introduce size bias in datasets, as sampling may favor certain age groups. Phenotypic, morphology or behavior variability in response to environmental conditions, can lead to misinterpretations of size or age distributions if these environmental effects aren't accounted for. Additionally, sexual dimorphism, with males and females differing in size or behavior, can cause sex-based bias if sampling methods don't differentiate between sexes or favor one over the other.

Addressing Fairness and Reducing Bias

Fairness in aquatic science models involves making sure that predictions and analyses are just and do not disproportionately impact any group or area. Implementing strategies to ensure fair research practices minimizes the influence of subjective biases. These strategies include:

- Equitable Representation: Ensuring that data from all relevant regions, species, and conditions are inclusively represented to create models that generalize well.

- Impact Assessment: Analyzing how model results influence different stakeholders, such as local communities and conservation efforts.
- Bias Mitigation: Identifying and addressing biases through techniques like re-sampling, applying domain expertise, or using statistical adjustments to ensure a balanced dataset. Conducting studies across multiple populations and habitats that include a wide range of species to provide a more comprehensive understanding of aquatic ecosystems. Standardization of sampling methods across space and time, collecting environmental data (e.g., temperature, salinity) alongside biological samples, and using multiple sampling techniques to capture a full range of sizes and life stages can minimize morphological biases. For class imbalance bias reduction, see Section "**Handling Imbalanced Data**".
- Transparency and Open Science: Openly available research allows verification and further study by the scientific community, facilitating thorough and unbiased peer review processes to critically assess research findings and methodologies.

In 2016, a basic framework of principles known as FAIR (Findable, Accessible, Interoperable, and Reusable) was introduced to enhance the management and governance of data and facilitate the reuse of scientific information[60]. Originally designed for scientific data, these principles have since extended to various digital assets, including AI models and datasets.

To improve the management and sharing of research data, FAIR EVA[61], has been developed within the European Open Science Cloud context. It is designed for specific data management systems, such as open repositories, which can be tailored to individual use cases in a scalable and automated environment. The tool aims to be flexible and adaptable, supporting various environments, repository software, and disciplines, in line with the flexibility of the FAIR Principles. FAIR EVA tool in the context of iMagine has been used to improve FAIRness of published training datasets.

Model delivery

The accessibility and reusability of trained AI models are important for researchers to foster collaboration and accelerate innovation. Once the AI model is trained and validated, it can be shared on the iMagine marketplace and deployed in the production aquatic image service. In this section, we have identified six different model deployment patterns to offer AI models in production. For each one, we detail the best practices to help use cases with the adoption of the tools and services available in the iMagine AI Platform that best fits their needs. As previously described in **D3.3**, the trained and

[60] Wilkinson, M., Dumontier, M., Aalbersberg, I., Appleton, G., Axton, M., Baak, A., ... Mons, B. (2016). The FAIR Guiding Principles for scientific data management and stewardship. Scientific Data, 3, 160018. https://doi.org/10.1038/sdata.2016.18

[61] Aguilar Gómez, F., & Bernal, I. (2023). FAIR EVA: Bringing institutional multidisciplinary repositories into the FAIR picture. *Scientific Data*, *10*(1), 764. https://fair.csic.es/es

validated models can be offered and made accessible to external users through different approaches. We refer to them in this section.

Model sharing via the iMagine marketplace

The simplest option is to provide your trained models via the iMagine Marketplace component of the Platform. The trained and validated models are integrated with API (recommended is DEEPaaS API[62], REST API for AI/ML/DL) and packed as Docker images. This enables external users to download the AI modules as Docker images and run them on their own or third-party external compute resources. This pattern corresponds with approach 2, "Marketplace download service delivery", described in more detail in **D3.3**. In addition, the marketplace offers "Marketplace inference service delivery" to run trained AI models for inference on the connected back-end cloud resources. The latter is also available for non-partners via short-lived (10 minutes) "Try" endpoints.

Processing files in an event-driven approach

To support this pattern, the project provides the OSCAR framework[63]. OSCAR is an open-source platform built on Kubernetes for event-driven data processing of serverless applications packaged as Docker containers. The execution of these applications can be triggered both by detecting events from object-storage systems, such as MinIO or dCache (asynchronous calls) or by directly invoking them (synchronous calls). The main benefit of this approach is the scalability of the inference jobs, as the OSCAR cluster can autoscale, adapting its size to the workload transparently for the users (and freeing up the resources when they are not needed).
For this pattern, producers and providers of the aquatic analysis service can choose between two options:

- Deploy a model in the project's OSCAR serverless inference platforms: the iMagine OSCAR clusters support multitenancy and offer accounting thanks to the Prometheus and GoAccess services. They can collect metrics like resources consumed by inference executions, the number of deployed services over a period of time or the geolocation of the users interacting with the services. With this approach, models can be made accessible through the iMagine Marketplace component of the Platform. This allows users to choose and run the trained AI models for inference on the connected back-end cloud resources of the iMagine infrastructure (corresponds with approach 1, "Marketplace inference service delivery", described deeply in **D3.3**). Moreover, the main advantage of this option is that the operational responsibility of the platform is on the project. However, users will need an EGI Check-in account to access the OSCAR platform and the computing resources provided for inference are limited. Currently, there are two

[62] https://docs.ai4os.eu/projects/deepaas/en/stable/
[63] https://docs.oscar.grycap.net/

different OSCAR clusters available for the iMagine project, and the details are collected in **Table F**.

- Deploy your own OSCAR cluster: use case partners can decide to have their instance of the OSCAR framework, deployed by the project partner on computational resources from the iMagine consortium or even on third-party external compute resources. Support and documentation are offered in this case to deploy the OSCAR open-source platform, but the management of the platform relies on the project partner.

Table F - Details of the OSCAR inference platforms available for the iMagine project

Cluster endpoint	Description	Resources	VOs supported	Related endpoints
https://inference.cloud.imagine-ai.eu	Cluster shared between AI4EOSC and iMagine	• Frontend: 4 vCPU, 8 GB • Working Nodes (4): 4 vCPU, 8 GB • Site: NCG-INGRID-PT	vo.ai4eosc.eu vo.imagine-ai.eu	• **MinIO endpoint**[64] • **Console MinIO endpoint**[65] • **Kubernetes dashboard endpoint**[66]
https://inference-walton.cloud.imagine-ai.eu	Cluster dedicated to the iMagine project	• Frontend: 6 CPUS, 16 GB RAM, 30 GB disc • Working Nodes (4): 6 CPUS, 16 GB RAM, 30 GB disc • Site: Walton Institute	vo.imagine-ai.eu	• **MinIO endpoint**[67] • **Console MinIO endpoint**[68] • **Kubernetes dashboard endpoint**[69]

Processing streams of data

For applications that need to process streams of data, and want to avoid the phase where the algorithm loads the weights of the AI model for each file, the iMagine project offers mainly two options:

[64] https://minio.crazy-kowalevski5.im.grycap.net/
[65] https://console.minio.crazy-kowalevski5.im.grycap.net/
[66] https://crazy-kowalevski5.im.grycap.net/dashboard/
[67] https://minio-inference-walton.cloud.imagine-ai.eu
[68] https://console-minio-inference-walton.cloud.imagine-ai.eu
[69] https://gracious-varahamihira6.im.grycap.net/dashboard/

- OSCAR exposed services[70]: OSCAR allows the secure deployment of auto-scaled services that expose their API, so users can directly interact with it. This is recommended for applications already integrated with the DEEPaaS API[71]. Nevertheless, partners can also decide to design their own service presentation layer and model execution scheme and offer it through OSCAR exposed services. With the latest developments, authentication via Ingress has been enabled to better secure access to exposed services.
- The training Nomad cluster: partners can also deploy an inference endpoint at the training Nomad cluster.

With these approaches, models will only load the weights once. Thus, the primary advantage of this approach is its readiness; once the endpoint is established, the model remains loaded in memory, enabling rapid responses to inference queries. However, these methods do not release CPU and RAM resources, as they remain allocated even when the service is idle. Therefore, these approaches are particularly useful for applications that require real-time predictions rather than batch processing. Notice that this pattern is related to approach 1, "Marketplace inference service delivery" and 3, "Inference service delivery", described with more details in **D3.3**.

Processing historical data

Another pattern we have detected is the need to process historical data, consisting typically of a significant number of files that need to be processed with the AI model. For this approach, partners can benefit from OSCAR Batch[72]. This is a tool designed to perform batch-based processing using the OSCAR framework. OSCAR Batch includes a coordinator service where the user provides a MinIO bucket containing files for processing. This service calculates the optimal number of parallel service invocations that can be accommodated within the cluster and distributes the image processing workload accordingly among the services. It ensures the efficient use of available CPU and memory resources in the OSCAR cluster. Notice that both the MinIO storage instance and the OSCAR cluster are provided for iMagine partners (details in **Table F**).

Composing inference pipelines involving several AI models

Additionally, iMagine project partners can also benefit from low-code composition tools, to graphically compose inference AI pipelines through the AI4Compose[73] component. AI4Compose is a framework responsible for supporting composite AI by allowing the workflow composition of multiple inference requests to different AI models. This solution relies on Node-RED and Elyra, two widely adopted open-source tools for graphical

[70] https://docs.oscar.grycap.net/exposed-services/
[71] https://github.com/ai4os/DEEPaaS/releases/tag/v2.4.0
[72] https://github.com/grycap/oscar-batch
[73] https://github.com/ai4os/ai4-compose

pipeline composition, employing a user-friendly drag-and-drop approach. Node-RED, in combination with Flowfuse[74] to support multitenancy, serves as a powerful graphical tool for rapid communication between different services; meanwhile, Elyra provides a visual Notebook Pipeline editor extension for JupyterLab Notebooks (support available in EGI Notebooks[75]) to build notebook-based AI pipelines, simplifying the conversion of multiple notebooks into batch jobs or workflows.

AI4Compose is integrated with OSCAR, being able to invoke the services pre-created in the platform (users can use for that the project's OSCAR serverless inference platforms (Table F), or their OSCAR instances). The integration with OSCAR is made through flow and node implementations offered as reusable components inside both Node-RED and Elyra visual pipeline compositors. With AI4Compose, users will gain agility and resource efficiency as they can leverage the management of the computing platform to OSCAR, which provides a highly scalable infrastructure to support complex computational tasks. Furthermore, AI scientists can easily design, deploy and manage their workflows using an intuitive visual environment, reducing the time and effort required for the maintenance of inference pipelines.

Retraining AI models

Finally, as described in **D3.3**, users may request a retraining of an already trained AI model using their data to make the model more precise for the specific classification/prediction cases they are facing. This can also be supported by the project, which corresponds with approach 4, "Retraining service delivery", described with more details in **D3.3**.

AI model Drift Tools

In aquatic imaging services, such as those used to monitor marine life or underwater environments, conditions such as lighting, water clarity, dirt on the camera and species behavior can vary significantly. These variations can lead to a deterioration in model performance if not properly controlled. Drift tools are important in AI, particularly in production settings and aquatic image services, as they identify and manage the gradual decline in model performance. In AI models, data drift happens when the distribution of input data shifts from the model's original training data, causing a drop in accuracy and reliability. Drift tools enable continuous monitoring and allow early detection of shifts in data distribution or model performance. This ensures that models remain accurate and reliable, leading to more effective AI systems in such a dynamic environment.

[74] https://forge.flows.dev.ai4eosc.eu/
[75] https://notebooks.egi.eu/

Frouros[76] [77] is a drift detection tool for machine learning systems that was developed as part of the AI4EOSC project. It aims to identify significant changes during inference, such as concept drift (changes in the learned concept) or data drift (changes in feature distributions), which can degrade model performance. By detecting these changes, users can determine whether the model predictions are unreliable due to changing feature-target relationships, data changes or problems in the data acquisition pipeline.

Frouros is implemented in Python and supports both concept and data drift detection with 32 detectors, including classical and state-of-the-art methods. It outperforms existing libraries in the variety and number of detectors offered. Frouros is designed for simplicity and offers:

- Datasets: Real or synthetic datasets for testing purposes.
- Detectors: Organized into categories for concept drift and data drift.
- Callbacks: Allows execution of custom code at key stages, similar to PyTorch Lightning and Keras.
- Metrics: Contains evaluation metrics such as prequential error metrics to assess detector performance.

Frouros provides an easy-to-use, comprehensive toolkit for ensuring the reliability of AI models in dynamic environments.

An overview of Use Case Experiences

UC1 Marine Litter Assessment

In this use case, the main objective is to classify drone images in a tile-wise approach. This process consists of two steps: first, by detecting litter accumulations, and then by identifying the present litter categories more specifically. A key aspect of this task was aligning the original classification categories with EU guidelines for litter monitoring strategies and refining the previously existing processing methodology.

For data labelling, we used a pre-labeled dataset and conducted tests to align these initial categories, which were designed for AI application and drone images, with official EU litter lists, particularly the JLIST[78]. The original idea was to translate the existing labels into JLIST categories. However, test results revealed that the detailed JLIST, which is tailored for manual on-site beach litter assessment, is not suitable or translatable for use with drone footage. Reasons for that include the image resolution, which is not high enough to make a detailed, manual identification. In addition to that, the images

[76] https://github.com/IFCA-Advanced-Computing/frouros
[77] Céspedes Sisniega, J. & López García, Á. Frouros: an open-source python library for drift detection in machine learning systems. *SoftwareX*, 26:101733, 2024, doi:10.1016/j.softx.2024.101733.
[78] https://mcc.jrc.ec.europa.eu/main/dev.py?N=41&O=459

contained a lot of overlapping objects, as well as fragments of objects, that are not identifiable from images only.

The main performance metrics we focused on were the standard ones for classification tasks: Accuracy, Precision, Recall and F1-Score.

In terms of preprocessing, we resize the input tiles to standard sizes of 128×128 and 64×64 pixels. We also applied augmentations, including horizontal flips, vertical flips, and random rotations. We did not specifically perform any cleaning of the dataset or handling of noise or anomalies. We investigated whether the unbalanced nature of the dataset influences the model performance. This was achieved by comparison of the Precision, Recall and F1-score for each individual class in the dataset. The result showed stable and consistent values for the underrepresented classes as well as for the classes that have more images present in the dataset. This is why we did not apply any form of data balancing.

For the classification tasks, we used two Convolutional Neural Networks (CNNs) to detect and identify plastic litter. We tested four state-of-the-art CNN models for plastic litter detection, comparing larger models with smaller ones, in terms of their number of parameters. In the category of the smaller, more compact models, we found MobileNetV2 to have a higher performance than SqueezeNet1.1. In the second category, more complex CNN models with a higher number of tunable parameters, the result indicated that DenseNet121 was the best performing model, compared to ResNet50.

Our training datasets are published on Zenodo, and the trained AI models are shared via iMagine marketplace. For deployment, we are using the standard strategies within the iMagine project, with processing methodology and model inference available as a Docker container on the AI4OS Dockerhub and GitHub. Additionally, we are integrating the models into OSCAR for easy-to-use inference runs.

Currently, we have not implemented a drift detection tool, as we do not anticipate a continuous stream of data that would require such monitoring.

UC2 ZooScan - EcoTaxa Pipeline

The service is a complete rewrite of the existing ZooProcess software, using modern libraries. It adds AI components to

1. sort images containing more than one object and then
2. separate plankton objects touching each other on such images, as the subsequent analysis (consisting of classifying images into detailed taxonomic groups and measuring specific features) requires only one object per image.

The processing starts from a large scanned image of a preserved plankton sample on a dedicated instrument (the ZooScan). The initial preprocessing steps on the full images, such as background subtraction, make the segmentation “easy” in the sense that simple

thresholding can be used. But this creates cases of under-segmentation when objects touch each other. This is where semantic/instance/panoptic segmentation models can improve over regular computer vision approaches.

The first step is to identify the under-segmented images; for this, we trained a binary classifier: unique vs. multiple objects per image. We used a MobileNet network pre-trained on ImageNet and fine-tuned it with our data. MobileNet was chosen thanks to our experience in the classification of such images, which indicated that such small and efficient networks are sufficient to capture the needed characteristics in the relatively small, sparse, and grayscale plankton images.
To prepare the training set, we took advantage of existing datasets in which images of multiple objects were already sorted separately from the rest.
The preprocessing steps, before the network, were standard data augmentation (rotation, resizing, cropping, etc.) and the resizing to a common input size was done dynamically, as part of this data augmentation. We used just in time scaling because it was easy to implement that as one operation within the data augmentation chain.
We repeated the experiments a few times, but since model training was still relatively short, it was not relevant to scale all images beforehand.
The data was imbalanced, with many more images of unique objects than of multiple objects. To address this, we used a combination of resampling and weights to actually bias the classifier towards the prediction of multiple objects. Our goal was to maximize the recall of these instances, ensuring that as many of them as possible are available for the next step, that is segmentation.
The evaluation metrics were the binary cross-entropy (i.e., the loss used for model training), the overall accuracy and the recall and precision of the "multiple" class.
We did not use any specific tool to track the experiments, since the training was quite straightforward.

For segmentation, we tested instance segmentation using MaskRCNN and panoptic segmentation using Mask2Former. We obtained better results with the latter. Yet, the masks produced by any deep network do not match the object contour pixel-per-pixel, which is a problem to exploit information about the shape of the contour for example, and are not as reproducible as a simple threshold-based segmentation. To get the "intelligence" from a deep model and the reliability of a deterministic computer vision approach, we use the deep masks to define the centroid of regions. These regions are then expanded with a watershed algorithm[79] to cover all non-background pixels on the image based on a simple threshold segmentation. In the end, this produces masks whose borders match the threshold-based segmentation, but that are separated among different regions (i.e., the original detections of the deep network), which are (or should be) the different objects.
As for classification, we assembled the training dataset from existing data, in which human operators had manually traced white lines to separate touching objects. We

[79] https://scikit-image.org/docs/stable/auto_examples/segmentation/plot_watershed.html

therefore had a ground truth of multiple objects and where they should be separated from one another. We used all examples in the dataset because they all correspond to realistic situations, even the more extreme ones. Furthermore, we acknowledge that the rarer cases (e.g., large jumbles of many objects sticking together) are not well handled in the predictions. However, the solution is not to remove these cases from training. Instead, we should provide more examples of such cases to create a more balanced dataset. No data augmentation was applied so far, and we are not aware of weighting schemes in the case of segmentation.
The dataset was published on the SeaNoe platform, with a metadata record in Zenodo.
The evaluation metrics were Intersection Over Union (IoU) relative to the manually defined ground truth, as well as the difference in the number of retrieved objects between the ground truth and the result after the watershed.
No specific experiment tracking tool was used; the results of various attempts were collected in a large spreadsheet.

For both components (classification and segmentation), the trained models and associated code will be shared using the standard procedures in iMagine (GitHub repository in the ai4os-hub organization, Marketplace, etc.). For the production service, we will primarily use the dedicated inference service, through OSCAR. This service provides fast and scalable inference capabilities that we need for this use case. Users will mostly access it through a third software component, which we are also developing. This component will provide a Graphical User Interface to acquire the scan, enter metadata, invoke the AI components, and upload the result for further processing in the EcoTaxa[80] application.

We did not yet consider drift detection tools but, at some point, we will apply the model trained on the ZooScan images from one kind of plankton net to those from another plankton net. While the imaging instrument and settings stay the same (and the output is checked to be very consistent), the two nets yield different taxa and organisms of different size. The intensity of the (expected) decrease in performance will inform us on the generalization potential of our current model. If the performance degrades by more than a few % of IoU, we will retrain a single model but with example images from several nets.

UC3o Marine Ecosystem Monitoring at EMSO OBSEA

The OBSEA is an underwater observatory that has been acquiring multiparametric data since 2009. Over the years different cameras have been deployed, having a vast archival of pictures starting from 2011. This data has historically been manually analyzed to extract ecological information of the fish community present in the area, which is very time-consuming and repetitive. The main purpose of this use case is to use AI

[80] http://ecotaxa.obs-vlfr.fr/

techniques to automate the detection and classification of the fish specimens captured by the cameras at OBSEA and provide a timeseries of fish detections. This data can be easily processed by scientists who can spend their time doing actual science instead of counting fish picture by picture.

Although there was a lot of historical data, most of the cameras had biofouling issues and had very low resolution. However, from July 2023 several new HD cameras were deployed. Taking advantage of these cameras a new dataset with high quality images has been built.

For the image labeling, several tools were tested, such as roboflow, label studio, biigle, etc. Although some of these tools have interesting features most of them have bugs, are quite slow or require subscriptions. Thus, the chosen solution was labellmg, a simple but effective python package to label pictures. At first glance it doesn't have astonishing features, but due to its simplicity it is very easy to complement with custom-made scripts for dataset management and semi-automatic labeling.

Once a small dataset was labeled, several AI algorithms were tested, such as Faster-RCNN and YOLOv8. The latter proved to be a balance between precision and speed, which makes it ideal for our application. Additionally the different flavors of the algorithm (nano, small, medium, large, xlarge) makes it possible to train different versions of the algorithms with different speed/precision ratios. In our case we use nano for real-time video inference (dissemination purposes) and xlarge for slower-but-precise inference in pictures (scientific purposes)

In order to cover the seasonality of the fish community and to avoid biases in our training data, it was decided to produce a year-long dataset from July 2023 to July 2024 with the newly installed HD cameras. Due to the nature of the ecosystem, at the beginning the dataset is highly imbalanced, the species that are the base of the trophic chain are hundreds or even thousands of times more frequent than big predators. To mitigate this, we complemented our dataset with pictures from the MINKA[81] community in those uncommon species. At the moment of writing this deliverable the our dataset is composed of 5354 images grouped in 21 classes with more than 35k labels.

The training of the algorithm was performed in the iMagine platform, using the default module and the command-line interface. In order to improve the performance of the model, the YOLOv8 built-in data augmentation techniques were used. The hyperparameters configuring the data augmentation were optimized following the official documentation. Once a first functional version of the algorithm was achieved, a module for inference was uploaded at the iMagine platform marketplace, called obsea-fish-detection[82].

[81] https://minka-sdg.org/

[82] https://dashboard.cloud.imagine-ai.eu/marketplace/modules/obsea-fish-detection

To compare different versions of the algorithm the mAP@50 metric was chosen. This metric is the mean of the average precisions of all classes with a IoU (intersection over union) of 50%. In our application both precision and recall metrics are equally important, so the mAP@50 provides a good overall performance evaluation. To track the experiments ClearML and MLflow platforms were used to track the training experiments. The latest experiment achieved a mAP@0.5 of 0.866.

The most challenging part in preparing our data was to deal with the nature of underwater pictures, specially the water turbidity. The turbidity in sea water blurs objects that are a few meters away from the camera, making it difficult even to the human eye to clearly identify blurred fish specimens. Thus fish in the pictures appear only as a smudge in the background. As a tradeoff it was decided to label those fish that had some feature that could be used to classify the species (e.g. shape of the tail, stripes pattern, etc.).

Next steps include deploying two models into production: the full-fledged version for scientific data production and the nano version that allows for faster inference (at the cost of performance) to have real-time video inference.

It is also planned to perform inference to the whole data bank starting from 2011 to extract biological information. For this task it is envisioned to use the OSCAR framework for large-scale batch inference. The docker containers have already been produced and a large-scale test is planned for the following weeks.

UC3a Marine Ecosystem Monitoring at EMSO Azores

The data is from a project called DeepSeaSpy[83], where images of deep sea habitats are annotated by citizens. The goal is to automatically identify animals based on the annotations from citizens, with the training of a deep learning neural network (Yolov8). Because the data was annotated by citizens, it will be necessary to deal with problems specific to citizen science, meaning a cleaning step on the dataset is mandatory to ensure that the Yolov8 models can effectively learn from it.
There are 3979 images, and 253 323 annotations in the untouched dataset. Of which, 3403 images are from the Juan de Fuca ridge, and 576 are from the Azores.
With 15 different species, there is a non-avoidable data imbalance in the dataset depending on the citizens labelling preferences, which is heavily leaning toward species that are the easiest to identify. It was decided to focus the next steps on two specific species (*Buccinidae* and *Bythograeidae*), mainly for the following reasons :

[83] https://www.deepseaspy.com/en

- Those two species are from different habitats (*Buccinidae* in Juan de Fuca, *Bythograeidae* in Azores), and they are the ones that have the most annotations respectively to their habitats.
- The imbalance with the other species makes it difficult to train a multiclass model with good performance on all the species, and because of the quality of the images, a multiclass model may raise identification confusions between species of the same habitat.
- Species that are harder to identify have a much higher level of incorrect identifications by citizens, which dampers the correctness of the identifications from the trained model.
- Because the images are very different between the two habitats (different layout, composition, different species…), it was important to have one species representative of each habitat.
- Each species may need specific cleaning solutions, since bounding box areas/annotation densities change between species. Focusing on a few classes may help make better choices without compromises for the cleaning step.

Corrections were added to the original annotations, based on problems encountered and potential performance enhancements for the model. For instance, the way bounding boxes were labelled was corrected (from lines to rectangles), while making sure the angle of the lines would not create unusable rectangle bounding boxes. Furthermore, a padding of 25 pixels was added to one of the two species of interest (Bythograeidae) as better performance was seen from the Yolov8 trained models.
The main issue encountered was the redundancy of the annotations. It was decided to go with a 2-step cleaning approach, using the redundancy as a way to cross validate the presence of an animal.
First, the pipeline unifies bounding boxes based on an IoU threshold : bounding boxes that have an IoU of 0.6 or higher with any other bounding box are kept. The others are discarded. Then, a measure of the minimal IoU between bounding boxes that are still overlapping is done. If they are of 0.4 or higher, they are kept in the final dataset. This way, the dataset was reduced to 20 979 annotations. Those specific IoU thresholds were chosen after numerous tests, visual confirmations and model performances. The best models were selected by trying different cleaning parameters. Model experiments are tracked by keeping files locally, and having documentation on the work done.
It is still planned to carry the automatic identification on other species, but it may be better to wait for more images or annotations to do so.
The untouched Buccinidae dataset has 98 282 annotations. After the 2-step cleaning process, those numbers were reduced to 14 662 annotations.
The untouched Bythograeidae dataset has 2426 annotations. After the 2-step cleaning process, those numbers were reduced to 305 annotations.
The difference between the number of annotations for Bythograeidae and Buccinidae is mainly explained by the difference in the quantity of images.
To evaluate performances, the results of the training of Yolov8 models were used.

The metrics used to evaluate the models were mainly the precision, recall, and mAP50-95 metrics.
For the training on Buccinidae, expert annotations were used as validation for the training on citizen annotations. The best model has a precision/recall of 0.6, but low confidence on its detections (mAP50-95 at 0.15 vs. mAP50 at 0.5, meaning there is a higher precision with lower confidence).
For the training on Bythograeidae, no expert annotations were available for the validation of the training on citizen annotations, so a part of the citizen dataset was used as validation. The best model has a precision/recall of 0.4, and like the Buccinidae model, a low confidence in its detections (mAP50-95 at 0.1 vs. mAP50 at 0.3, meaning there is a higher precision with lower confidence).
The dataset and the trained models are available on SeaNoe, with a Zenodo link (**https://zenodo.org/records/13759095**) to make it as accessible as possible. The AI models will be deployable from the iMagine module named Deep Sea Detection. No use of a data drift tool is being considered as the species detection is very specific to the sample location. No other deep sea sample location has the same species repartition as the one in this use case.

UC3s Marine Ecosystem Monitoring at EMSO SmartBay

In EMSO Smartbay we are looking at 3 use cases, Marine Species detection at the Smartbay Underwater Observatory, Video Quality Assessment of the Smartbay Observatory video feeds (and video Archive) and Nephrop burrow detection for prawn fishery surveys.
In the Marine Species detection and prawn burrow use cases, we have used a "self-hosted" instance of CVAT for Image Annotation. Images are collated and loaded to a self-hosted instance of a Minio[84] Object store. In Minio the images are stored in S3 compatible storage buckets, these are effectively "Cloud Storage" accessible via http urls and are added to CVAT as S3 storage locations. Folders of images are then used to create annotation "Projects" and annotation "Tasks". The Images in a task are then manually annotated in the CVAT web interface, using bounding boxes labelled according to the object classes defined in the CVAT project. When annotation tasks are complete, the Annotation data (images and labels) can be exported from CVAT then in COCO format. We then upload the task datasets to Roboflow projects for collation, training dataset analysis and preparing the training data for export in YoloV8 format.
We have focused on training YOLOv8 object detection models for the Marine Species and Nephrop Burrow data and have used an "On Premise" GPU for training but will also use the iMagine platform for further model training runs. We have also begun to use the MLFlow instance on the iMagine platform to record Model Training runs and hope to use this to gather metrics on training and model performance and explore what dataset and training tuning can result in better performing models.

[84] https://min.io/

In EMSO Smartbay we have found it difficult to find and obtain published versions of annotated North East Atlantic Marine Species datasets or prawn burrow datasets in usable annotated formats. We have supplemented local imagery with CC-BY licensed imagery from the MINKA portal[85] a citizen science-based public repository of Marine Species pictures. We have used a copy of the minka downloader tool[86] developed by EMSO-OBSEA to download images for target species. While this data isn't annotated with bounding boxes, it is useful as it has community-identified reference images of species which can be annotated and used in our training datasets.
We have used student summer bursars for the bulk of our annotation work so far, but there is a desire in the Marine Institute to continue to strive for and develop an easy to use on-premise platform or solution for collating and annotating imagery datasets.
We have used the "Histogram Equalisation" tool in CVAT when annotating data. The CVAT annotation environment has a "Histogram equalisation" Tool, that is implemented in CVAT using a javascript port of OpenCV. This is an algorithm for improving the contrast of an image, so features can be better distinguished in overexposed or underexposed images (very bright or very dark images). The image is only enhanced for the annotator to better distinguish features, it doesn't alter the image for training. Our bursar students did, however, use "noise" and "exposure" in Roboflow to augment some training datasets for Nephrops and Marine Species.
For the Nephrops Burrow Detection use case one of the Student bursars annotated 3331 image frames from a number of different prawn Fishing ground stations, this was done to get a variety of burrow images from the different fishing area seabed types.

Five annotation classes were used: prawn burrow; small prawn burrow; crab burrow; closed burrow; gate kept burrow, which is a prawn burrow with a prawn visible inside. 1171 control images of seabed with none of the labels present were also included resulting in a total of 4502 images for the first dataset.)

However, the resulting dataset was extremely skewed, containing over 5000 annotations of prawn burrows and only 79 gate kept burrows.
The student imported the annotated dataset into a project on the Roboflow platform and used the platform to analyse, refine and more balance the classes in the dataset by reducing the disparity in class numbers between the classes. This brought the total image number down to 468. Roboflow also has a number of image augmentation options, the student chose to use "noise" and "exposure" augmentation to copies of the images to strengthen and expand the dataset, which brought the final image count for the model training dataset up to 1200 images.

Initially there were also 2 "burrow complex" annotation classes which consist of Prawn burrows connected "back to back" or in a "T" shape pattern. These "burrow complex" classes were removed as it appeared to confuse the model.

[85] https://minka-sdg.org/
[86] https://github.com/obsea-upc/minka-downloader

Before the student finished they had begun looking at an approach in python to using YOLOv8 Oriented Bounding Boxes and Object tracking and counting functionality in YOLOv8 to try and look at the angles of burrows in relation to each other, to better determine and count "Burrow Complexes" in video sequences.

For our Video Quality Assessment use case, we have focused on implementing a "Proof of Concept" with the pre-trained DOVER VQA[87] model. This model has "out of the box" proved useful in scoring Videos for technical and Aesthetic quality. In our limited experimentation with the model so far we have found the scoring system useful in identifying poor and good quality video footage (**Figure A**).

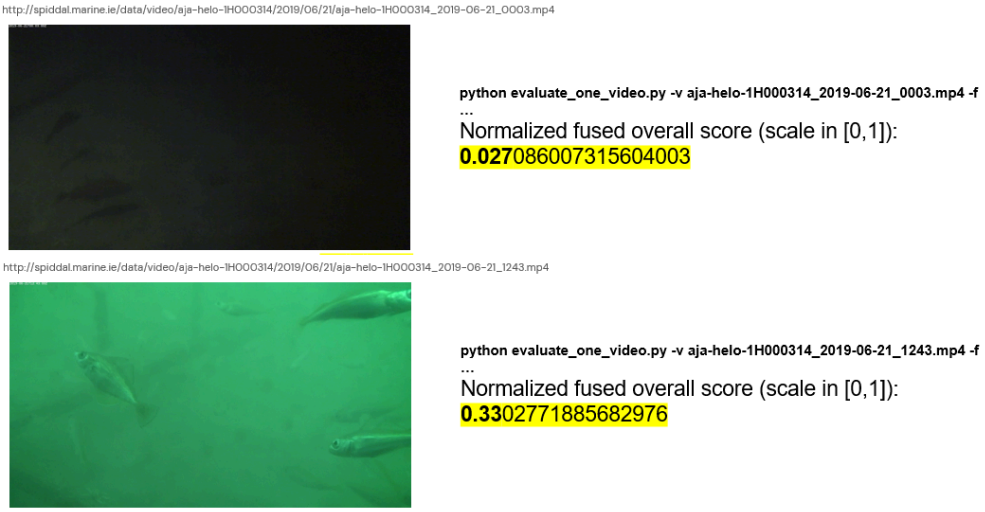

Figure A - example DOVER VQA algorithm video quality scoring of 2 minute smartbay video files

UC4 Oil Spill Detection

In UC4, the consortium consisting of Orbital EOS, the University of Trento (UniTN), and CMCC is working towards creating an end-to-end, AI-enhanced oil spill detection system. To achieve this goal a system that detects oil spill from satellites using AI (ORBITAL EOS) was implemented and recorded several incidents around the world. This dataset was organised and made available at Zenodo[88] and THREDDS[89], making all these observations publicly available (University of Trento). Using this dataset, an AI hybrid modelling approach was implemented, using Medslik-II oil spill model and a BayesOpt, an AI driven parameter search, to optimise the results of simulations using two oil spill observations.

[87] https://github.com/VQAssessment/DOVER
[88] https://doi.org/10.5281/zenodo.11354663
[89] http://thredds.imagine.disi.unitn.it

Initially, satellite imagery is collected and labelled internally and proprietary models developed by Orbital EOS[90] are applied to these datasets. These models identify oil spills in the available images, generating estimates of the spill type (i.e., mineral or biogenic), location, area, and volume. Active (i.e., Synthetic Aperture Radar) and passive (i.e., optical) sensors are used to sample the ocean surface and model outputs are delivered in standardised format. These estimations are then used in subsequent steps of the processing chain, which includes generating real or hypothetical oil spill scenarios. This process accounts for potential false positives, where an oil spill is detected but did not actually occur in the real world.

From this approach, a dataset of events ranging from 2018 to 2022 was collected and systematised into a database by the University of Trento. This dataset contains around 300 events related to 172 oil spill images from around the world.
A THREDDS catalogue has been deployed at UNITN premises to support environmental scientists from various disciplines in browsing and accessing the provided datasets as Open Data[89], for any purpose they may have. The THREDDS catalogue includes both oil spill images and NetCDF data (Medslik-II simulation outputs) available also in JSON representation. The entire dataset provides a benchmark baseline for future research activities, and it is expected to be further extended in the future.

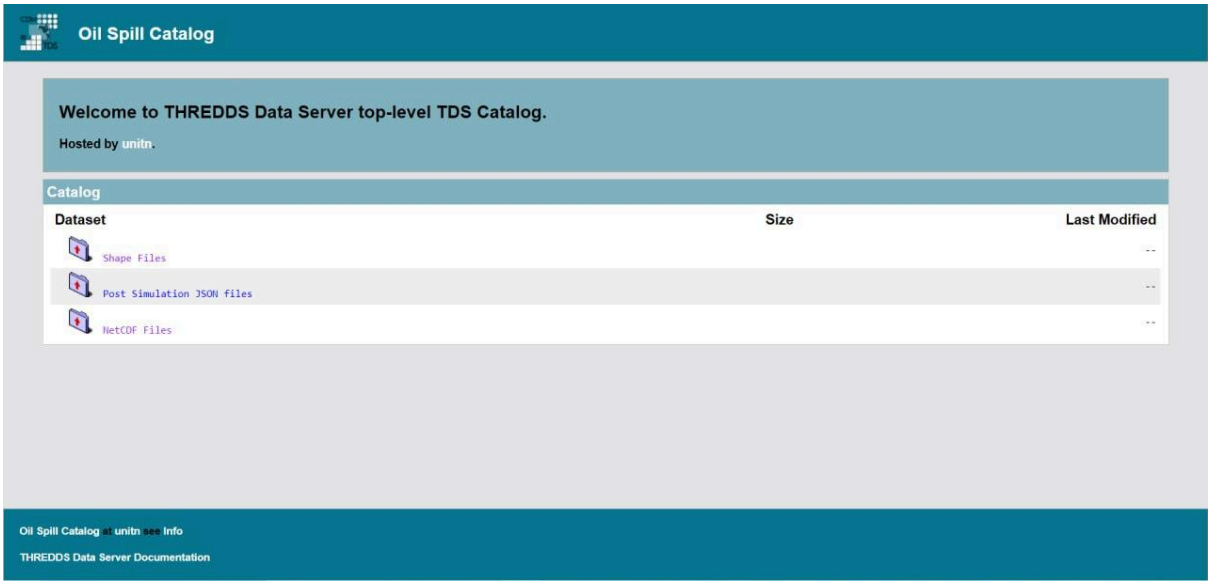

Figure B - Oil spill catalogue via THREDDS service

Based on the oil spill image detections dataset, CMCC is working on the integration of these components in an end-to-end pipeline which includes an AI-driven approach to improve the accuracy of oil spill events simulated by the Medslik-II numerical model. Numerical simulations rely on a set of physical parameters whose values affect the shape of the simulated spill in space and time.

[90] https://www.orbitaleos.com/

In UC4, as an AI approach, we use Bayesian Optimization to scan the physical parameters space to sample the best configuration, i.e., the one maximising a specific score function. For this part of the use case, data labelling is not needed, because the images were already provided by Orbital EOS.

We use the Fractions Skills Score (FSS) to quantify how closely the simulation matches the segmented oil spill images. Therefore, we set up an optimization framework in which the simulation is repeated several times, with the goal of maximising the FSS by exploring a set of physically consistent parameters, which will then be used to initialise the model.

The individual simulations are then compared and the best one is chosen as the best set of parameters. Considering the nature of the approach, experiment tracking was not necessary because each set of images plus the environmental conditions can make the results change from one to another event.

At the end of the optimization, the parameters that maximise the metric are identified, producing an optimal simulation that returns a visually representative image of the oil slick.

Due to all these facts described above, the solution does not use a machine learning model itself, so at the end, the results are outputs from a physical model with AI enhanced parameters. Also, data drift will not be considered for this approach because not only the images are important, but also the environmental conditions, which naturally add uncertainty in the optimization. Furthermore, the oil spill model will be in constant development, which can also alter the way the BayesOpt algorithm works.

This framework will be made available at the iMagine marketplace and also at our current production system[91]. The hybrid model will be focused on advanced users that know how to use satellite imagery and shapefiles to perform the optimisation on their own.

UC5 Flowcam Plankton Identification

FlowCam is a high-throughput imaging device producing between 300-400,000 particle images on a yearly basis in our current monitoring context. To store and analyse this high volume of data we established semi-automatic data pipelines to preprocess raw output data and store in an internal database. Convolutional Neural Networks, based on an Xception architecture[92] have been trained on manually validated FlowCam images of phytoplankton cells to speed up identification of particle images. Images are manually checked by scientists post inference through an in-house developed labelling tool to ensure data quality. The current model used was trained using a training-set consisting

[91] http://witoil.cmcc-opa.eu/
[92] https://ieeexplore.ieee.org/document/8099678

of 337 613 Images spread over 95 classes (training set available on Zenodo[93]). A module of the Flowcam phytoplankton identification service has been built on the iMagine platform. The user has the option to split the train/validation/test according to their preference and to use data augmentation using the Python albumentations package (Buslaev et al. 2020). The categorical Cross-Entropy loss function was used for the phytoplankton classes. The full monitoring image library of 1,865,953 manually validated FlowCam images targeting eukaryotic microphytoplankton in the 55-300µm range is openly available in the Marine Data Archive (MDA) and linked to an IMIS discovery dataset record (Integrated Marine Information System) available via Lagaisse et al., (2024)[94]. A known challenge in dealing with this type of imaging datasets is a large class imbalance, with a few excessively abundant classes and many rare classes with a small number of images in the dataset. To overcome this, thresholds per trained class are established in sampling of the training set from the image library in model training, with a minimum of 100 images per class and a maximum number of images of 10,000. Augmentation techniques are implemented to artificially upsample (rare) classes, and available through the FlowCam module in the iMagine platform. Additional functionality to increase applicability of models and training dataset across FlowCam device versions are included in the module as well by providing code for image transformation to deal with differences in image resolutions and RGB or grayscale images. To assess model performance after training, notebooks are made available to the user to assess simple metrics like precision, recall and F1 score on a class based level. Currently, drift monitoring is not included, as models are trained on a ground truth and regularly retrained on yearly incoming validated monitoring data. Drift monitoring however, could be considered in the future.

UC6 Underwater Noise Identification

The model's objective is to predict the distance to the closest vessel. However, challenges arise due to vessels that are either dark or exhibit irregularities in their AIS (Automatic identification system) transmissions.

To address this, we employed a strategy to connect local minima (i.e., the closest vessels) within a specified window frame. This method allows for continuous annotation of the recordings based on the distance to the nearest AIS data point. The choice of window frame size is crucial: a smaller window frame increases the dataset's temporal resolution by providing more data points but also raises the risk of missing the closest vessel if there are gaps in AIS data transmission. Conversely, a larger window frame reduces the risk of missing the closest vessel due to intermittent data but at the cost of decreased temporal resolution. Therefore, selecting an optimal window frame is essential to balance the trade-off between data resolution and accuracy in vessel identification.

[93] https://zenodo.org/records/10554845
[94] https://doi.org/10.14284/680

Through experimentation and filtering, a 6-minute window frame was predominantly used. However, a transition to a 5-minute window frame was implemented during the latter half of the year for the Grafton data. This adjustment resulted in 240 and 288 data points for the 5-minute and 6-minute windows, respectively, within a 24-hour period. Despite these efforts, approximately 8% of the data had to be excluded due to irregularities in AIS data.

Features were extracted using the autoprocessor from CLAP, employing a window size of 1024, a hop size of 320, and 64 mel bins for computing STFTs and generating log-mel spectrograms. Each 10-second audio file produced input features with a size of (1001, 64).
Ultimately, 26,465 WAV files were generated over 116 days, including 40 days with overlapping stations and 76 unique days.

The dataset was initially divided into training, validation, and testing sets. To ensure data independence, full-day deployments from different locations and dates were used for each set (i.e., data is independent within the same day but not across different days). This means that for any given day and location, the data is entirely allocated to either the training, validation, or testing set, with no overlap across these sets for the same day. The distribution split is 79.4%, 10.6%, and 9.9% due to the uneven availability of data points across different days. The data was spread through two stations and throughout most of 2022 to battle data bias.

The wav files were labelled based on AIS data, but occasionally the sampling rate of AIS is too small which would cause some gaps in the interpolation, these were manually filtered out (about 8% of total data). The data has not been published yet.

Subsequently, all 10-second audio files were converted to mono with a sampling rate of 48 kHz. Features were extracted using the autoprocessor from CLAP, employing a window size of 1024, a hop size of 320, and 64 mel bins for computing STFTs and generating log-mel spectrograms. Each 10-second audio file produced input features with a size of (1001, 64).

To visualize the computed input features, UMAP was used to project the data into two dimensions. Data points were color-coded based on distance and speed. It revealed a clear pattern related to distance, while the speed data is less distinct. Consequently, the model focused solely on distance.

To visualize the computed input features, UMAP[95] was used to project the data into two dimensions. Data points were color-coded based on distance and speed. It revealed a

[95] https://arxiv.org/abs/1802.03426

clear pattern related to distance, while the speed data is less distinct. Consequently, the model focused solely on distance.

We used categorical Cross-Entropy as the loss function but adjusted it to account for the fact that our categories (distance ranges) are not independent. The model predicts distance classes: 0-1 km, 1-2 km, ..., 9-10 km, and 10+ km. To improve the loss function, we introduced a weighted penalty based on the severity of the prediction error. For example, a prediction that is off by 1 km is penalized less than one that is off by 5 km. Instead of standard one-hot encoding, the loss function was adjusted to reflect the degree of the error.

During training the current RMSE was shown to track the experiments. At the end, the model achieves a Root Mean Squared Error (RMSE) of 1.76 km over a range of 10 km. Where most values are near the diagonal of the confusion matrix, indicating high precision. Some underestimations may be due to inaccuracies in the AIS data, suggesting that the model could be correcting errors in the AIS data. Thus, predictions showing shorter distances compared to AIS values could represent a more precise reflection of the true distances.

The model has yet to be published together with the data, but is currently running on the imagine platform. Currently, drift monitoring is not included, however, it could be considered in the future.

UC7 Beach Monitoring

The use case aims to process images from beach imaging systems to automate the extraction of important coastal features. This includes determining the shoreline position from crowd-sourced smartphone imagery from CoastSnap sites[96], and identifying beach seagrass wracks (i.e., Posidonia oceanica accumulations) and detecting rip currents from beach video monitoring stations.

In the shoreline extraction case, crowd-sourced images were labelled programmatically using R software (SCLabels dataset). CVAT, on the other hand, was used to label images from beach video-monitoring systems, including a dataset for the beach seagrass wrack identification case (BWILD), and another one for the detection of rip currents (currently in progress). BWILD and SCLabels are both publicly available in Zenodo (Table E - UC7).

All available crowd-sourced images were included in SCLabels as they were already subjected to a quality control (e.g., removed duplicates) and constituted an affordable amount (1717 images). In the BWILD and rip currents datasets, however, a subset of

[96] https://doi.org/10.5194/essd-15-4613-2023

images was selected from a repository containing more than 250,000 images. Image selection was based on unsupervised clustering and the balance of images with absence/presence of coastal features of interest. Before clustering, images were subjected to basic filters such as tests for darkness and sun glint. Additionally, all the selected images underwent a visual inspection to identify and eliminate those containing excessive noise or persistent issues that hindered accurate labelling. Throughout this process, particular attention was paid to maintaining a balanced representation of images captured under varying lighting and meteoceanic conditions. However, in all cases there exists an imbalance in the number of images from different sites, each outlooking a particular beach or coastal stretch with similar geometry, and an imbalance in the temporal distribution of images. These arise from the inability to control factors such as citizen participation in specific areas and times (CoastSnap sites) and the prevalence of coastal features and processes in particular beach areas or seasons (seagrass wracks and rip currents). We did not approach this problem directly.

For the shoreline extraction case, we focused on landwards/seawards segmentation assuming that, if perfectly resolved, the interface between landwards and seawards pixels could be considered the shoreline. To this end, we tested the U-Net and Bi-LSTM networks. For the U-Net training we used patchify[97] to split images of varying sizes into smaller patches overlapped by a patch cell size of 50% and compatible with the U-Net architecture (256×256). Rather than resizing, we chose patchify to avoid the creation of excessively large steps in the shoreline due to pixel interpolation during the downsampling process in the resizing. For the Bi-LSTM training, images were splitted in rows, and each row processed independently by the Bi-LSTM network (589292 rows). To maintain consistent row lengths, rows were equally padded with black pixels on both the right and left sides when necessary. This row-wise approach enabled training models individually for each beach due to its lower requirement for input images compared to U-Net. This separate training strategy significantly enhanced the performance of Bi-LSTM, and proved that it is a suitable approach when training data is limited.

Tackling the beach seagrass wracks identification, we focused on the delineation of wracks of varying densities. We tested the U-Net (segmentation), the YOLOv8-v9 (object detection and mask segmentation) in their different scaled variants, and a combined approach leveraging the object detection capabilities of YOLO and the zero-shot segmentation capabilities of SAM. In the use of YOLO networks we used the random data augmentation setting, which applies a set of image transformations randomly during training including image flips, translation, HSV (Hue Saturation Value) adjustments, scaling, mosaicking, etc. By making the model more generalizable, this approach helps address dataset imbalances, enhancing the model's ability to perform well on a broader range of images. For the SAM segmentation, we used images cropped to the bounding boxes resulting from the 'best YOLO model'. The main limitation in the use of this combined approach is the uncertainty derived from the error inheritance of

[97] https://pypi.org/project/patchify/

YOLO bounding boxes to make SAM predictions. From the comparison of the three approaches, the results suggest focusing on YOLO. However, independently of the approach, models' performance decays in the detection and/or segmentation of low-density beach seagrass wracks. To approach this, further experiments will consider merging low- and high-density wracks classes into one general class.

For model evaluation we used common performance metrics such as accuracy, loss, precision, recall, F1 score and IOU. For experiment management and tracking we used MLflow, which allowed logging model parameters and metrics in real-time, enabling performance monitoring, comparing different experimental runs, and maintaining a registry for reproducible experiments.

Before sharing and deploying a definitive model for each problem tackled in the use case, we aim to conduct additional experiments. This will involve testing different DL architectures, including additional data pre-processing steps, extended data augmentation to address image imbalance, training models for the rip currents case, and consider drift monitoring for the developed models.

UC8 Freshwater diatoms identification

The use case aims to develop a diatom-based bioindication service using automatic pattern recognition algorithms for individual microscope images from freshwater environments.
For our use case, the development of a pipeline for diatom identification was not difficult from a technical point of view. Our bottleneck was the limited size of our initial training dataset. The proof-of-concept of our prototype was done by pre-training the models using a synthetic dataset (virtual microscope images) and then fine-tuned using a limited real dataset (real microscope images). The synthetic dataset was gathered by 1) collecting individual images of diatoms (from atlases (ca 15,000 individual diatom thumbnails representative of ca 200 diatom species, at least 30 images/species), 2) using data augmentation by varying size and orientation of the thumbnails, and 3) creating virtual microscope images using seamless, to paste the thumbnails on a grey background, thus mimicking a realistic set of microscope images containing various diatoms.

Using the synthetic dataset, the performance of the detection network (YOLOv5) was improved by up to 25% for precision and 23% for recall at an Intersection-over-Union (IoU) threshold of 0.5[98].
The diatom thumbnails dataset was used to train a deep learning classifier with EfficientNet as the backbone. The classifier was evaluated based on the accuracy score,

[98] Venkataramanan et al. (2023). Usefulness of synthetic datasets for diatom automatic detection using a deep-learning approach. https://doi.org/10.1016/j.engappai.2022.105594

which was 94%. Of the ca 200 diatom species included in our dataset, 113 were classified with 100% accuracy. We also studied other classification methods to better consider the high inter-class similarity and intra-class variance of the different diatom species. This can make it difficult for traditional classification methods to accurately distinguish between different species. To address this issue, we proposed a method for learning feature representations that can group visually similar-looking images of each class together, while also ensuring that the inter-class features are widely separated from each other. This approach was also tested using a standardized plankton image dataset (WHOI-Plankton Dataset)[99]. Additionally, we introduce a method for estimating uncertainty in classification performance. It involves using the proximity of a data point to different class features to estimate the uncertainty in the network's prediction. We also show how this method can be used to obtain a reliable estimate of the prediction confidence and detect out-of-distribution samples[100]. Standard evaluation metrics were used, namely the classification accuracy, the Expected Calibration Error (ECE), the Negative Log-Likelihood (NLL), the Area Under the Receiver Operating Characteristics (AUROC) and the Area Under the Precision-Recall curve (AUPR). The effectiveness and generalizability of our proposed feature representation and uncertainty estimation method was demonstrated by testing on various standard datasets (COCO, MNIST, CIFAR, SVHN).

To further improve our current models but also develop new ones, a larger dataset of real diatom microscope images is being consolidated. Highly trained diatom experts focused on the taxonomic classification task using BIIGLE (rotating bounding boxes) while low-level experts such as students focused on object segmentation using LabelBox (instance segmentation masks). These datasets will be then used to re-trained our detection and classification pipeline. This new dataset will also be used to test a pipeline based on instance segmentation and morphological parameter extraction from the obtained segmentation masks. This approach will be compared to unsupervised approaches (e.g., GANs) able to explore the morphological variability of diatoms in an unsupervised way. This will be further used by biologists for their morphometric analyses.

To date, the diatom thumbnails dataset has been published on our institutional repository (DOREL), which is connected to the national one[101]. The different models have been published on GitHub, and it is available on the Marketplace of the iMagine Project.

---

[99] Venkataramanan, A., Laviale, M., Figus, C., Usseglio-Polatera, P., & Pradalier, C. (2021). Tackling inter-class similarity and intra-class variance for microscopic image-based classification. In *13th International Conference on Computer Vision Systems (ICVS 2021)*. Virtual Event, Austria. https://doi.org/10.1007/978-3-030-87156-7_8

[100] Venkataramanan, A., Benbihi, A., Laviale, M., & Pradalier, C. (2023). Gaussian latent representations for uncertainty estimation using Mahalanobis distance in deep classifiers. In *2023 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)* (pp. 4490-4499). Paris, France. https://doi.org/10.1109/ICCVW60793.2023.00483

[101] https://dorel.univ-lorraine.fr/dataset.xhtml?persistentId=doi:10.12763/UADENQ

The real images dataset will be released. Currently, there is no need for drift monitoring of the developed model, but it can be considered in the future.