IT UNIVERSITY OF COPENHAGEN

Finding Anemo: Comparing Classifier Performance on Image Embeddings of Marine Species in Danish waters

ALTEA FOGH

alfo@itu.dk

Student ID: 20360

Course Code: KIREPRO1PE (7.5 ECTS)

Supervisor: **Stella Grasshof**

stgr@itu.dk

Date: 15 December 2025

Abstract

To address the challenges set by the Ocean Decade initiative, the MOSAR project aims to assess the impact of artificial reefs on the presence of fish in the harbour of Hundested, Denmark. The data collected is generally used to train classifiers to recognise and count species, but errors in data labelling and discrepancies in taxonomical distinctions can lead to decreased performance. We thus decided to assess the role of using embeddings extracted from the available labelled data to train simple classifiers to recognise the marine species. The performance is compared to a baseline of using a ResNet50 fine-tuned on the image data. We find a simple neural network to be the best model for this task. We finally argue for using this pipeline to address mislabelling by introducing a human-in-the-loop approach that leverages the embeddings for classification of species and for recognition of possible mislabelled samples. By defining a more involved approach, we can solve the original issues and leverage existing models to curate better datasets.

Keywords: Underwater Computer Vision, Embedding Classification, Human-In-The-Loop Approach

1 Introduction

Since 2021, the Ocean Decade initiative [1] has been established to provide a framework for international engagement and collaboration, in order to develop and increase the efforts towards the protection and understanding of the oceans. The policies are designed at all levels of interaction, from global to local, private to public, to align research, investments and initiatives to improve human relationships with the oceans. Technology in many forms is at the forefront of the initiative, to be used for the understanding and monitoring of the underwater environment. Machine learning, and especially Computer Vision (CV), is a crucial part of ocean monitoring, as it allows to process huge amounts of collected data in a short time, to assess the underwater conditions, count fish species and detect positive or negative impacts of human behaviours. This key aspect is highlighted in challenge 7 [2] of the Ocean Decade challenges, while challenge 2 [3] highlights the need to understand positive and negative changes in ecosystems and protect environments and biodiversity.

Automatically detecting species and monitoring the marine environment with machine learning is a complex task, both due to the sheer amount of data collected and due to the varying quality of the data. Marine images are often impacted by water turbidity, variation in light, and presence of debris [4]. Human annotators face challenges due to these issues, causing misclassifications that are then used as training data, rendering the task even more complex. Automated tools can therefore support marine monitoring by providing initial predictions that can then be refined by human experts. In this project, we are going to discuss how to introduce a human-in-the-loop approach that would allow for correcting any misclassifications from the model, while leveraging the speed of automated systems and existing foundation models. Data enhancement techniques could be leveraged to improve visibility [5], but it would require additional, possibly computationally expensive steps; we thus want to assess the performance of our system without them first.

In Denmark, the MOSAR (Monitoring of Small-Scale Artificial Reefs) project [6] represents one of the ongoing initiatives aimed at tackling the Ocean Decade Challenges. The project uses underwater cameras which are directed towards artificial reefs, and the collected videos are used to assess the changes that the reefs cause to the marine population. Manual annotation on the videos is used to train CV models which can then be employed for automatic classification. As the data is collected from unconstrained underwater environments, it varies in quality and visibility, which is also worsened by the use of frames extracted from videos, effectively decreasing the contextual information. In previous work [7], the dataset collected for the MOSAR project was used to test the combination of different embedding extractors and clustering algorithms in a zero-shot setting, with the aim of defining a classification pipeline for underwater species, by leveraging pre-trained models. The results showed that, for this data, the embedding extractor that performed best across all experiments was DINOv2 [8], and especially in combination with Agglomerative Clustering. More details in the related works section 2. However, as the clustering performance was far from the baseline performance, we decided to take a step back and assess the role of the embeddings themselves in the pipeline. By decoupling the embeddings from the clustering, we can define whether the discrepancy in performance is due to the difficulty of clustering or of representing images in a vectorised manner. Solving this doubt could allow future work to focus on techniques to improve clustering strategies or on different methods for embedding extraction. Clustering could then play a crucial role in the classification of new species, as it would allow for the detection of similar images, which could then be labelled manually, allowing the annotator to inspect a few images representative of the cluster. Considering the quality of the images that were used, it is also relevant to assess the capability of the embedding extractor in terms of representing the images in such a way that they can be used for classification. Therefore, this project aims at training a classifier on the embeddings obtained from DINOv2, and assessing how good the performance is in terms of identified classes. The results will be compared to an image classifier, trained using transfer learning to leverage pre-trained capabilities, and fine-tuned on the dataset. Using embeddings instead of full images for the classifier would allow us to decrease training time and computational costs by leveraging foundation models that are already available. Since underwater classification requires continuous re-training due to the changing environments and the appearance of new species, reducing computational costs is important. While foundation models have a great impact on global emissions [8], they can be leveraged to decrease task-specific re-training [9].

The research question for this paper is:

Can DINOv2 embeddings provide a reliable representation for marine species classification under challenging underwater conditions such that they can accurately be used for downstream tasks and dataset improvements?

The following paper is structured as follows to answer this question: The related work section 2 will provide an overview of both common underwater datasets and papers that have used similar pipelines in different contexts. The dataset will be presented in section 3,followed by exploratory data analysis, and the implementation details in section 4. Section 5 will provide a detailed overview of the experiments, and the analysis of the results, while section 6 will discuss the relevance of this method and results. The research question will be answered in section 7, and any final remarks will also be highlighted.

2 Related Work

2.1 Underwater overview and datasets

Underwater detection as a field has been analysed extensively. Particularly, in 2024 Jian et al. [4] conducted a comprehensive review of underwater object detection and the related datasets, to provide a survey of the current research on the topic. They discuss both traditional feature-engineering approaches and deep-learning-based methods, and they summarise key datasets used for underwater target detection. They then highlight both challenges and future prospects of the field. Challenges consisting, for example, of the need of emphasising diverse characteristics exhibited by underwater objects, the need to incorporate different perspectives in underwater scenes, the need to improve the detection of small objects, and the lack of robustness in the models for this task. They also highlight the need for better technologies to address issues such as high turbidity and refraction. Including background information is also a relevant challenge to be tackled, together with increasing the data availability.

Another relevant overview of underwater datasets is provided by Ayyagari et al. [10], who conducted a systematic analysis of the feasibility of leveraging fish detectors trained on different datasets, to detect fish (and other marine species) in new underwater marine datasets. They also compare the accuracy and training time of pre-trained models with ones without pre-training. They conclude that models pre-trained with OzFish [11] yield faster convergence and comparable performance with smaller training datasets, while they highlight that the performance is very dependent on the pre-training dataset, and its size. They highlight the importance of testing on diverse marine datasets and rendering these publicly available. They additionally state that it is important to decouple detection from classification to address the unique challenges of both tasks. While detection can use large available datasets for learning features, classification can benefit from unsupervised techniques such as clustering, which only becomes feasible after detecting the objects. This statement is very relevant for this project, as we are currently working on the assumption that a future model developed from this pipeline would receive data for which the species have been detected, and would only have to handle the classification.

2.1.1 Underwater datasets While we are working with proprietary data, multiple datasets have been made publicly available for training and testing.

Among the overviews of datasets for underwater marine species detection and classification, Fish4Knowledge [12] has been highlighted as a relevant benchmark. Specifically, it is a project funded by the European Union which aims at allowing more automatic data analysis of underwater species. They used live video feeds from 10 underwater cameras, amounting to 90 thousand hours of recordings, to investigate applicable methods for capture, storage, analysis, and querying of multiple video streams. They collected 27370 images of fish, divided into 23 clusters with related representative species.

DeepFish [13] is a dataset consisting of 40.000 (circa) underwater images from 20 habitats of tropical Australia. Saleh et al. [13] collected both point-level and segmentation labels to learn to automatically monitor fish count, identify location and define size. They provide an in-depth analysis of the dataset characteristics and a performance evaluation of state-of-the-art approaches on their benchmark.

Another dataset, collected in Danish waters, is the Brackish dataset [14]. It is composed of annotated image sequences of fish, crabs, starfish, Jellyfish, and shrimp captured in the Limfjorden in northern Denmark. The water, light, and visibility conditions are very similar to the ones of this project. The dataset was the first publicly available European underwater image dataset with bounding boxes annotations for different local species.

An additional dataset is the FishNet [15] one, developed by Khan et al., to automatically monitor aquatic species in order to assess the situation and take the relevant actions for marine preservation. They designed a dataset aimed at addressing the need for systems that can recognize, locate, and predict species and functional traits.

None of the presented datasets are similar enough to the one used for this project, either due to the marine species presented, the annotation type, or the format of the data. While it would be unreasonable to expect a dataset to match the one we are using, it also makes it difficult to compare the performance to available benchmarks. Due to the scope, time, and size of the datasets, it was not possible to test the current pipeline on the mentioned datasets. However, future iterations of the project could aim at testing whether a similar dataset to the one we are using exists, and how comparable the pipeline performance is between the two.

2.2 Embeddings-based classification

2.2.1 Embedding classification in natural language processing Embeddings have long been used in natural language processing (NLP) to encode words into numerical vectors which should represent both the word and its contextual information, effectively generating different embeddings for the same words in different contexts [16]. The embeddings have been used for text classification to assess the performance difference between classical and contextualised embeddings [16] by Wang et al, for example. Additionally, Da Costa et al. [17] provide a comprehensive survey of classification methods built on different types of embeddings. These are two of numerous examples of how the task is commonly analysed in NLP.

2.2.2 Image embedding classification Embeddings are also widely used in image-classification tasks. For example, Tian et al. [18] show that a strong embedding model can be more effective for classification than a sophisticated meta-learning model. Specifically, they train on an image classification task on the meta-learning training data to learn an embedding model and subsequently use the embeddings extracted for a simple linear classifier. They achieved

a better performance than the state-of-the-art (SOTA) few-shot-learning method they had available.

In their paper, Li et al. [19] analyse the combination of seven foundation-model embedding extractors with classical machine-learning classifiers for multi-class radiography classification. They concluded that MedImageInsight embeddings paired with support vector machines (SVM) or multi-layer perceptrons (MLP) adapters yielded the best results in terms of mean area under the curve (AUC). While their domain differs from ours, their findings support the effectiveness of using high-quality embeddings with simple classifiers.

For the FungiCLEF 2024 challenge, Chiu et al. [20] present an approach to the task of fine-grained classification of poisonous fungi identification, which uses a SOTA self-supervised vision model to extract features that are then used for the classification downstream task without the need for fine-tuning on the vision backbone. They incorporate metadata as additional input for the model, and train only on metadata and image embeddings. Gustineli et al. [21] solved the PlantCLEF 2024 competition by leveraging both base and fine-tuned DINOv2 models to extract generalised feature embeddings. They then trained classifiers to predict multiple plant species within one image. They train a linear classifier with the embeddings from the cropped, single-label images, and perform inference with both full images and grid-based ones. These papers offer a strong case for using our pipeline to solve the task. They offer suggestions for which simple classifiers to use and provide a baseline of performance that can be considered for our project.

2.2.3 Image embedding classification in underwater domains In the domain of Plankton Recognition, Kareinen et al. [22] study large-scale self-supervised pre-training for fine-grained plankton recognition. They use masked autoencoding (MAE) and a large dataset of plankton images to pre-train a general-purpose plankton image encoder. They show that pre-training with plankton data increases accuracy compared to standard ImageNet pre-training when the amount of training data is limited.

Still in the plankton domain, Ciranni et al. [23] conducted a comparison between in-domain and out-of-domain supervised and self-supervised pre-training in terms of the quality of embeddings generated for downstream image classification of plankton. They used both ResNet50 pre-trained on a large-scale plankton dataset (in-domain self-supervised pre-training) and well-established self-supervised methods such as DINO, MoCO, SimCLR, and DeepCluster. They evaluate the embeddings extracted with four different classifiers, namely, Logistic Regression, SVM, Random Forest, and shallow MLP. They achieve between 85% and 94% mean accuracy on their target datasets. They conclude that self-supervised pre-training provides useful features which can be used for plankton classification in the framework of feature extraction with transfer learning.

This paper is the closest we could find that applies the process we have chosen. The domain is very similar to fish species recognition, while plankton images, from visual inspection of example data, have very different characteristics. They appear to have very well-defined image instances that lack water influence in terms of turbidity, debris, and other issues related to movement, given by recording videos in unconstrained underwater environments.

While Kareinen et al. suggest that pre-training with domain images improves performance, Ciranni et al. argue for using self-supervised models. Taking inspiration from both papers, we will compare the performance of an in-domain pre-trained model and an out-of-domain self-supervised one, and reach our own conclusions.

2.3 Finding Ariel

This paper leverages the findings of "Finding Ariel: Comparing Embedding Extractors for Zero-Shot Clustering and Classification of Fish in Danish Waters" [7] (mentioned as Finding Ariel for brevity), and additionally analyses the performance capabilities on the available dataset. However, as this work is independent from that study, we briefly outline its methodology and results. The goal of Finding Ariel was to evaluate the combination of different embedding extractors: one designed for the task (DINOv2), and three embedding extraction backbones from pre-trained classifiers (ResNet50-ImageNet, MegaDescriptor, and Resnet50-FishNet). These three were pre-trained with different amounts of data from different domains, furthest to closest to underwater environments (ImageNet, WildLife Dataset, FishNet). The embeddings from the four extractors were then clustered with 4 clustering algorithms techniques, to compare performance. Having identified DINOv2 and Agglomerative clustering as the best combination, additional experiments were conducted, aimed at improving performance through preprocessing. These steps included merging classes to obtain a more balanced dataset, gray scaling the images, and applying contrast enhancement. The final results, after using a k-nearest neighbour classifier trained on the detected clusters, showed that gray scaling had a positive impact on the performance, while merging classes and enhancing contrast did not offer particular improvements. After careful consideration, we identified issues with the way the data was split in Finding Ariel, and resolved the issue for this project. This change, however, makes the comparison between the two papers impossible.

Given the above overview of the related works and the mentioned considerations, we will conduct the experiments to assess how well the presented pipeline works for cropped instances of marine species collected from unconstrained underwater environments in Danish waters.

3 MOSAR Dataset

3.1 Image data

The dataset used in this project was collected by Anemo Robotics[24] to test their underwater cameras for the MOSAR project [6].

The cameras are set in the harbour of Hundested in Zealand, Denmark, to assess the impact of artificial reefs on the presence of local marine species. There are two cameras pointed towards artificial reefs and one as a control, aimed at the rocky wall of the harbour. From the videos captured over the span of three months, July to October 2024, individual salient frames were manually selected by the company, and shared for this project. The total number of images is 5992 from 5 different camera views. As the frames were manually labelled around the marine subjects by a marine biologist, it was possible to crop around the bounding boxes to obtain a dataset

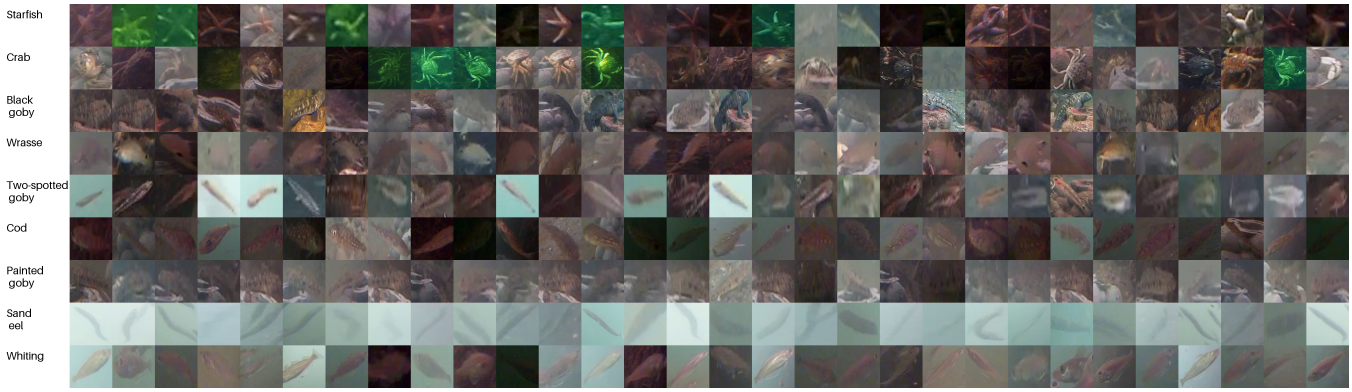


Figure 1: Mosaic representation of 30 randomly selected instances for every class.

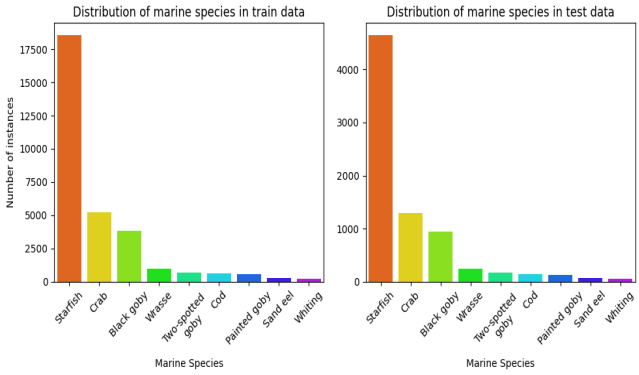


Figure 2: Marine species distribution for train and test data

of single subjects. This is relevant for this project as we are interested in seeing how well the models can learn to recognize specific species for classification, having already conducted some detection task, as suggested in section 2. The overall dataset comprises 38,572 cropped subjects. The cropped images were split into train and test (80/20 - 30,877/7695) in a stratified manner to preserve distribution, and the validation set was additionally extracted from the test set by dividing it in half, maintaining the distribution of species. Figure 2 shows the data distribution for training and test data.

Figure 1 shows a mosaic of randomly selected images from each class. Specifically, each row is a different class, and contains 30 randomly selected instances from that class. It is clear to see that some classes have a wide variety of image types. For example, both the Starfish class and the Crab class have green instances, blurry ones and easy-to-recognise ones. On the other hand, the Sand Eel class appears to have very homogeneous images, in the same colour spectrum and view of the subjects. Similarly, the Whiting row appears to have numerous images with fish in very similar positions, some of which very similar to the Sand Eel. Painted Goby and Black Goby appear to have a few instances that closely resemble one another. However, after a brief research, it became clear that the two species can be easily confused due to the similarity in scale patterns, which will probably be an additional challenge with the dataset. Overall, when analysing the results of the classifiers, these peculiarities of the data will be taken into consideration.

3.2 Preprocessing

The image data was pre-processed by first extracting the embeddings with DINOv2 and then applying column-wise normalisation.

3.2.1 DINOv2 - Embedding extractor DINOv2 is a foundation model developed by Meta AI research [8]. It is a self-supervised method that is aimed at learning general-purpose visual features from a large curated dataset. The curated dataset is collected from multiple sources, amounting to 1.2B images, which were then decreased with different methods (such as clustering) to 142M images. The result of the work is a set of pre-trained visual models trained on the data with different vision transformers. We used the small model with 22M parameters to extract embeddings from the cropped images. The resulting embedding vector is of size 384. This model was chosen based on the best results from Fogh et al. [7].

3.2.2 Normalisation The embeddings vectors were then normalised with the Scikit-learn Normalize function [25], as per standard practices on high-dimensional representation. This column-wise normalization applies L2 normalization to each feature independently, scaling the values within each of the 384 dimensions so that the L2 norm of the entire column is 1. This process ensures that no single dimension dominates the others, standardizing the contribution of each feature across all embedding vectors.

3.3 Embedding-Space Distribution Tools

Having normalised the embeddings, we decided to assess the shape of the data in two ways. The visual inspection was done with T-SNE, decreasing the dimensions to 2. However, as the data is highly dimensional and reducing it to 2 dimensions causes information loss, we decided to additionally conduct a median analysis. This consisted of calculating the median embedding vector for each class and the distance of each data point to the medians. This process will allow us to determine whether the data has a tendency to be close together or whether we cannot see any such patterns.

3.3.1 T-distributed stochastic neighbour embedding (T-SNE) T-SNE is a common visualisation tool for high-dimensional data, which gives each data point a location in a 2 or 3 dimensional map [26]. In practice, it converts the similarities between data points to probabilities and creates a single map that reveals structures at different

scales. High-dimensional data that lies on several low-dimensional manifolds is well captured by this method. In order to determine the types of decision boundaries to consider for the classifiers, we used T-SNE to visually inspect the data. However, as the algorithm decreases to 2 dimensions for visualisation purposes, we used it as an initial analysis to detect any pattern before choosing the classifiers.

3.3.2 Euclidean distance We initially calculated the distance from the median vectors with Euclidean distance [27]. As the data is normalised to the feature vector, we decided to assess whether simple Euclidean distance could work for the data. It assumes that the features are uncorrelated and have equal variance in all directions, distributed in a hypersphere. This assumption is the reason why we decided to also calculate a different type of distance.

3.3.3 Mahalanobis distance We decided to use Mahalanobis distance (MD) [28] to calculate each embedding distance to the class medians. Unlike Euclidean distance, MD is robust to imbalanced data and non-spherical distributions because it measures distance in units of standard deviation. It thus assumes that the data is a hyper-ellipsoid shape, which might possibly be closest to the actual data shape. Its core strength is its ability to account for feature correlations by incorporating the inverse of the variance-covariance matrix of the data.

3.4 Exploratory Data Analysis

Before deciding which model to use for classification, we used T-SNE to plot the data and assess whether there are any discernible patterns. Figure 3 presents the data reduced to 2D. It is clear to see that Starfish occupies the majority of the plot, having a mostly cohesive shape, with a few instances spread around in the plot. Similarly, for crab, we can see that the majority of the data is in the same space, but there is a higher overlap with other data points. Sand Eel is the only other class that has its own conglomerate of data points, while the rest of the classes have less defined groups and tend to overlap with one another. Specifically, Black Goby and Painted Goby are often overlapping, and similarly, Wrasse and Whiting. Overall, there are some discernible patterns that can be seen in this visualisation, which should be kept in mind for both analysing the results and deciding which classifiers to use. However, the visualisations are reduced from 384 dimensions, which results in information loss. We thus won't completely rely on this visualisation and will still test different classifiers.

Given the results of the T-SNE visualisation, we decided to analyse the shape of the data in terms of the distance of each data point to the median embedding of each class. Ideally, we would want each embedding to be closest to its own class median, but we have seen from the images and the T-SNE visualisation that there are variations in the data that would warrant a deeper analysis.

As previously mentioned, we calculated the Euclidean distance and the Mahalanobis distance. While the first assumes the data to be a hyper-sphere, the second modifies the calculations and assumptions to account for a hyper-ellipsoid. Both results are presented in figure 4. Unfortunately, without having a metric to compare the two, it is impossible to say which one is more relevant for this use-case. The euclidean distance analysis in figure 4a, shows that

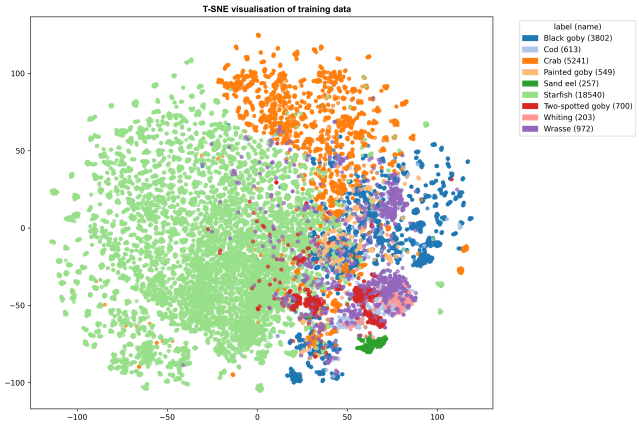
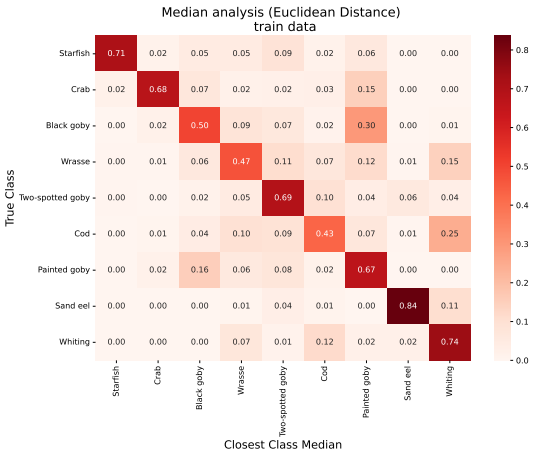


Figure 3: Visualisation of training data after embedding extraction and normalisation using T-SNE. In the legend, the number of instances of each class.

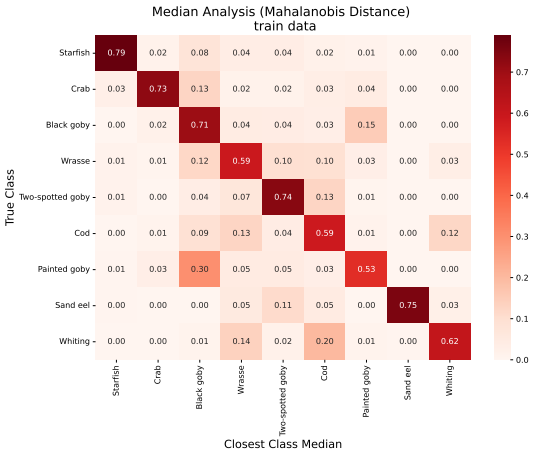
Black Goby, Wrasse, and Cod are the classes whose embeddings are the furthest from their median. Wrasse appears to be closest to different classes almost equally, while Cod is closest to Whiting. On the other hand, Black Goby and Painted Goby have numerous instances that are closest to the other class' median. Overall, this analysis assumes that the data is a perfect hypersphere, which is unlikely even after normalisation. On the other hand, the Mahalanobis distance analysis in figure 4b, while it follows the same trend as the Euclidean distance one, appears to show a higher percentage of embeddings being closest to their own class median. Similarly to before, Painted Goby and Black Goby have instances closest to each other's medians, and Wrasse and Cod are again not consistently closest to their own class, with 41% of the data closest to other medians. Wrasse and Cod, and Cod and Whiting also have at least 10% of instances that are closest to the other class (similarly to the Euclidean distance analysis), which suggests that there might be a general overlap in the hyperspace classes. While we will use the Mahalanobis distance for future analysis of the data, we can consider that the trends presented by the two analyses are very similar. Some of the classes, especially Black Goby, and Painted Goby, Cod, and Whiting and Cod, and Wrasse, have overlapping instances which could make the classification rather difficult. This result is consistent with the T-SNE visualisation and with the considerations from the mosaic shown previously, which makes both analyses more reliable.

4 Implementation details

The proposed pipeline for the project is as follows: The cropped images were fed into a DINOv2 embedding extractor, which outputted a 384-length vector per image. The vectors were subsequently column-normalised, to make sure no feature would be more impacting than the others. The normalised embedding was used to train a set of classifiers with the aim of learning to distinguish the 9 different classes. The different models were chosen based on simplicity, related work, and ease of use. The results were compared with a ResNet50 classifier, pre-trained on ImageNet and



(a) Median analysis based on Euclidean Distance



(b) Median Analysis based on Mahalanobis distance

Figure 4: Heatmaps (row-normalised) of the distance between the class median and the embeddings. The true class is the class of each embedding, while the predicted class represents the class of the closest median.

fine-tuned on different numbers of layers. In the following paragraphs we are going to outline which models we chose and why, in addition to implementation details specific to the models. We are going to work with embeddings extracted from the cropped image data presented in section 3.

4.1 Classifiers

4.1.1 Support Vector Machines (SVM) The first set of classifiers used are the support vector machine (SVM) models from the Scikit learn library [29]. SVMs [30] look for the optimal hyperplane to separate the classes, and use kernel functions to extend to patterns that are not linearly separable by transforming the original data to map into a new space. Overall, SVMs are effective for high-dimensional spaces and are rather memory efficient, as they use only a subset of points to define the decision function. The model is available for multi-class classification, while it is originally defined for binary, and it treats the problem with a one-vs-one or one-vs-rest approach. The choice for this classifier was made in accordance with [19], and due to the simplicity of the model compared to the task. The class weights for the model were balanced to avoid ignoring the classes with fewer samples. Given the lack of specificity from the visualisation of the data, we deemed it best to test the performance of different kernels. Specifically, we tested the different kernels available for the Support Vector Classifier (SVC) function (linear, Radial Basis function (RBF), Sigmoid, Polynomial 2nd, 3rd and 4th degree), which uses a one-vs-one approach, compared to the faster LinearSVC, which is designed to handle larger numbers of samples (in the order of 10^4) and uses one-vs-rest approach [29]. Specifically, we chose RBF as it measures the similarity between points in their Euclidean distance, which should work well with the normalised data and usually works well with class boundaries that are highly non-linear. The Polynomial kernel of different degrees were chosen as they allow for different shapes of decision boundaries. The sigmoid kernel acts similarly to a neuron in a Neural Network and uses tanh as an activation function. Finally, while it is highly unlikely that the data is linearly separable, the linear kernel was chosen as it is faster and usually produces boundaries that are clearly interpretable. it also allowed to test the difference between the simple linear kernel and the LinearSVC, which uses the o-v-r approach [29]. The only hyper-parameter tuned was the regularisation parameter C, which is a trade-off of misclassification of training examples versus simplicity of the decision surface. All other hyper parameters were set to the default value.

4.1.2 Neural network classifier For the second classifier, we chose a feed forward neural network built with PyTorch. This model should catch the non-linear decision boundaries better than a linear model, while not being architecturally and computationally very complex. The class weights for the model were balanced to avoid ignoring the classes with fewer samples. The architecture is defined following the tutorial from PyTorch [31], and it is composed of three hidden layers with ReLu activation functions. While there are much more complex architectures, we deemed it interesting to assess the performance of an architecturally simpler model, and draw a conclusion on the necessity of increasing complexity after seeing if it would offer improvements. The hyperparameter tested was the hidden dimension value which was the same for each layer. We decided to test for different values of the hidden dimension, chosen arbitrarily among common values (256, 512, 1024, which are powers of 2) and randomly chosen ones (850, 950).

4.2 ResNet50 - Baseline comparison

To compare the performance of a classifier on embeddings, with a more complex model designed to classify images, we decided to use ResNet50 pre-trained on ImageNet as a benchmark [32]. As previously mentioned in the related work section (2), we are interested in comparing an in-domain pre-trained model with an out-of-domain self-supervised one. Ideally, the model should be able to leverage the information derived from the images themselves, in addition to the transfer learning capabilities of being pre-trained on a large dataset. The model chosen was the one from PyTorch, the default one pre-trained on ImageNet1k_v2, with 25M parameters. It has 50 layers, and it is trained on ImageNet 2012, with 1.28M

images for training, divided to 1000 classes, of which a few are unspecified fish or starfish classes. The 50 layers are divided into 5 blocks: Convolutional layer block (1st), Bottleneck block (2nd), Identity and residual connections block (3rd), pooling block (4th), and a final fully connected block (5th). It also has input and output blocks, but they do not count when we describe which blocks we are freezing. We froze different numbers of layer blocks and fine-tuned the remaining. Specifically, we froze up to the second later block, up to the third, and up to the fourth (leaving only the fully connected layer to be fine-tuned). The decision on which layer blocks to freeze or not was determined by knowing that up until layer2, the model learns low-level features [33], which are relevant independently of the domain. As a note, other CV models could have been used instead of ResNet50, for example YOLO, which is a state-of-the-art computer vision model [34]. However, we wanted to compare the performance on single-image croppings, instead of on the whole image in one pass. This decision was made with the idea of maintaining a strict division between detection and classification, as suggested by Ayyagari et al. [10]. While YOLO does both detection and classification in one pass [34] on the whole image, setting bounding boxes and relative labels, the ResNet50 model used was set up to work on the cropped data directly.

4.3 Macro F1 - Metric

To assess the performance of the models, we decided to use the Macro F1 score, which computes the unweighted mean of all the per-class F1 scores [35]. It was deemed more relevant to use an unweighted mean as it is important to have a classifier that is able to detect all classes equally, thus penalising results that systematically mis-classify the classes with fewer instances.

The relevant code is provided in the Github repository [36].

5 Experiments and Results

The following section will provide an overview of the experiments that were carried out by tuning hyperparameters, and the subsequent choice of one classifier for each set of experiments, to further analyse in terms of errors and discernible trends.

5.1 Hyper-parameter tuning experiments

For the baseline ResNet50, we decided to test different combinations. Specifically, we tested how freezing different layers blocks and fine-tuning the remaining could influence the results. In table 1,we can see the Macro F1 score and the test loss for the experiments. Specifically, we froze everything but the fully connected layer in the first experiment, everything up to, and including, the 2nd layer (fine-tuning layer blocks 3, 4 and fully connected) and everything up to, and including, the 3rd layer (fine-tuning layer blocks 4 and fully connected) in the other two experiments, and fine-tuned the remaining on the training dataset. The results are on the test dataset. Freezing up to layer 2 offered the best results, which were only minorly higher than freezing up to layer 3, while being much better than freezing up to the fully connected layer. We will use the model with up to 2 frozen layer blocks as a baseline comparison for the other experiments.

Table 1: Macro F1 score results of the ResNet50 model with different frozen layers and fine-tuned on the training dataset.

Frozen Layer Block	Macro F1	Test Loss
Up until Layer 2	**0.80**	0.18
Up until layer 3	0.79	0.21
Up until Layer 4	0.57	0.49

Table 2: F1 score result of SVM with different kernels

	Macro F1 score	Best C
Linear	0.76	100
Radial Basis Function (RBF)	0.52	1
Sigmoid	0.63	0.1
Polinomial 2nd degree	**0.83**	100
Polynomial 3rd degree	0.83	50
polynomial 4th degree	0.82	50
LinearSVC	0.77	20

Table 3: Macro F1 score and Accuracy for different values of hidden dimension of a Neural Network.

Hidden Dimension	Macro F1
256	0.77
512	0.79
850	**0.82**
950	0.78
1024	0.80

Table 2 shows the macro F1 scores after testing SVM with different kernels. Notably, the Linear kernel from SVC and the LinearSVC results are rather similar, while the difference could be attributed to the hyper-parameters and the multi-class approach. Both RBF and Sigmoid performed poorly for this dataset. The results using the Polynomial kernels are very similar to one another. We chose to use the 2nd degree one for the comparison with the other models. Similarly to the SVM, we decided to test different hyper-parameters for the Neural Network classifier. Specifically, the hidden dimension of the model at different arbitrary values. Table 3 shows the Macro F1 Score for each experiment. The best result was obtained with 850 as a hidden dimension value.

5.2 Comparison of model trends and errors

Having identified the models that perform best in terms of Macro F1 scores, we deemed it interesting to analyse the confusion matrices that each model generates, as the Macro F1 scores are very close to one another. By analysing the confusion matrices, we are able to assess where the models fail and if there are any particular trends to consider for future experiments.

The benchmark model is the ResNet50 fine-tuned on the data with up to 2 frozen layer blocks. Figure 5 presents the row-normalised confusion matrix of the chosen baseline model performance on the test set. The model performs almost perfectly for both Starfish and Crab, which are the classes with the majority of the images, and almost as well for Black Goby and Two-spotted Goby. Sand Eel is also classified correctly 91% of the times, with minor misclassifications.

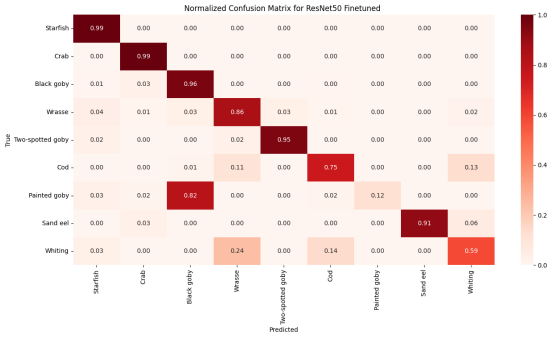


Figure 5: Confusion Matrix (row normalised) for the ResNet50 model fine-tuned on the data.

Wrasse and Cod have a lower correct percentage for classification, for which Cod specifically, has a few consistent misclassifications with other species. Whiting is confused with Wrasse and Cod more than with other classes, while Painted Goby is misclassified 82% of the time as Black Goby, making it the worst performing class for the model. The model performs overall rather well for most classes, and the confusion of Painted Goby with Black Goby could be attributed to the similarity of the species as mentioned in the data section 3 and the overlap of the data seen in the T-SNE visualisation in figure 3.

The SVC 2nd degree polynomial model represented by the confusion matrix in figure 6 performs rather well across the majority of the classes. It classifies almost always correctly Starfish and Crabs, which was to be expected due to the high prevalence of the two classes, but also Black Goby, Two-spotted Goby, and Whiting. For Sand Eel, the data is classified correctly 100% of the times, which could be attributed to the similarity of the images that was first seen in figure 1. Wrasse was misclassified with most other classes, while Cod had a misclassification prevalence with Whiting. Finally, Painted Goby is the class that is classified the worst, with double the percentage of instances misclassified as Black Goby, similarly to the ResNet50 performance. This could be attributed to the similarity in the species, and again, the overlap of the data seen in the T-SNE visualisation and in the median analysis.

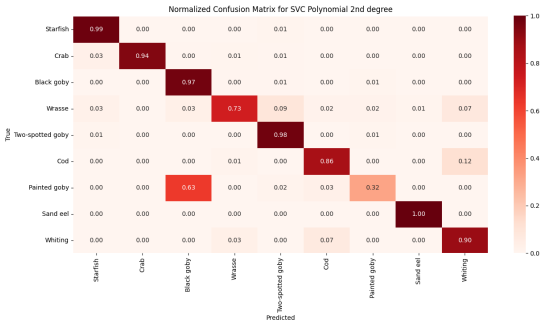


Figure 6: Confusion matrix (row normalised) for SVC with polynomial kernel 2nd degree

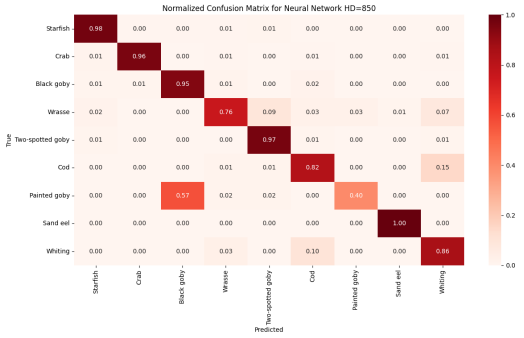


Figure 7: Confusion Matrix for Neural Network Classifier with hidden dimension=850

Figure 7 shows the confusion matrix for the neural network classifier we chose. Similarly to the previous model, the confusion matrix shows that the model performs well on most classes, classifying at least 95% of the instances correctly for 5 classes, and behaving poorly only for the Painted Goby class. Again, it misclassifies Painted Goby as Black Goby for 57% of the data, while it recognises it 40% of the times, offering an improvement on the SVC . Sand Eel is once again classified correctly 100% of the times. Overall, the classes are classified consistently slightly better than the SVC model.

After analysing the three confusion matrices, we can state that, while all three models consistently recognise the majority classes, and overall make almost the same misclassification errors, the best one in terms of correct classifications is the Neural Network. While the Macro F1 score was slightly lower than the one for the SVC model, the confusion matrix reports higher overall correct classifications and minor, but consistent, errors for some species. It is clear to see that the trends seen in the T-SNE visualisation and in the median analysis are reflected in the performance of the models. The errors were consistent and clear to attribute to way the embeddings are distributed in the hyperspace. Overall, the results of the Neural Network classifier are satisfying considering the data and the set-up of the project.

5.3 Median analysis on test data

As a final analysis, we decided to conduct the median analysis with Mahalanobis distance on the test data, and compare the true test data distribution with the results from the neural network classifiers' predictions. By comparing the two in the same context, we can see whether there are any improvements, and make considerations about the results of the classifier.

In figure 8, the heat-map represents the percentage of test embeddings closest to their own median compared to the other classes' median. We can see that, while most classes have a majority of embeddings closest to their own class, all of the classes have a dispersion of embeddings that are closest to other classes. Black Goby embeddings are often closer to Painted Goby, which is interesting considering that the previous confusion matrices show how usually Painted Goby is the one misclassified as Black Goby. This result could explain the misclassification. Cod has 12% of its instances

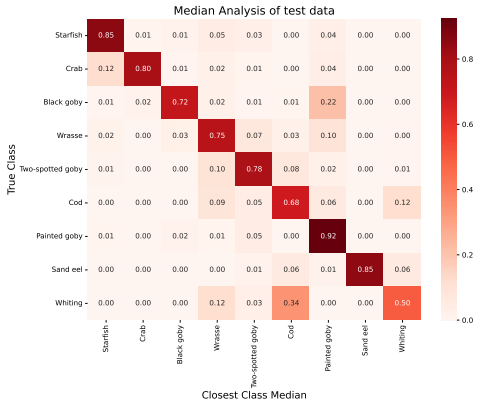


Figure 8: Median analysis of the test data with Mahalanobis distance .

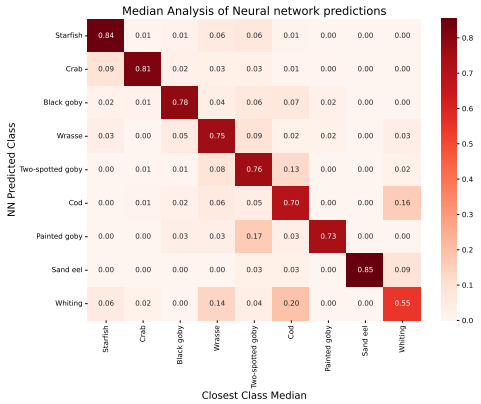


Figure 9: Median analysis of predictions from neural network classifier (with Mahalanobis distance).

closest to the Whiting class median, while Whiting has 34% of its instances closest to Cod. This is similar to the training data and is reflected in the misclassification errors seen above. Additionally, 12% of Crabs are closest to the Starfish's class median, which could be due to the green instances.

Overall, these results additionally explain some of the misclassification errors we saw from the confusion matrices of the different models.

A similar analysis was then carried out for the prediction of the neural network classifier. The results are showed in figure 9. The values on the diagonal are very similar to the values from the test data. There is still a scatter of embeddings that are closest to other classes' medians. However, some trends that were seen before, such as the Painted Goby- Black Goby consistent connection disappear, while the Painted Goby now has 17% of instances closest to the Two-Spotted Goby. On the other hand, Cod and Whiting maintain their relationship, and a few other interesting differences appear.

Overall, we can see that some of the patterns of the test data distribution are maintained by the neural network classifier, while others, especially the Black Goby - Painted Goby changed. However, the confusion matrix for the classifier shows a consistent misclassification of the two, which suggests that the shape of the data might have an influence on the classification. Additional analysis of these results would be part of future iterations of the project.

6 Discussion and Future Work

We have tested the performance of two classifiers trained on embeddings extracted from images of marine species, compared to a classifier trained on the images themselves. The analysis of the results shows that the macro F1 scores across the three classifiers are very similar. However, the qualitative differences in the errors suggest that the feed forward neural network provides the optimal performance profile for this scenario. The running time and other computational costs, while not specified for each model, demonstrated that the classifiers trained on embeddings are overall much faster than training the image classifier. Even when accounting for the DINOv2 embedding extraction time, the total time required for the embedding-based classifiers was approximately one-fifth of that required for fine-tuning the ResNet50. Leveraging the foundation model output allowed us to get informative representations, which worked well in this scenario and offered very high overall performance. Analysing the median distance allowed us to test whether the embeddings distribution was consistent for the data. The main errors were attributed to specific classes that were not easily distinguishable from one another. This is a key point to underline. We worked with data that were taxonomically divided in different ways depending on the species. The Starfish class was grouped into one big category, while the Gobies were divided into three fine-grained species, which are difficult to distinguish for non-experts (which also suggests that the expert annotators leveraged temporal information to label the species). The performance difference between Starfish and the different Goby classes can then be explained by this discrepancy in groupings. It could be that if we merged all the Goby classes into one, the overall misclassification would decrease. This solution could solve some issues, but would allow others to arise. Depending on the task at hand, we might need the fine-grained classes, or we might be mostly interested in assessing the general presence of species. In the context of this dataset, where the videos were recorded to assess the effect of the artificial reefs on the fish presence, we could benefit from merging some classes and having a broader overview of which species increase or decrease in numbers. As the data was directly provided by the company, it is relevant to assess whether the taxonomical incongruity is due to their specific needs or to a misinterpretation of the use of automated systems. Thus, the suggestions presented here do not include a targeted solution for the company, but a general consideration of the situation.

Having discussed decoupling detection from classification, we could include this pipeline in the broader context of having a simpler, off-the-shelf detection model that recognises the marine species from the background. This step could be done on videos, which would include more background and temporal information, which would add to the efficacy of the detection model. From the detected species, we could extract embeddings, which could then be used to classify the main species available. At this step, we could add a way to define if the images are too different from any of the

available classes, which would allow a human to step in and assess whether it is a new species to classify or an error. This could be done by assessing the distance of the embeddings to the available classes, or using clustering and outlier detection.

By using this Human-in-the-loop approach, we would be able to leverage the usefulness of foundation models and of classifiers, and relieve human annotators of a large section of the data labelling. The additional data could then be used for training new classifiers, which would be better and better at recognizing new species, at lower costs.

While it can be argued that the current trend is to automate as much as possible, using machine learning and artificial intelligence in general for solving as many tasks as we can think of, it is reasonable to discuss the necessity of maintaining a human-centric approach to solve this type of problem [37]. Decreasing human annotation time would free resources to be used for more complex tasks, but it is crucial to maintain control over what is being detected, and a critical approach to the use of the different tools. Using the pipeline presented above would help decrease both time and mislabelling, and would allow for more time to be dedicated to curating well-labelled, task-specific datasets, which could complement the well-known ones presented in the related work section. Having well-curated datasets that can be safely shared between organisations would also help solve challenge 9 [38] of the ocean decade challenges, which pushes towards the sharing of data and resources among all involved parties. While proprietary issues are relevant to consider, having a world-wide data bank of different environments could convince more organisations to share their data in exchange for facilitated access to different ones.

Future work could include testing different architectures for the feed forward neural network and other classifiers. We could also consider whether we should filter out the data based on the median analysis and only use the data closest to its own class median to train the classifiers. This process would provide cleaner training data and decrease the overlap between some classes which we saw in the results section. Unfortunately, it was out of scope for this project. Additionally, we could implement the whole pipeline, recording the videos, detecting species, extracting embeddings, classifying, introducing human feedback, and improving the performance. Introducing temporal information, and increasing knowledge from background analysis would bring additional information that could be leveraged for better results.

It is important to note that there is a problem with the data that was noticed too late in the process to be fixed in this project, but that will be fixed for future iterations. When splitting the data, we relied on image names to make sure that there would be no overlap or similarity between train and test data. However, at a later date it appeared that some images with different names have very similar images, which could cause data leakage between train and test data. Unfortunately, with no means to test this due to time and scope, we have to consider that the high results of the classifiers could also be attributed to this issue. The analysis of the results and the conclusions drawn from it, however, are still relevant in the context of the broader goal of the project.

7 Conclusion

In this paper, we set out to answer the question "Can DINOv2 embeddings provide a reliable representation for marine species classification under challenging underwater conditions such that they can accurately be used for downstream tasks and dataset improvements?". To answer it, we first extracted the embeddings with DINOv2, normalised them, and used them to train different classifiers to compare with a ResNet50 pre-trained on ImageNet. Specifically, we tested Support Vector Machines with different kernels, a feed forward neural network with different values for the hidden dimension and the number of layer blocks to freeze for the ResNet50. We chose one final model per type of classifier, and compared their performance on the test set in terms of correct/incorrect classifications. We chose the neural network with 850 hidden dimensions as the best-performing model overall. We additionally assessed the distribution/compactness of the embeddings in terms of how close they are to one another within each class, as a tool to argue for the possibility of misclassified instances.

We argued for the validity of our approach and defined how it could be useful in a broader context, both for individual stakeholders and in relation to a worldwide effort to protect the oceans. We conclude that the embeddings extracted with DINOv2 can offer a useful representation of marine species in unconstrained underwater environments, and that they can be crucial for improving the pipeline for data collection, labelling and usage in the context of marine monitoring.

References

[1] United Nations. Vision and mission, ocean decade - achieving what we want by 2030, 2021.

[2] United Nations. Challenge 7: Sustainably expand the global ocean observing system, 2024.

[3] United Nations. Challenge 2: Protect and restore ecosystems and biodiversity, 2024.

[4] Muwei Jian, Nan Yang, Chen Tao, Huixiang Zhi, and Hanjiang Luo. Underwater object detection and datasets: a survey. *Intelligent Marine Technology and Systems*, 2(1):9, 2024.

[5] Xiao Jiang, Haibin Yu, Yaxin Zhang, Mian Pan, Zhu Li, Jingbiao Liu, and Shuaishuai Lv. An underwater image enhancement method for a preprocessing framework based on generative adversarial network. *Sensors*, 23(13), 2023.

[6] Anemo Robotics. Anemo robotics - mosar dashboard, 2025.

[7] Altea Fogh, Yucheng Lu, Malte Pedersen, and Stefan Hein Bengtson. Finding ariel: Comparing embedding extractors for zero-shot clustering in danish fish classification. 2025.

[8] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2024.

[9] Anke Tang Yong Luo Han Hu Bo Du Yonggang Wen Dacheng Tao Hongling Zheng, Li Shen. Learning from models beyond fine-tuning. *Learning from models beyond fine-tuning*, 2025.

[10] Devi Ayyagari, Talukder Wasi Alavi, Navlika Singh, Joshua Barnes, Corey Morris, and Christopher Whidden. Dataset selection is critical for effective pre-training of fish detection models for underwater video. *ICES Journal of Marine Science*, 82(4):fsaf039, 2025.

[11] Australian Institute of Marine Science (AIMS). Ozfish dataset - machine learning dataset for baited remote underwater video stations, 2019.

[12] Robert Fisher, Yun-Heh Chen-Burger, Daniela Giordano, Lynda Hardman, and Fang-Pang Lin. *Fish4Knowledge: Collecting and Analyzing Massive Coral Reef Fish Video Data.* Intelligent Systems Reference Library. Springer, United Kingdom, 2016.

[13] Alzayat Saleh, Issam H Laradji, Dmitry A Konovalov, Michael Bradley, David Vazquez, and Marcus Sheaves. A realistic fish-habitat dataset to evaluate algorithms for underwater visual analysis. *Scientific reports*, 10(1):14671, 2020.

[14] Malte Pedersen, Joakim Bruslund Haurum, Rikke Gade, Thomas B. Moeslund, and Niels Madsen. Detection of marine animals in a new underwater dataset with varying visibility. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019.

[15] Faizan Farooq Khan, Xiang Li, Andrew J Temple, and Mohamed Elhoseiny. Fishnet: A large-scale dataset and benchmark for fish recognition, detection, and functional trait prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 20496–20506, 2023.

[16] Congcong Wang, Paul Nulty, and David Lillis. A comparative study on word embeddings in deep learning for text classification. In *Proceedings of the 4th International Conference on Natural Language Processing and Information Retrieval*, NLPIR '20, page 37–46, New York, NY, USA, 2021. Association for Computing Machinery.

[17] Liliane Soares Da Costa, Italo L Oliveira, and Renato Fileto. Text classification using embeddings: a survey. *Knowledge and Information Systems*, 65(7):2761–2803, 2023.

[18] Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B Tenenbaum, and Phillip Isola. Rethinking few-shot image classification: a good embedding is all you need? In *European conference on computer vision*, pages 266–282. Springer, 2020.

[19] Xue Li, Jameson Merkow, Noel CF Codella, Alberto Santamaria-Pang, Naiteek Sangani, Alexander Ersoy, Christopher Burt, John W Garrett, Richard J Bruce, Joshua D Warner, et al. From embeddings to accuracy: Comparing foundation models for radiographic classification. *arXiv preprint arXiv:2505.10823*, 2025.

[20] Christopher Chiu, Maximilian Heil, Teresa Kim, and Anthony Miyaguchi. Fine-grained classification for poisonous fungi identification with transfer learning. *arXiv preprint arXiv:2407.07492*, 2024.

[21] Murilo Gustineli, Anthony Miyaguchi, and Ian Stalter. Multi-label plant species classification with self-supervised vision transformers. *arXiv preprint arXiv:2407.06298*, 2024.

[22] Joona Kareinen, Tuomas Eerola, Kaisa Kraft, Lasse Lensu, Sanna Suikkanen, and Heikki Kälviäinen. Self-supervised pretraining for fine-grained plankton recognition. *arXiv preprint arXiv:2503.11341*, 2025.

[23] Massimiliano Ciranni, Ani Gjergji, Andrea Maracani, Vittorio Murino, and Vito Paolo Pastore. In-domain self-supervised learning for plankton image classification on a budget. In *Proceedings of the Winter Conference on Applications of Computer Vision*, pages 1588–1597, 2025.

[24] Anemo robotics. Anemo robotics, 2025.

[25] scikit-learn developers. Normalize, 2025.

[26] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[27] J.C. Gower. Properties of euclidean and non-euclidean distance matrices. *Linear Algebra and its Applications*, 67:81–97, 1985.

[28] R. De Maesschalck, D. Jouan-Rimbaud, and D.L. Massart. The mahalanobis distance. *Chemometrics and Intelligent Laboratory Systems*, 50(1):1–18, 2000.

[29] scikit-learn developers. Support vector machines, 2025.

[30] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[31] PyTorch. Build the neural network, 2025.

[32] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[33] Amit. Guide to freezing layers in pytorch: Best practices and practical examples, 2024.

[34] Ultralytics. Object detection.

[35] Kenneth Leung. Micro, macro weighted averages of f1 score, clearly explained, Jan 4, 2022.

[36] Altea Fogh. Research project, github repository, 2025.

[37] Bárbara C Benato, Cristian Grosu, Alexandre X Falcão, and Alexandru C Telea. Human-in-the-loop: Using classifier decision boundary maps to improve pseudo labels. *Computers & Graphics*, 124:104062, 2024.

[38] United Nations. Challenge 9: Skills, knowledge, technology and participatory decisionmaking for all, 2024.