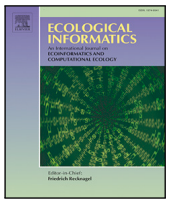
Ecological Informatics 90 (2025) 103272

Contents lists available at ScienceDirect

Ecological Informatics

journal homepage: www.elsevier.com/locate/ecolinf



Convolutional neural networks and vision transformers for Plankton Classification

Loris Nanni [a], Alessandra Lumini [b,*], Leonardo Barcellona [a], Stefano Ghidoni [a]

[a] Department of Information Engineering, University of Padova, Via Giovanni Gradenigo, 6b, Padova, 35131, Italy
[b] Department of Computer Science and Engineering, University of Bologna, via dell'Università 50, Cesena, 47522, Italy

ARTICLE INFO

Keywords:
Plankton
Neural networks
Ensemble
Transformers

ABSTRACT

In this paper, we present a study on plankton classification for automated underwater ecosystems monitoring. The study considers the creation of ensembles combining different Convolutional Neural Network (CNN) models and transformer architectures to understand whether different optimization algorithms can result in more robust and efficient classification across various plankton datasets. Tests involved different variants of the Adam optimizer and multiple learning rate variation strategies applied to several CNN architectures, building an ensemble of classifiers. Such ensembles were tested together with transformer-based models in a detailed comparative analysis considering feature extraction efficiency, computational cost, and robustness to species imbalances. The study highlights the performance of individual nets and ensembles on multiple plankton datasets, and discusses the potential for generalizing this approach to broader aquatic ecosystems. Experiments demonstrate that combining diverse neural network models in a heterogeneous ensemble significantly improves performance with respect to other state-of-the-art approaches across all the problems considered. Final results show that the ensemble-based approach achieves a remarkable accuracy improvement over individual CNN models and over standalone Vision Transformers.

1. Introduction

Oceans represent the vital life force of the Earth, contributing more than 70% of its oxygen and over 97% of its water supply. Their significance to sustaining life, including that of humans, cannot be understated. However, the escalating human population and corresponding resource consumption have significantly increased pressures on marine ecosystem services (Larkum et al., 2006). Consequently, it is imperative to monitor and preserve the oceanic ecosystem to safeguard marine habitats, including plankton populations and coral reefs, which play crucial roles in marine food cycles, habitat provision, and nutrient cycling.

Plankton, including both marine and freshwater microorganisms, form the foundation of the marine food chain. Their importance extends beyond simple trophic interactions, as they also play a critical role in the global carbon cycle and in regulating the Earth's climate (Hays et al., 2005). The composition of plankton communities is highly sensitive to environmental perturbations, both natural and artificial. These microorganisms can exhibit physiological, morphological, and behavioral changes in response to environmental changes, making them potential biosensors for detecting deviations from a computed healthy baseline, indicative of potentially harmful environmental changes.

Accurate taxonomic classification of plankton is therefore essential for biodiversity monitoring, assessing food web dynamics, and detecting ecological shifts driven by climate change. Plankton mediate key biogeochemical cycles: for example, sinking of plankton detritus fuels the biological carbon pump. Because they have short lifecycles and broad spatial coverage, plankton communities respond quickly to environmental change: shifts in plankton diversity and size-structure are early indicators of ecosystem health, plankton abundance and composition govern the productivity and resilience of aquatic food webs and the global carbon cycle. Taxonomically resolved plankton data are essential for assessing biodiversity and ecosystem status. Species- or genus-level counts of plankton support global biodiversity indicators, plankton communities are strongly affected by climate drivers, and in turn influence climate feedbacks (Bachimanchi et al., 2023; Ratnarajah et al., 2023). Warming, acidification, stratification, and changing circulation can reshape plankton assemblages (e.g. favoring smaller, warm-water species). Tracking these changes requires taxonomic identification: different species (even within one group) can have very different temperature tolerances or roles in carbon sequestration. Species-level data allow detection of climate-driven shifts. Different

* Corresponding author.
E-mail address: alessandra.lumini@unibo.it (A. Lumini).

https://doi.org/10.1016/j.ecoinf.2025.103272
Received 21 October 2024; Received in revised form 7 June 2025; Accepted 8 June 2025
Available online 25 June 2025

plankton taxa have different roles in the carbon cycle (e.g. calcifiers vs. non-calcifiers, silicifi ers, nitrogen fixers). Loss of plankton diversity under warming suggests reduced ecological stability. This matters for climate adaptation: fisheries, water quality, and carbon uptake all depend on plankton communities. Monitoring plankton taxonomy thus informs management of ecosystem services under climate change (Henson et al., 2021).

The limitations of traditional methodologies for measuring plankton diversity and abundance have stimulated the development of some automated plankton monitoring tools, some of which have recently been applied in freshwater systems.

Recent years have seen a surge in digital imagery for monitoring underwater ecosystems (Ciranni et al., 2024a). Advances in technology have facilitated the acquisition of high-resolution images, allowing for more detailed studies of plankton morphology and behavior (Olson and Sosik, 2007). However, such scenarios are very static and few interesting events occur over hours or days of observation, which cause a pressing need for automated detection and classification systems. Various researchers have explored automated methods, leveraging computer vision and machine learning techniques, to accurately annotate marine imagery. While AI-based classification primarily focuses on taxonomic levels, its application in functional group classification remains a challenge. The functional classification of phytoplankton, which is essential for understanding ecological interactions, requires integrating additional environmental parameters such as nutrient concentrations and fluorescence data. The integration of AI for large-scale marine ecosystem monitoring has also extended beyond image analysis to include the quantification of phytoplankton functional types (PFTs) from spatiotemporal datasets. For instance, Zhang et al. (2024) developed the first AI-driven, gap-free global PFT dataset, which provides daily global coverage with a resolution of 4 km over a 25-year period. This product significantly enhances our understanding of phytoplankton dynamics and their role in global biogeochemical cycles, further demonstrating the potential of AI technologies in marine monitoring and management. Machine learning (ML) and deep learning (DL) methods have been applied to classify microalgae of different species (Chong et al., 2024a) and predict the concentration of C-phycocyanin (CPC) in Spirulina platensis (Chong et al., 2024b).

Designing machine learning solutions for plankton image analysis faces challenges due to the difficulty in obtaining accurate annotations and the vast biological variability. Severe imbalance is common in datasets, with only a few samples representing numerous species. Additionally, the intrinsic variability at the species level poses methodological hurdles. Researchers are increasingly aiding marine experts in analyzing diverse image corpora with modern computer vision techniques, aiming to combine massive data availability with precision and speed.

Deep learning, particularly Convolutional Neural Networks (CNNs) few years ago and Transformers nowadays, has emerged as a predominant approach for underwater imagery analysis (Ciranni et al., 2024a), bringing about a revolutionary increase in performance. This paradigm shift has supplanted traditional techniques that relied on handcrafted descriptors and classification algorithms like Support Vector Machines (Zhao et al., 2010). The transition to deep learning has allowed for the automation of complex feature extraction processes, resulting in significantly higher accuracy and robustness in plankton image classification. According to Kyathanahally et al. (2021), the adoption of deep learning techniques has led to breakthroughs in identifying subtle morphological differences among plankton species, which were previously challenging to discern using conventional methods. The ability of deep learning models to handle large and diverse datasets has further cemented their role as indispensable tools in modern marine biology.

The groundbreaking application of deep learning for plankton image classification was first demonstrated in a competition featuring the Kaggle-121 dataset,[1] encompassing 30,000 samples across 121 species. The winning team proposed an ensemble of CNNs trained on augmented segments of the dataset. This competition underscored the efficacy of ensemble methods in enhancing accuracy, alongside the substantial time investment required for training deep neural networks.

Despite the many challenges, subsequent works introduced various CNN architectures and ensemble techniques, illustrating the potential of deep learning in plankton image analysis. Py et al. (2016) explored several CNN configurations, incorporating an inception layer capable of processing multi-size input images, and evaluated them on the Kaggle-121 dataset. Some methods were specifically devised to address the significant species imbalance prevalent in plankton datasets. For instance, Lee et al. (2016) proposed a class-normalized pre-training approach, while Wang et al. (2017) attempted to augment underrepresented species by generating new samples using a CGAN-like architectures. Recent studies (Khaldi and Khaldi, 2024) have demonstrated the effectiveness of deep learning techniques for classifying microscopic images of marine organisms, including harmful phytoplankton, using CNN models like ResNet, ResNeXt, DenseNet, and EfficientNet. Their study utilized transfer learning techniques, achieving impressive classification accuracy, but also highlighted the challenge of differentiating morphologically similar species. This further underscores the need for accurate and scalable solutions in marine ecosystem monitoring, particularly when dealing with organisms that pose a threat to aquatic environments. To address the challenges of model complexity and computational cost in plankton classification, Yuan et al. (2024) proposed an improved MobileNetV2 architecture, achieving a high accuracy of 95.46% across 12 plankton types. While lightweight models effectively mitigate computational cost and model complexity, allowing for fast and efficient in situ recognition, they may not always be necessary when dealing with larger datasets. For instance, with datasets containing 40,000 images, a more complex ensemble approach can be employed, as classification (i.e. inference, not the training) can still be completed within minutes using a sufficiently powerful GPU. The adoption of CNN ensembles has contributed to enhanced classification performance, as demonstrated by Lumini and Nanni (2019), whose two-stage ensemble comprised diverse backbone CNN architectures trained on different images. Similarly, Kyathanahally et al. (2021) introduced the LakeZoo Plankton dataset and presented an ensemble of six CNNs achieving impressive classification performance across both new and existing datasets. Maracani et al. (2023) explored in-domain and out-of-domain transfer learning, emphasizing the crucial role of the number of plankton species during in-domain pre-training in optimizing subsequent classification tasks. See Ciranni et al. (2024a), Eerola et al. (2024) for a comprehensive overview of recent advancements in plankton image analysis.

Recent advancements in AI-driven plankton classification have explored real-time and edge computing applications for marine ecosystem monitoring. One notable example is RAPID (Real-time Automated Plankton Identification Dashboard), an AI-based system deployed on the Plankton Imager, a high-speed line-scan camera designed for onboard zooplankton classification (Pitois et al., 2025). RAPID utilizes a ResNet-based classifier to categorize plankton into three broad groups: Copepods, Non-Copepod Zooplankton, and Detritus, processing data in real time during oceanographic surveys.

While RAPID demonstrates the feasibility of real-time classification at sea, it also highlights computational constraints that limit taxonomic resolution and classification granularity. These challenges reinforce the need for scalable deep learning frameworks capable of handling more detailed classifications across diverse plankton taxa.

Our study builds upon these advancements by systematically evaluating ensemble deep learning approaches for plankton classification. Unlike RAPID, which focuses on real-time classification with limited

[1] https://www.kaggle.com/competitions/datasciencebowl/overview

taxonomic depth, our approach leverages CNN and Transformer-based ensembles to achieve higher classification granularity across multiple plankton datasets. Additionally, we investigate the role of advanced optimization techniques, such as Adam variants, in improving classification performance and model robustness.

Despite recent advancements, several key challenges remain in automated plankton classification. CNN-based models are highly effective in extracting local spatial patterns, while Transformers provide a more global contextual understanding. However, neither approach alone fully addresses issues related to dataset variability, species imbalance, and model generalization. To overcome these limitations, our work systematically evaluates ensemble architectures combining CNNs and Transformers, assessing their effectiveness across multiple datasets.

In this study, we investigate the effectiveness of ensembles comprising diverse neural network models, considering both CNNs and transformers, fine-tuned on multiple datasets. Our analysis, conducted across seven well-established datasets (Zheng et al. (2017), Kyathanahally et al. (2021), Ciranni et al. (2024b), Batrakhanov et al. (2024) and Kraft et al. (2022)), demonstrates significant performance enhancements compared to existing state-of-the-art methods. Unlike previous studies that focused on either CNNs or Transformers individually, our work systematically evaluates the benefits of combining these architectures into a single ensemble framework. Additionally, we analyze the impact of different optimization strategies, including Adam variants, in improving classification robustness. Despite the computational complexity, our proposed system offers the advantage of plug-and-play functionality across diverse problem domains, requiring minimal parameter tuning and dataset-specific optimization. The paper is structured to present the CNN architectures employed, experimental setups, and findings, concluding with directions for future research. Additionally, a dedicated discussion contextualizes our results with recent literature in ecological informatics, examining prior applications of AI-driven classification in marine and freshwater monitoring.

The takehome messages of this work are the following:

- We use a well-specified testing protocol, using public datasets, so it could be a baseline for future works on this topic, allowing fair comparison between different methods;
- We show that combining multiple learning strategies and Adam variants is a feasible way to create ensembles of CNNs, offering improved performance;
- Combining different topologies of CNNs offers a higher improvement than combining different topologies of transformers: in general, ensembles of transformers have a smaller boost with respect to a stand-alone network when compared with ensembles of CNNs.
- Rejection-based classification improves efficiency: By discarding 20% of the most uncertain patterns based on the difference between the top two classification scores, we can classify the remaining 80% using efficient models without any performance loss. This strategy significantly reduces computational cost while maintaining accuracy.
- The proposed ensemble achieves SOTA or comparable performance to SOTA in all tested datasets. We recognize that a key limitation of ensemble methods is their reliance on available significant computational power. While our proposed system performs well with GPU support, it is not ideal for edge computing. However, satellite connections for data uploads have become increasingly accessible. Many expeditions, for instance, utilize Starlink connections to transfer data to servers for analysis.

2. Methods

In this study, the application of deep learning techniques involves fine-tuning of established neural network architectures, all initialized with ImageNet pre-trained weights, on the target plankton datasets.

We evaluate multiple architectures taken from the most promising models proposed in the literature, with the dual objective of identifying the most appropriate models for classification tasks and leveraging their diversity to construct an ensemble.

2.1. Convolutional neural networks

Convolutional Neural Networks (CNNs) (Goodfellow et al., 2016) are deep neural networks specifically designed for computer vision and image processing tasks. They use convolutional filters to automatically extract features from images and pass them through various layers to identify patterns and characteristics. CNNs are effective for image classification, similarity-based clustering, and object recognition, using neurons organized in layers to perform these functions.

In this work, we consider three CNN architectures – ResNet50, EfficientNetD0, and MobileNetV2 – selected for their diverse design philosophies and proven efficacy in various computer vision tasks. Their selection was guided by three main criteria: (i) previous success in plankton classification or related tasks, (ii) computational efficiency for large-scale image analysis, and (iii) diversity in network topology to enhance ensemble performance. ResNet50 was chosen for its deep feature extraction capabilities, EfficientNetD0 for its balance between accuracy and model efficiency, and MobileNetV2 for its suitability in real-time, low-resource environments. This ensures a robust ensemble where each model contributes unique strengths.

ResNet50. (He et al., 2016) ResNet50, or Residual Network, is designed to address the vanishing gradient problem by introducing shortcut connections that allow gradients to flow directly through the network. This architecture comprises 50 layers, including convolutional, pooling, and fully connected layers. It is widely recognized for its depth and robustness in feature extraction, making it a suitable choice for complex image classification tasks.

EfficientNetD0. (Tan and Le, 2019) EfficientNetD0 is part of the EfficientNet family, which scales network dimensions – depth, width, and resolution – using a compound coefficient to achieve optimal performance while maintaining a reduced number of parameters and computational load. The smallest model in the family, EfficientNetD0, balances accuracy and efficiency, making it particularly suitable for large-scale image analysis and resource-constrained environments.

MobileNetv2. (Sandler et al., 2018) MobileNet is designed for efficient execution on mobile and embedded devices. It employs depthwise separable convolutions, significantly reducing the number of parameters and computations compared to traditional convolutional layers. MobileNet is known for its lightweight nature and fast inference times, making it suitable for real-time applications where computational resources are limited.

These networks are evaluated using various optimizers and learning rate strategies to identify the best combinations for our specific tasks. The selection of these architectures allows us to compare performance across different design paradigms, providing comprehensive insights into their relative strengths and suitability for diverse image classification scenarios. A detailed comparison of accuracy, interpretability, and computational efficiency is provided in the Results section, highlighting trade-offs among these architectures. For all the tests we use a batch size of 30 and training pattern shuffling in each epoch; for more details see the experimental section.

2.2. Vision transformers

Vision Transformers (ViTs) have emerged as a powerful alternative to Convolutional Neural Networks (CNNs) for various computer vision tasks. Unlike CNNs, which rely on convolutional operations to capture spatial hierarchies, ViTs leverage the self-attention mechanism to model long-range dependencies within an image.

In this work, we evaluate several Vision Transformer-based models, including ViT, DeiT, Swin Transformer, CoaT, and BEiT2, selected for their state-of-the-art performance, diverse architectural characteristics, and ability to model long-range dependencies in image classification tasks. Vision Transformers (ViTs) process entire image patches simultaneously, capturing global spatial relationships that complement the local feature extraction of CNNs, allowing the ensemble to leverage both fine-grained texture information and broader contextual cues. Each selected architecture contributes unique strengths to the ensemble: ViT serves as a foundational Transformer model for image recognition, DeiT introduces data-efficient training mechanisms, Swin Transformer incorporates hierarchical feature representations, CoaT merges convolutional and attentional mechanisms for improved hybrid modeling, and BEiT2 leverages masked image modeling for enhanced representation learning. This diverse selection enhances the ensemble's robustness, ensuring improved generalization across multiple plankton datasets.

ViT. (Vision Transformer) (Dosovitskiy et al., 2020) is one of the pioneering models that applies the transformer architecture to image classification by treating image patches as tokens. ViT demonstrated that transformers could outperform traditional CNNs when pre-trained on large-scale datasets.

DeiT. (Data-efficient Image Transformers) (Touvron et al., 2021) is designed to be trained with fewer data by leveraging a distillation token and knowledge distillation from a CNN teacher. This approach makes DeiT particularly efficient in terms of data requirements while maintaining high accuracy.

Swin. (Shifted Window Transformer) (Liu et al., 2021) introduces a hierarchical structure with shifted windows, enabling the model to capture both local and global features efficiently. This design not only improves computational efficiency but also enhances performance on various vision tasks.

CoaT. (CoAtNet) (Dai et al., 2021) combines convolutional and attentional layers to leverage the strengths of both paradigms. CoAtNet effectively merges the efficiency and inductive biases of convolution with the flexibility and scalability of attention mechanisms, making it suitable for a wide range of data sizes and achieving strong performance across various vision tasks.

BEiT2. (BEiT v2) (Peng et al., 2022) extends the concept of BERT from natural language processing to vision tasks, using a masked image modeling objective to pre-train the transformer. BEiT2 achieves significant improvements in various downstream tasks by learning rich visual representations.

For the transformer models, the Timm library (Wightman, 2019) was used. Each model was fine-tuned with four GPUs (Nvidia RTX 3090) with a batch size of 64. The optimizer used was AdamW with a weight decay of 0.03 and a learning rate following Nanni et al. (2023). Ten percent of the training data was used as a validation split, and early stopping was applied after seven consecutive epochs without improvement in the validation and the model with the best F1-score in validation was used for the test.

2.3. Adam variant optimization methods

Traditional stochastic optimization methods, such as Stochastic Gradient Descent (SGD), often struggle with issues such as vanishing gradients, slow convergence, and sensitivity to learning rate selection. Adam (Kingma and Ba, 2014) was introduced as an adaptive learning rate method that combines momentum and per-parameter scaling, improving stability and convergence speed in deep networks. However, standard Adam has been found to suffer from poor generalization in some cases, leading to the development of various Adam variants that address specific limitations.

In this study, we explore multiple Adam-based optimizers to enhance model robustness and training efficiency. The selected variants include AdamW, which decouples weight decay from the optimization process to prevent over-regularization; DGrad, which adjusts the update rule to improve resilience against sharp minima; Hyperbolic Adam (Hyp), which stabilizes gradient updates in highly non-convex loss surfaces; and MinD, which minimizes gradient noise for more stable updates. These methods were chosen based on their effectiveness in improving convergence rates and avoiding suboptimal solutions.

By incorporating these adaptive optimization techniques, we aim to improve the generalization ability of our models, ensuring that the ensemble-based classification remains robust across multiple datasets without excessive fine-tuning.

To gain a clearer understanding of the tested optimizers (Nanni et al., 2023), it is helpful to revisit the Adam optimization algorithm and DGrad (Nanni et al., 2022b). In this paper we do not make comparisons among different variants of Adam, but only test their effectiveness while creating ensembles; for such a comparison we refer the reader to Nanni et al. (2023).

Recall the Adam update rule:

$$m_t = \beta_1 \times m_{t-1} + (1-\beta_1) \times g_t \tag{1}$$

$$v_t = \beta_2 \times v_{t-1} + (1-\beta_2) \times g_t^2 \tag{2}$$

$$m_{\hat{t}} = \frac{m_t}{(1-\beta_1^t)} \tag{3}$$

$$v_{\hat{t}} = \frac{v_t}{(1-\beta_2^t)} \tag{4}$$

$$\theta_t = \theta_{t-1} - \frac{\alpha \times m_{\hat{t}}}{(\sqrt{v_{\hat{t}}} + \epsilon)} \tag{5}$$

where the constants used are: $\beta_1 = 0.9$; $\beta_2 = 0.999$; $\alpha = 0.001$, m_t, is the first moment of the gradients; v_t, is the second moment of the gradients; g_t, is the gradient at time step t; β_1 and β_2 are the decay rates; α is the learning rate; ϵ is a small constant added for numerical stability; $m_{\hat{t}}$ and $v_{\hat{t}}$ are bias-corrected estimates of the first and second moments, respectively; finally, θ_t represents the current set of parameters.

A simple variant of Adam that is widely used (mainly with transformers) is AdamW, which replaces (5) with (6); AdamW adjusts the weight decay term w to appear in the gradient update:

$$\theta_t = \theta_{t-1} - \frac{\alpha \times m_{\hat{t}}}{(\sqrt{v_{\hat{t}}} + \epsilon)} + w_{t-1}\theta_{t-1} \tag{6}$$

In the experimental section, we explain the reasons why different Adam variants find different minima, being useful for creating ensembles.

DGrad. DGrad (Nanni et al., 2022b) is a variant of DiffGrad (Dubey et al., 2019) which is based on the absolute difference between the current and the moving average of the element-wise squares of the gradients (av_t), see (7)–(9), for getting a method that is more resilient to variations in the gradient differences. The net parameters are updated using (10), where the weighting factor defined in (9) is applied:

$$\Delta ag_t = \left| g_t - av_t \right| \tag{7}$$

$$\Delta \widehat{ag}_t = \left(\frac{\Delta ag_t}{\max\left(\Delta ag_t\right)} \right) \tag{8}$$

$$\xi_t = \mathrm{Sig}(4 \cdot \Delta \widehat{ag} t) \tag{9}$$

$$\theta_t = \theta_{t-1} - \xi_t \frac{\alpha \times m_{\hat{t}}}{(\sqrt{v_{\hat{t}}} + \epsilon)} \tag{10}$$

Hyperbolic optimizer (Hyp). The weighting factor ξ_t is calculated as:

$$lr_t = \frac{-1}{(a\Delta ag_t + b)} + c \tag{11}$$

$$\xi_t = \frac{lr_t}{max(lr_t)} \tag{12}$$

and (12) is applied in (10) for parameters update.

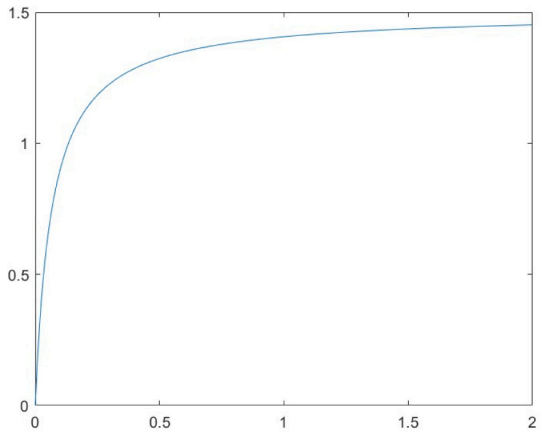


Fig. 1. Plot of lr_t for $a = 10$, $b = 2/3$, and $c = 3/2$. On y-axis, the lr_t value. On the x-axis, the Δag_t value.

With $a = 10$, $b = 2/3$, and $c = 3/2$ (the parameters used in our tests) we obtain the following Fig. 1. Using this function, the weighting factor ξ_t is (almost) close to 1 if Δag_t value is higher than 0.5.

MinD optimizer. MinD exploits DGrad and Exp (Nanni et al., 2022b) in different ways: ξ_{t1} is calculated using DGrad while ξ_{t2} is obtained through Exp. For details on Exp, refer to Nanni et al. (2022b). Then, ξ_t is obtained as:

$$\xi_t = min(\xi_{t1}, \xi_{t2}) \tag{13}$$

which is applied into (10) for parameters update.

Angular Injection optimizer (AIO). This approach is based on AngularGrad (Roy et al., 2021) and injection (Dubey et al., 2022). It produces a score to regulate the step size by utilizing the angular information of the gradient from previous iterations. The AIO considers the angle/direction of the gradient vector, not just its magnitude. To leverage the variations in gradients during optimization steps, an angular coefficient is introduced:

$$A_t = tan^{-1}\frac{(g_t - g_{t-1})}{(1 + g_t g_{t-1})} \tag{14}$$

$$A_{min} = min(A_t, A_{t-1}) \tag{15}$$

$$lr_t = \frac{-1}{(aA_{min} + b)} + c \tag{16}$$

where $a = 10$, $b = 2/3$, and $c = 3/2$. In this approach, ξ_t is obtained as follows:

$$\xi_t = \frac{lr_t}{max(lr_t)} \tag{17}$$

next, (17) is applied in (23).

To leverage curvature information during optimization, the curvature-guided (weighted) second-order momentum is incorporated into the first-order momentum:

$$g_s = g_t^2 \tag{18}$$

$$k = 2 \tag{19}$$

$$avg_t = \beta_1 \times avg_{t-1} + \frac{(1-\beta_1)(g_t - delta \times g_s)}{k} \tag{20}$$

$$delta = \theta_{t-2} - \theta_{t-1} \tag{21}$$

$$avgsq_t = \beta_2 \times avgsq_{t-1} + (1-\beta_2) \times g_s \tag{22}$$

$$step = \xi_t \times (\frac{avg_t}{\sqrt{avgsq_t} + \epsilon}) \tag{23}$$

$$\theta_t = \theta_{t-1} - step \cdot \frac{\alpha \times \sqrt{(1-\beta_2^t)}}{(1-\beta_1^t)} \tag{24}$$

where ξ_t in (23) is defined in (17); β_1 and β_2 are defined and initialized as in Adam.

This approach works in the following way:

- in the first training iteration, we use base Adam;
- in the second training iteration, first we set ξ_t=1 in (23) then we use avg_t and $avgsq_t$ calculated using gradients calculated by the standard Adam algorithm in the first training iteration (i.e., utilizing Eqs. (2) and (3));
- starting from the third training iteration the approach works as detailed above.

The choice of using base Adam in the first iteration establishes a stable foundation for gradient and moment estimation. Transitioning to the modified approach in the second iteration, with specific settings, ensures a smooth adjustment to the new algorithm, reducing initial instability. Full implementation of the variant from the third iteration capitalizes on stabilized estimates, thereby enhancing training efficiency and model performance.

3. Experiments

3.1. Datasets

As observed by Ciranni et al. (2024a), the literature on plankton classification boasts a wealth of available datasets, thanks to the continuous advancement of acquisition systems over the last three decades. Such improvements have enabled the development of more refined analysis techniques and the establishment of standardized benchmark datasets for assessing the performance of innovative methodologies. To validate our approaches, we performed experiments on seven plankton datasets.[2] In Table 1 a summary of the datasets used in this study is reported.

In our study, the term "class" is used strictly in the context of machine learning, referring to the output categories of the classification task. For clarity, we occasionally use "species" to denote biologically meaningful categories; however, we acknowledge that certain labels—such as detritus, dirt, or unknown—do not correspond to actual plankton species but rather represent non-taxonomic or ambiguous entities. This distinction is now made explicit throughout the manuscript to avoid confusion between taxonomic ranks and classification labels.

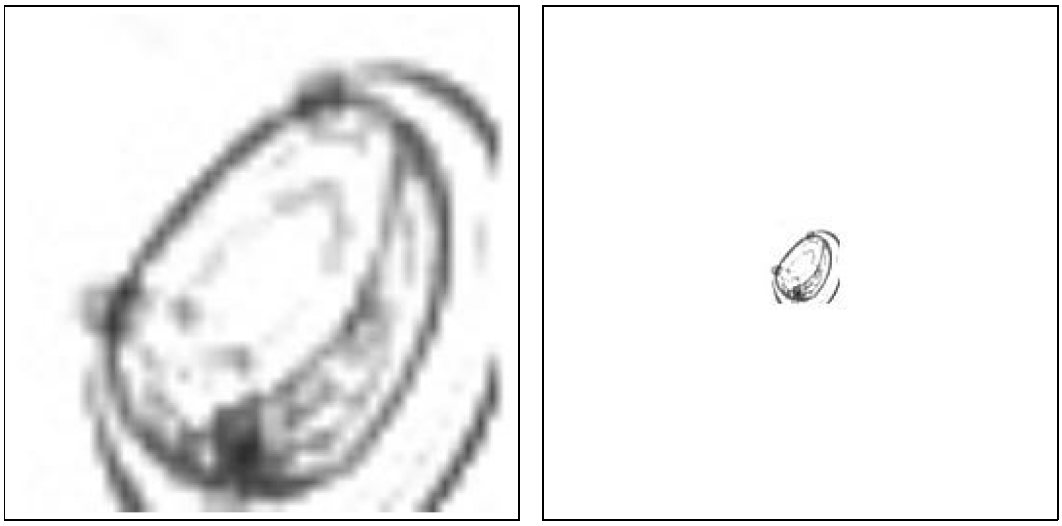
WHOI22 is a dataset containing 6600 grayscale images stored in TIFF format. These images, acquired by Imaging FlowCytobot from Woods Hole Harbor water, belong to 22 manually categorized plankton species, each with an equal number of samples. In our experiments, we adopt the same testing protocol proposed by Sosik and Olson (2007), which involves splitting the dataset equally between training and testing sets.

ZooScan20 is a small dataset consisting of 3771 grayscale images acquired using Zooscan technology from the Bay of Villefranche-sur-mer. Due to artifacts present in the images (resulting from manual segmentation), all images have been automatically cropped before classification. The dataset comprises 20 species with a varying number of samples for each class. We follow the testing protocol proposed by Zheng et al. (2017), which employs 2-fold cross-validation.

Kaggle-38 is a subset selected by Zheng et al. (2017) from a larger dataset acquired by ISIIS technology in the Straits of Florida and used for the National Data Science Bowl 2015 competition. This subset includes 14 374 grayscale images from 38 species. Although the species distribution is not uniform, each species contains at least 100 samples. We adopt the same testing protocol as proposed by Zheng et al. (2017), utilizing 5-fold cross-validation.

LakeZoo (Kyathanahally et al., 2021) comprises 17 943 RGB images of plankton, categorized into 35 species, with a highly variable number

[2] Three datasets are available from: http://ouc.ai/zhenghaiyong/research/mkl/Dataset.zip.



(a) Image pre-processed by sqr (b) Image pre-processed by pad

Fig. 2. Comparison of pre-processing methods.

of samples per class, ranging from a minimum of 10 to a maximum of 3321. These images were captured using the Dual Scripps Plankton Camera (DSPC) camera in Lake Greifensee, Switzerland, between 2018 and 2020, as part of a plankton community monitoring program. Samples were collected weekly and meticulously identified under a traditional microscope for accurate annotation. The dataset is divided into training (85%) and test (15%), from the training a validation of (15%) is extracted in Kyathanahally et al. (2021), sets,[3] ensuring a balanced species distribution across partitions. The DSPC camera captures images of plankton taxa at user-defined frequencies, with regions of interest (ROIs) containing plankton organisms extracted for subsequent feature extraction and classification.

WHOI15 (Ciranni et al., 2024b)[4] This dataset consists of phytoplankton images sourced from the WHOI large-scale collection, encompassing samples acquired over four years, from 2007 to 2010. This dataset includes 15 species: Asterionellopsis, Chaetoceros, Cylindrotheca, Dactyliosolen, Detritus, Dinobryon, Ditylum, Licmophora, Pennate, Phaeocystis, Pleurosigma, Pseudonitzschia, Rhizosolenia, Skeletonema, and Thalassiosira. For evaluation, a leave-one-out year testing protocol is applied, where images from three years are used for training, while those from the remaining year serve as the test set. This process is repeated for each year, resulting in four distinct training/test splits.

CrossD (Batrakhanov et al., 2024)[5] A total of 64,453 images were collected from 31 plankton species across two different domains. The species distribution is notably imbalanced, with the number of images per class-domain combination ranging from just 5 to 12,280, reflecting the natural rarity of each plankton species. CytoSense (CS) images were obtained from natural samples in the Baltic Sea during the summer of 2020 (July and August) at the Utö Atmospheric and Marine Research Station. To optimize the detection of different phytoplankton size species, two separate runs were conducted, each with distinct chlorophyll a fluorescence thresholds (30 mV and 80 mV) and sampling pump speeds (2 μ L/s and 5 μ L/s), targeting smaller and larger organisms, respectively. 12 831 images are extracted using CS. The Imaging FlowCytobot (IFCB) operated in continuous mode, with a new sample analyzed approximately every 23 min. A chlorophyll a trigger was used, and a 150 μ m mesh was placed at the instrument inlet to prevent clogging. 51 622 images are extracted using IFCB. The neural nets are trained using CS images and tested in the IFCB images.

Table 1
Description of the datasets used in this study.

Name	#Species	#Samples	#Colors	Protocol
WHOI22	22	6600	grayscale	50:50
ZooScan20	20	3771	grayscale	2CV
Kaggle-38	38	14 374	grayscale	5CV
LakeZoo	35	17 943	RGB	85:15
WHOI15	15	24 666	grayscale	ad hoc LOOCV
CrossD	31	64 453	grayscale	ad hoc 2CV
SYKE	50+1	214 309	RGB	evaluation data

SYKE (Kraft et al., 2022) consists of approximately 63 000 training labeled images representing 50 different phytoplankton taxa, meticulously defined, identified, and verified by expert taxonomists. These taxa encompass the most common phytoplankton species/groups found in the Gulf of Finland and the Northern Baltic Proper, following the taxonomy outlined in the Checklist of Baltic Sea Phytoplankton Species and the World Register of Marine Species (Hallfors, 2004). The dataset was collected from various locations in the Baltic Sea across multiple years to capture spatio-temporal variations in plankton communities (2016–2019). An evaluation set, created by considering the annotated samples of natural communities during an annual cycle, consisting of about 150 000 images, 50% of which do not belong to any training class, is used to assess performance. Images in the dataset exhibit wide morphological diversity, resulting in significant variations in size and aspect ratios. Image dimensions range from tens to hundreds of pixels vertically and tens to over a thousand pixels horizontally. Annotation of the images was performed manually with some samples fully labeled and others partially labeled to expand the labeled sets of certain species. Consequently, the dataset may not accurately represent real-life species proportions, but the number of images per species still provides insights into their prevalence in natural populations. The dataset is publicly available.[6]

[3] Available from: https://opendata.eawag.ch/dataset/deep-learning-classification-of-zooplankton-from-lakes.

[4] Available from: https://github.com/Malga-Vision/Anomaly-detection-in-feature-space-for-detecting-changes-in-phytoplankton-populations/tree/main/data.

[5] Available from: https://etsin.fairdata.fi/dataset/a53a55a9-a591-404a-a372-d657d7efb89f/data.

[6] https://doi.org/10.23728/b2share.abf913e5a6ad47e6baa273ae0ed6617a

The diversity of datasets used in this study plays a crucial role in enhancing and evaluating the generalization capability of our models. Each dataset presents unique challenges, including variations in imaging conditions, species distributions, and taxonomic coverage. By training and evaluating our ensemble across multiple datasets, we ensure that the models are exposed to a wide range of plankton morphologies and environmental conditions, reducing the risk of overfitting to specific datasets.

For instance, the WHOI22 dataset consists of high-quality grayscale images captured using flow cytometry, while ZooScan20 contains images acquired with Zooscan technology, which introduces different noise patterns and imaging artifacts. Kaggle-38 and LakeZoo, on the other hand, provide RGB images with varying degrees of illumination and contrast, reflecting real-world challenges in plankton image analysis. The SYKE dataset further enhances dataset diversity by incorporating a large number of phytoplankton species from the Baltic Sea, adding complexity in distinguishing between morphologically similar taxa. In the WHOI15 dataset, the testing protocol segments the data based on the year of acquisition, adding complexity and making the evaluation more reflective of real-world applications. The CrossD dataset presents the greatest challenge among those tested, as its training and test images are captured using different techniques, contributing to their inherent complexity. Despite this, the ensemble achieves performance comparable to that of humans.

By leveraging this dataset diversity, our ensemble-based approach improves model robustness and adaptability across different imaging conditions and taxonomic groups. The results demonstrate that our method maintains high classification performance across datasets without requiring dataset-specific fine-tuning, highlighting the strength of ensemble learning in ecological applications.

3.2. Data pre-processing

To ensure consistent input dimensions across the different CNN and Transformer architectures, we applied two pre-processing strategies: (i) square resizing (sqr) and (ii) padding only (pad), both designed to preserve the aspect ratio of plankton images while adapting them to the input size required by each model (ranging from 224 × 224 to 256 × 256 pixels; see Fig. 2).

All images were normalized following the standard RGB procedure (He et al., 2016). During training, data augmentation was applied through random horizontal/vertical flips and rotations. In the CrossD dataset, augmentation was used selectively: applied to all classes for Transformer-based models, and only to classes with fewer than 100 samples for CNN-based models, in order to mitigate class imbalance.

For CNNs in CrossD, pixel values in both training and test sets were adjusted to match the training-set mean. To improve training stability, at each epoch we monitored the loss, and for inference we selected the model from the epoch with the lowest recorded loss. This strategy was necessary to counter convergence instability; without it, performance would degrade significantly.

Square Resizing: This strategy first pads the image to achieve square dimensions while preserving the original aspect ratio. The padded image is then resized to match the required input size of the CNN models. This method is particularly useful when the original image dimensions exceed the CNN input size, as it allows the image to be downscaled while maintaining a standardized input shape. However, resizing can introduce slight distortions, potentially altering plankton morphological features.

Padding Only (pad): In this approach, the image is directly padded to the CNN input size without any intermediate resizing. White pixels are added around the plankton images to achieve the required dimensions while completely preserving their original shape. This method prevents warping or scaling artifacts but introduces additional background areas that could slightly affect feature extraction in some CNN architectures.

In our proposed ensembles, half of the networks use sqr and the other half use pad, ensuring diversity in the input representations across different models. This hybrid approach allows us to leverage the strengths of both strategies: sqr ensures uniform input sizes across networks, while pad preserves shape integrity, which can be crucial for fine-grained morphological analysis in plankton classification.

3.3. Performance indicators

The evaluation of the proposed approaches, and the comparison with the literature, is conducted using widely applied performance indicators in plankton classification problems: error area under the ROC curve, F-measure, and accuracy.

In the statistical analysis of binary classification, the F-measure (also known as F-score) is a measure of a test's accuracy calculated as the harmonic mean of precision and recall. To extend the definition of F-measure to a multi-class problem, the performance indicator is evaluated as the two-class value (one-vs-all) averaged over the number of species. Given C confusion matrices M_c related to the C one-vs-all problems, where each matrix includes the number of true positive samples (TP_c), true negatives (TN_c), false positives (FP_c), and false negatives (FN_c) for each species $c \in [1..C]$, the multi-class F-measure is defined as a function of true positive rate (Recall) and the positive predictive value (Precision):

$$R_c = \frac{TP_c}{TP_c + FN_c}$$

$$P_c = \frac{TP_c}{TP_c + FP_c}$$

$$F\text{-}Measure = 2 \times \frac{P_c \times R_c}{P_c + R_c}, \quad F = \frac{1}{c}\sum_c F_c$$

Accuracy is the ratio between the number of true predictions and the total number of samples:

$$A_c = \frac{TP_c + TN_c}{TP_c + FN_c + FP_c + TN_c}, \quad A = \frac{1}{C}\sum_c A_c$$

The Area Under the ROC Curve (AUC) is a performance metric for binary classification problems. Mathematically, it represents the probability that a classifier will rank a randomly selected positive instance higher than a randomly selected negative instance. Formally, the AUC can be expressed as:

$$AUC_c = \int_0^1 TPR_c(FPR_c)\, d\ FPR_c, AUC = \frac{1}{C}\sum_c AUC_c$$

where $TPR_c(t)$ is the true positive rate (Recall) and $FPR_c(t)$ is the false positive rate as functions of the threshold t.

The Error Area Under the Curve (EAUC) is defined as $1 - AUC$ and measures the area under the curve of errors, providing an indication of the misclassification rate of the classifier. The EAUC value ranges from 0 to 1, where a value closer to 0 indicates better classifier performance, and a value closer to 1 indicates worse classifier performance. In this work, we use values in percentage, therefore EAUC ranges from 0 to 100.

As further validation of the proposed ensembles, we compare them using the Wilcoxon signed-rank test (Mann et al., 1947); it is specifically used to compare paired data samples from individual assessments. Unlike parametric tests, this non-parametric method does not assume a specific underlying distribution, like normal distribution. Instead, it evaluates both the sizes and directions of differences observed between paired measurements.

Table 2
Performance ResNet50 ensembles (EAUC). In parentheses the number of networks used in the ensemble.

ResNet50	WHOI22	Kaggle-38	ZooScan20	LakeZoo	WHOI15	CrossD
E_SGD(10)	0.352	0.593	1.405	0.508	–	–
E_SGD_S(10)	0.329	0.536	1.362	0.500	–	–
E_A(1)	1.336	1.964	6.085	6.583	3.419	31.575
E_A(10)	0.448	0.788	1.925	1.099	0.788	23.685
E_A_Cosine(10)	0.525	0.698	2.228	0.427	0.696	19.833
E_A_dLR(10)	0.541	0.738	1.845	0.741	0.684	23.241
E_A_All(30)	0.386	0.638	1.491	0.421	0.612	20.177
E_A+Decay(20)	0.424	0.701	1.680	0.728	0.663	22.860
E_AV(30)	0.317	0.444	1.024	0.318	**0.418**	**12.311**
E_AV(10)	0.327	0.495	1.080	0.378	0.444	13.272
E_AV + E_A_All(60)	**0.287**	0.438	**1.007**	**0.315**	0.442	14.722
E_AV + E_A_All(30)	0.301	**0.435**	1.058	0.409	0.465	15.279

Table 3
Performance of EfficientNetB0 ensembles (EAUC).

EfficientNetD0	WHOI22	Kaggle-38	ZooScan20	LakeZoo	WHOI15	CrossD
E_SGD(10)	0.577	0.768	1.765	0.855	–	–
E_A(1)	1.195	1.390	3.591	0.932	1.300	25.201
E_A(10)	0.341	0.593	1.281	0.414	0.456	16.895
E_A_Cosine(10)	0.307	0.580	1.112	0.283	0.417	15.902
E_A_dLR(10)	0.351	0.551	1.095	0.302	0.453	15.616
E_A_All(30)	0.284	0.506	1.013	0.265	0.374	14.171
E_A+Decay(20)	0.308	0.524	1.101	0.319	0.421	14.980
E_AV(30)	0.327	0.461	1.114	0.312	0.407	14.322
E_AV(10)	0.359	0.508	1.193	0.368	0.466	14.255
E_AV+E_A_All(60)	**0.266**	**0.412**	**0.945**	**0.241**	**0.351**	**12.815**
E_AV+E_A_All(30)	0.278	0.431	0.964	0.249	0.407	13.334

Table 4
Performance of MobileNetV2 ensembles (EAUC).

MobileNetV2	WHOI22	Kaggle-38	ZooScan20	LakeZoo	WHOI15	CrossD
E_SGD(10)	0.393	0.568	1.570	0.600	–	–
E_SGD_S(10)	0.385	0.554	1.449	0.525	–	–
E_A(1)	1.231	1.206	3.468	1.463	1.645	26.345
E_A(10)	0.362	0.542	1.244	0.432	0.466	17.028
E_A_Cosine(10)	0.301	0.488	1.128	0.445	0.481	17.259
E_A_dLR(10)	0.338	0.541	1.352	0.333	0.503	18.410
E_A_All(30)	0.271	0.455	1.070	0.324	0.420	15.714
E_A+Decay(20)	0.313	0.495	1.160	0.331	0.443	16.572
E_AV(30)	0.325	0.435	1.092	0.329	0.388	**12.552**
E_AV(10)	0.346	0.467	1.192	0.420	0.458	14.531
E_AV+E_A_All(60)	**0.263**	**0.406**	**0.985**	**0.306**	**0.373**	12.853
E_AV+E_A_All(30)	0.284	0.424	1.016	0.332	0.402	13.101

3.4. Experimental settings

We conducted a series of experiments to evaluate the performance of three well-known CNN architectures (i.e. ResNet50, EfficientNetD0 and MobileNetV2), each combined with various optimizers and learning rate adjustment techniques. This comprehensive approach enabled us to assess the effectiveness of these combinations against established baseline methods. The results of these experiments are detailed in Tables 2, 3, and 4. Specifically, we compared several variants of the Adam optimizer with the original Adam and stochastic gradient descent (SGD).

We tested three different learning rate update strategies. The first one employs a constant value of 0.001. The second one makes use of Cosine Annealing, that starts with a high value and progressively reduces it to a minimum, and then swiftly increases it again, simulating a restart in the learning process. The adjustment follows the rule:

$$\eta_t = \eta^i_{min} + \frac{1}{2}\left(\eta^i_{max} - \eta^i_{min}\right)\left(1 + \cos\left(\frac{T_{cur}}{T_i}\pi\right)\right)$$

where $\eta^i_{min} = 0.0001$; $\eta^i_{max} = 0.001$; $T_i = 0.2$; T_{cur} is iteration/maxEpochs, where maxEpochs is the total number of training epochs and iteration is the number of training iteration, considering all the various epochs, thus a value from 0 to n_{it} ·maxEpochs, where n_{it} is the number of iterations in each training epoch. The third learning rate update strategy is Cosine Decay, which reduces the learning rate following a sinusoidal pattern:

$$learning\,Rate = 0.5\,(1 + \cos(T\pi))\,\eta$$

where $\eta = 0.001$; T = iteration / (2 ·maxEpochs).

3.5. Experimental results

In the first set of tests, we compared the approaches using EAUC, which considers the logits of the network and not just the predicted species. It is important to remember that the activation functions of each layer are chosen randomly in the stochastic method, so all networks in the related ensemble are different in that they use different activation functions. In Tables 2–4 the following ensemble methods are reported for each of the selected CNN architecture (ResNet50, EfficientNetD0, and MobileNet):

- **E_SGD(10)**: 10 networks trained using the SGD optimizer. This method serves as a baseline for comparison due to its widespread use in training deep neural networks;
- **E_SGD_S(10)**: 10 networks trained using the ensemble proposed in stochastic SGD optimizer (Nanni et al., 2022a). Note that EfficientNetB0 is not based on ReLU; trying to couple the method

proposed in Nanni et al. (2022a) with that topology the network does not converge, so the performance of this approach is not reported for EfficientNetB0;
- **E_A(x)**: x networks trained using the standard Adam optimizer with a constant learning rate;
- **E_A_Cosine(10)**: 10 networks trained using the standard Adam optimizer with a cosine annealing learning rate. This approach aims to improve convergence by gradually reducing the learning rate;
- **E_A_decayLR(10)**: 10 networks trained using the standard Adam optimizer with a cosine decay learning rate;
- **E_A_All(30)**: Ensemble of 30 nets obtained combining E_A(10), E_A_Cosine(10), and E_A_decayLR(10) using a sum rule;
- **E_A+Decay(20)**: Ensemble of 20 nets obtained combining E_A(10) and E_A_decayLR(10) using a sum rule;
- **E_AV(30)**: 10 networks trained for each of the three variants of Adam described in Section 2 (Hyp Adam, MinD Adam, AIO Adam), each with a constant learning rate of 0.001, for reasons of time, we did not run tests with other learning strategies. These variants are tested to investigate their performance improvements over standard Adam;
- **E_AV(10)**: 3 networks trained for each of the three variants of Adam described in Section 2 (Hyp Adam, MinD Adam, AIO Adam), each with a constant learning rate of 0.001. The 10-th net is randomly extracted among the nets that create E_AV(30). This ensures the ensemble size of E_A(10) is the same as E_AV(10) (both based on constant learning rate), allowing for fair comparisons;
- **E_AV+E_A_All(60)**: Ensemble of 60 networks obtained combining E_AV(30) and E_A_All(30) using a sum rule. This approach seeks to harness the benefits of multiple Adam variants and learning rate strategies;
- **E_AV+E_A_All(30)**: Similar to E_AV+E_A_All(60), but retaining only 50% of the networks for each approach. This ensures the ensemble size of E_AV+E_A_All(30) is the same as E_AV(30) and E_A_All(30), allowing for fair comparisons.

The following conclusions can be drawn from the results reported in Tables 2, 3, and 4:

- E_SGD_S(10) outperforms E_SGD(10): This indicates that the stochastic variant increases the diversity among the networks, which is beneficial in ensemble methods. However, E_SGD_S(10) only outperforms E_A(10) when using the ResNet50 topology. For other topologies, E_A(10) outperforms e_SGD_S(10). As future work, we will pair the idea proposed in Nanni et al. (2022a) for Adam and not just SGD.
- No clear winner can be found among learning rate strategies: the best performing method varies depending on the dataset and network topology, suggesting that the effectiveness of a learning rate strategy is context-dependent.
- E_AV(30) vs. E_A_All(30): E_AV(30) clearly outperforms E_A_All (30) only with the ResNet50 topology. For other topologies, E_AV(30) and E_A_Al(30)l perform similarly. Overall, the performance difference between E_A_All(30) and E_AV(30) across all topologies has a p-value of 0.51, indicating that we cannot reject the null hypothesis that the two approaches have similar performance.
- E_AV+E_A_All(30) outperforms both E_A_All(30) and E_AV(30): this method achieves a significant improvement with a p-value of 0.05, confirming that the proposed approach is an effective strategy for building an ensemble of CNNs by leveraging both methods while maintaining an optimal ensemble size. We also tried to couple the different learning strategies to Adam variants, but without getting interesting results; these methods independently adjust the learning rate for each parameter, so the learning rate strategies created for Adam are certainly not optimal for these variants.
- E_AV(10) outperforms E_A(10), both based on constant learning strategy: this method achieves a significant improvement with a p-value of 0.01, confirming that the proposed approach is an effective strategy for building an ensemble of CNNs by leveraging both methods while maintaining an optimal ensemble size.

In the next set of tests, shown in Table 5, we compare different vision transformer architectures. Specifically, the methods are named as $X_L(y)$ or X(y), where X denotes a network topology from the tested set of transformers (ViT, DeiT, Swin Transformer, CoaT, and BEiT2), the subscript $_L$ indicates the "large" version of the models (the "base" version is used when $_L$ is missing) and y is the number of networks. All the model use an input dimension of 224 pixels, except Swin_L and ViT_L which use an input dimension of 384 pixels. ViT_L uses patch size of 32 and Swin_L uses patch size 4 and window size of 12. Our internal tests (not reported here for brevity) have shown that using a larger input size improves performance, hence we use the larger size when a pre-trained network with such size was available. The expression "all(y)" represents the sum rule combining the y networks for each topology (ViT, DeiT, Swin Transformer, CoaT, and BEiT2), while "all_L(y)" combines only the topology whose input size is 384 (Swin_L, Vit_L and BEiT2_L).

For vision transformers, we also tested various Adam variants. However, we did not observe any statistically significant difference in performance. Therefore, for sake of space, we only report the results obtained with the standard AdamW optimizer. In any case, in papers where transformers are used, AdamW is almost always applied as the optimizer, while most papers where variants of Adam are proposed use CNN as net.

The following conclusions can be drawn considering the results reported in Table 5:

- Combining multiple networks with the same topology boosts the performance of the stand-alone network, X(7)/X_L(7) always outperform X(1)/X_L(1).
- The method $\text{all}_L(7)$ achieves similar performance to all(7) but uses fewer networks, making it our recommended transformer-based ensemble.
- The method $\text{all}_L(3)$ outperforms $\text{BEiT2}_L(7)$ (similar size ensemble), although the improvement achieved by combining different transformer topologies is smaller than that achieved by combining different CNN topologies (see Table 6). Possible explanations are that transformer topologies are more similar to each other and that they achieve performance close to the maximum attainable with the given training set.

Finally, Tables 6, 7, and 8 compare different ensembles proposed here with the literature. In these tables, we report only the performance of methods using the same protocol and method for performance calculation. For example, if performance using 5-fold cross-validation was reported for ZooScan20, it is not included to ensure fair comparison to the 2-fold cross-validation used in this paper. The ensemble approaches compared in the above tables are:

- **FusCNN**: Ensemble of E_AV+E_A_All(60) combining the nets of all the three CNN topologies.
- **FusCNN_r**: Similar to FusCNN, but using only one network out of three to allow a fair comparison with E_AV+E_A_All(60) for individual topologies, ensuring ensembles have the same size.
- **CNN_Trans**: Weighted sum rule between $\text{all}_L(7)$ and FusCNN (i.e., FusCNN + 4 · $\text{all}_L(7)$). To prevent overfitting, we used the weight proposed in Nanni et al. (2023), where CNNs and vision transformers are compared across a large set of datasets.
- **2023Tran**, it is the ensemble proposed in Nanni et al. (2023) created only by transformers whose input size is 224.

The following conclusions can be drawn considering the results reported in Tables 6–8:

Table 5
Performance of transformers ensembles (EAUC). Some methods are not run in LakeZoo/WHOI15/CrossD to reduce computation time.

Methods	WHOI22	Kaggle-38	ZooScan20	LakeZoo	WHOI15	CrossD
Deit(1)	0.747	0.729	1.558	–	–	–
$Swin_L(1)$	0.338	0.619	1.320	0.498	0.416	8.083
$Vit_L(1)$	0.512	0.774	1.780	0.415	0.640	12.564
CoaT(1)	0.479	0.635	1.909	–	–	–
$BEiT2_L(1)$	0.419	0.597	1.738	0.465	0.411	9.180
Deit(7)	0.303	0.476	1.139	–	–	–
$Swin_L(7)$	0.243	0.412	0.904	0.295	0.301	6.532
$Vit_L(7)$	0.258	0.451	0.986	0.299	0.536	8.617
CoaT(7)	0.223	0.441	0.953	–	–	–
$BEiT2_L(7)$	0.222	0.418	0.896	0.322	0.299	6.316
all(1)	0.258	0.418	0.948	–	–	–
all(7)	**0.198**	0.365	**0.748**	–	–	–
$all_L(7)$	0.199	**0.364**	0.785	**0.264**	**0.294**	**5.711**
$all_L(3)$	0.230	0.390	0.834	0.284	0.308	5.765

Table 6
Ensemble between CNNs and Transformers.

Methods	WHOI22	Kaggle-38	ZooScan20	LakeZoo	WHOI15	CrossD
FusCNN	0.243	0.380	0.877	0.247	0.355	11.970
$FusCNN_r$	0.263	0.393	0.884	0.269	0.356	12.223
$all_L(7)$	0.199	0.364	0.785	0.264	0.294	5.711
Nanni et al. (2023)	0.210	0.367	0.796	0.272	–	–
2023Tran	0.226	0.400	0.917	0.320	–	–
CNN_Trans	**0.189**	**0.328**	**0.681**	**0.232**	**0.280**	**4.723**

Table 7
Comparison vs. state-of-the-art methods (Accuracy).

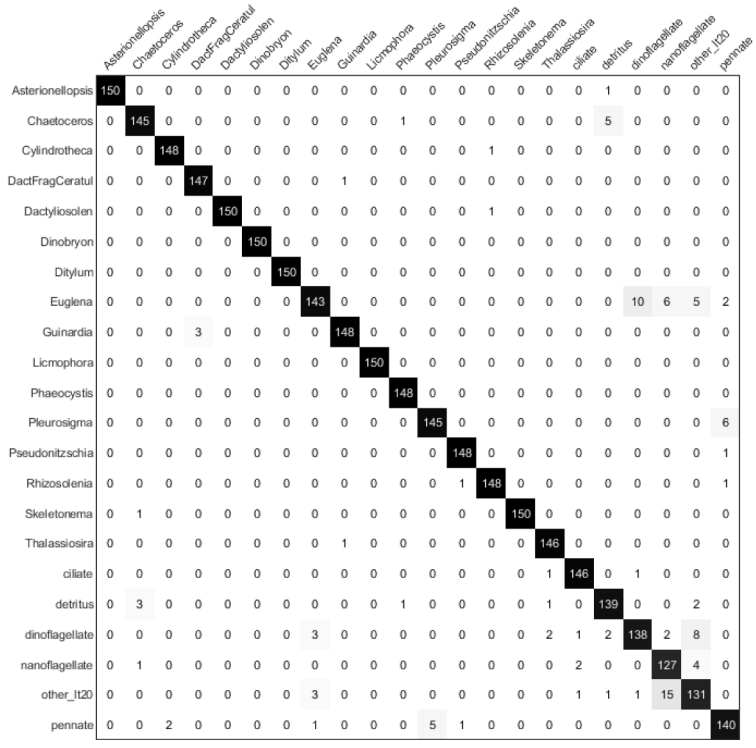
Methods	WHOI22	ZooScan20	Kaggle-38	LakeZoo
CNN_Trans	**96.6**	**90.6**	95.4	97.6
Maracani et al. (2023)	**96.6**	–	**95.5**	–
Nanni et al. (2023)	96.4	90.3	94.9	97.3
Kyathanahally et al. (2021)	96.1	89.8	94.7	**97.7**
Lumini et al. (2020)	95.8	88.8	94.2	–
Lumini and Nanni (2019)	95.3	88.3	94.1	–
Gorsky et al. (2010)	–	78.9	–	–
Sosik and Olson (2007)	88.0	–	–	–

Table 8
Comparison vs. state-of-the-art methods (F1-measure).

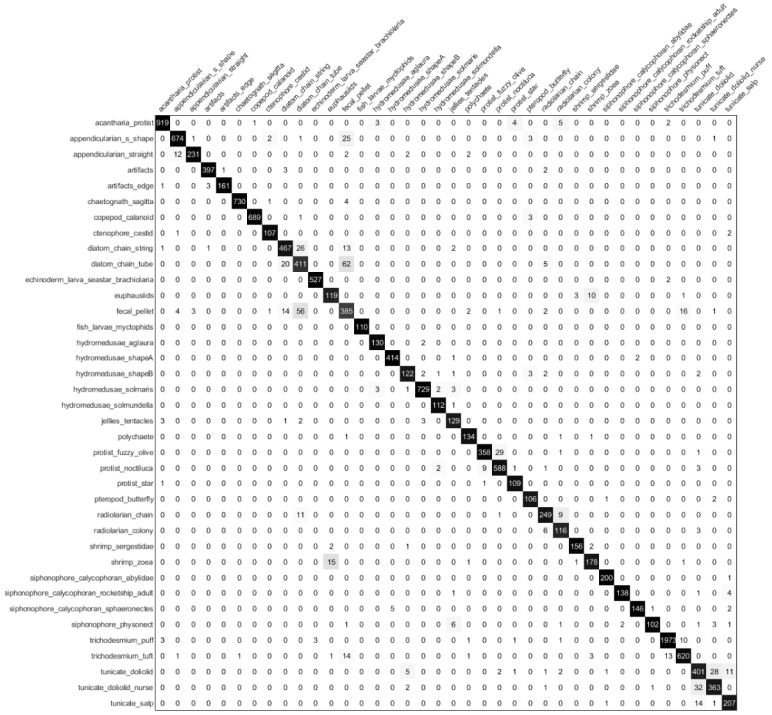
Methods	WHOI22	ZooScan20	Kaggle-38	LakeZoo
CNN_Trans	**0.966**	**0.924**	**0.946**	**0.891**
Maracani et al. (2023)	**0.966**	–	0.945	–
Nanni et al. (2023)	0.964	0.918	0.939	0.889
Lumini et al. (2020)	0.958	0.902	0.927	–
Lumini and Nanni (2019)	0.953	0.897	0.926	–
Zheng et al. (2017)	0.900	0.894	0.846	–
Gorsky et al. (2010)	–	0.787	–	–

- CNN_Trans consistently achieves the highest performance, surpassing $all_l(7)$, indicating that CNNs and Transformers extract complementary information. It is well-documented in the literature that Vision Transformers capture global image information, while CNNs excel at extracting local features (Bbouzidi et al., 2024).
- FusCNN brings the performance of CNN ensembles closer to that of $all_l(7)$ transformer ensembles. In LakeZoo, FusCNN outperforms $all_l(7)$, suggesting that CNNs can compete effectively with Transformers under certain conditions. It is important to note that comparison is not fair since transformers are pretrained using larger input size; interestingly, ensembles of proposed CNN have a performance similar to the ensemble of transformer reported in Nanni et al. (2023).
- CNN_Trans achieves state-of-the-art (SOTA) performance in this domain. Although other SOTA ensembles are significantly smaller in size, our choice to use a more complex system is justified by the fact that our individual networks, despite having the same topology and pre-training approach, do not perform as well as those reported in other studies. This is likely due to the use of many default hyperparameters without dataset-specific optimization to avoid overfitting.
- Evaluating performance from 2007 to the present reveals notable improvements, both in absolute performance and in the breadth of datasets used for validation, enhancing the reliability of comparisons. The transition from pre-2019 hand-crafted feature methods to current deep learning approaches has shown significant advancements. Recent years have seen incremental improvements with the adoption of computationally intensive methods. However, it is noteworthy that current performance levels are approaching those of human experts, as evidenced by confusion matrices provided at the end of the subsection (Figs. 3 and 4), where most errors stem from images that challenge even human experts.
- The WHOI15 and CrossD datasets are relatively new and have seen limited use in the literature. Therefore, we provide only a commentary-based comparison with the original paper that introduced these datasets. In terms of accuracy, our ensemble CNN_Trans model achieves an accuracy of 84.7% in CrossD, the best result reported in Batrakhanov et al. (2024) is 64.8%, using a system trained using CS images and tested in the IFCB images, as in this paper; in WHOI15 we can compare our approach with original paper when images extracted in 2007, 2008, and 2009 are used in the training set, while testing on the samples from 2010. In Ciranni et al. (2024b) reports an average Sensitivity of 0.768 and an average Specificity of 0.725, CNN_Trans obtains largely better results, Sensitivity of 0.9514 and Specificity of 0.9962, respectively. In any case, in Ciranni et al. (2024b) the goal is to create a fast method, suitable for handling and classifying new classes not present in the training set, their aim is not to create a method to achieve the highest possible performance.

Analyzing the confusion matrices helps identify the species that are more challenging to differentiate: nanoflagellate vs. other_lt20 in the WHOI22 dataset, copepoda vs. copepod_petit in the ZooScan20 dataset, and fecal_pellet vs. diatom_chain_tube in the Kaggle38 dataset.

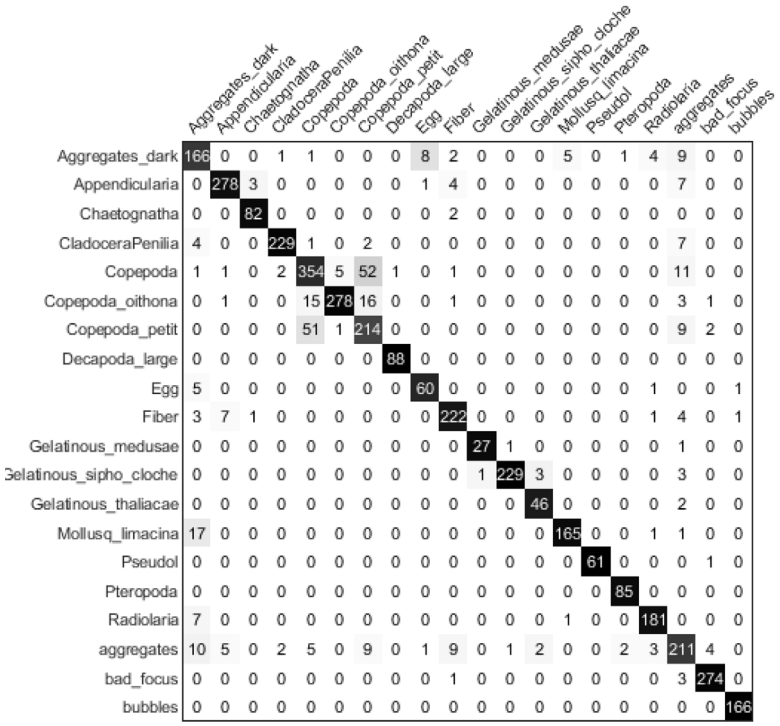


(a) WHOI22

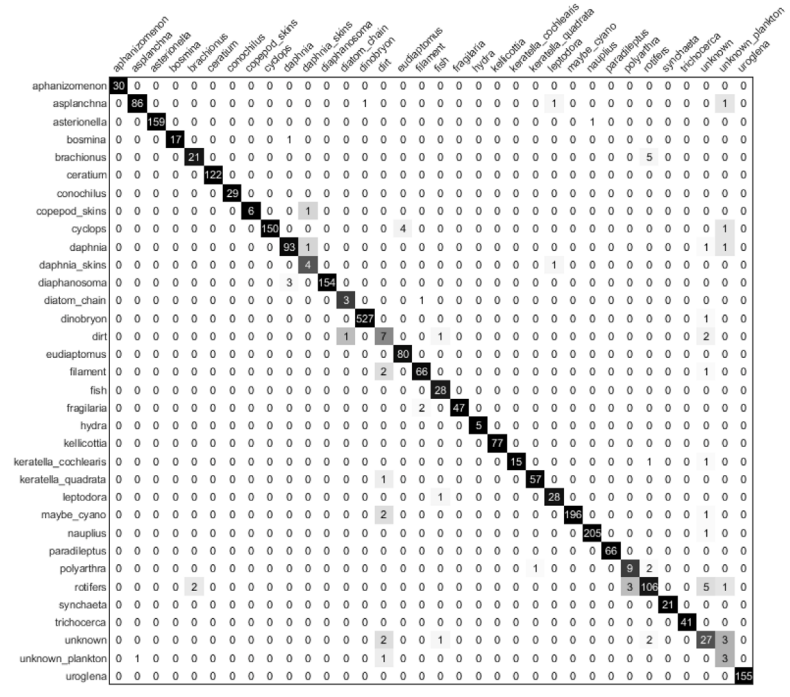


(b) Kaggle-38

Fig. 3. Confusion matrices per the approach CNN_Trans (Part 1).



(a) ZooScan20



(b) LakeZoo

Fig. 4. Confusion matrices per the approach CNN_Trans (Part 2).

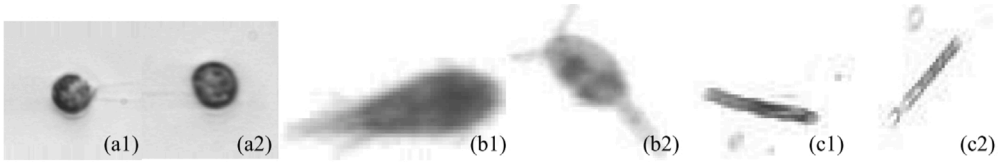


Fig. 5. Misclassified images: (a1–a2) WHOI22 dataset: nanoflagellate vs. other_lt20 (b1–b2) ZooScan20 dataset: copepoda vs. copepod_petit (c1–c2) Kaggle-38 dataset fecal_pellet vs. diatom_chain_tube.

Table 9
P-values obtained using Wilcoxon signed rank test.

	CNN_Trans	$all_L(7)$	FusCNN	$BEiT2_L(7)$
CNN_Trans	–	0.001	0.001	0.001
$all_L(7)$	–	–	0.001	0.003
FusCNN	–	–	–	0.834

Fig. 5 illustrates representative examples from the most frequently misclassified class pairs, with two images per dataset. These examples highlight the high visual similarity between distinct classes—often stemming from taxonomically different categories that exhibit similar morphology or imaging artifacts. Such intra-class visual overlap poses a significant challenge even for human experts, making it difficult for the model to distinguish between them reliably. These misclassifications reflect intrinsic limitations of the datasets, where visually similar samples are assigned to different labels, rather than shortcomings of the classification approach itself.

In Fig. 6 we report some images from CrossD dataset, notice the large intra-class variation.

For better assessing the comparison among the different methods, we report the *p*-value obtained using Wilcoxon signed-rank test (Mann et al., 1947), see Table 9. This test confirms that: an ensemble of diverse transformer architectures outperforms an ensemble of identical transformer architectures; transformers outperform CNNs; a fusion of transformers and CNNs surpasses both individual methods.

We are aware that the improvements in the various ensembles are not remarkable, but as pointed out earlier our goal is to test how much performance can be increased by using ensembles and whether ensembles of CNNs can have similar performance to ensembles of vision transformers; we are focusing in non-edge computing applications, where performant GPUs can be used and in applications where we want to maximize performance, e.g., toxic phytoplankton detection. The computation times for a batch size of 10,000 images, using a Titan V with 24 GB of RAM (2018 GPU) are: 11.3 s - ResNet50; 18.1 s - EfficientNetB0; 10.4 s - MobileNetV2. Considering these results, we can classify 120 000 images in 8 h using the CNN ensemble with a GPU released in 2018; this is a feasible solution considering that all inferences can be performed in parallel, so we can halve the computation time by using two GPUs.

The transformers inference time, using an Nvidia RTX 4090 with 24 GB of RAM and a batch size of 100, are: $BEiT2_L(1)$, 538 ms; $Vit_L(1)$, 365 ms; $Swin_L(1)$, 1040 ms. Therefore, the ensemble of transformers can classify more than 200k images in 8 h. It is also clear that if in a given application we need to perform classification using a low-power device, distillation techniques become imperative to greatly reduce the computational power needed by the ensemble. The best trade-off performance vs computation time is obtained by $BEiT2_L(1)$, it can classify almost 200 images/second with a good performance.

For our final test of this subsection, we report the accuracy of both the BEiT2 and ResNet models while varying the rejection rate. The results are reported in Tables 10–12. We have used the following rejection rate strategy: let us define $\theta_1(x)$ as the highest score among the different species given a pattern x; $\theta_2(x)$ is the second highest score of that pattern; $\theta(x)=\theta_1(x)-\theta_2(x)$, the x% of plankton classified with the lowest value of $\theta(x)$ are rejected.

By discarding 20% of the patterns (i.e. rejection rate of 20%), we can classify 80% of the patterns using the approaches reported in Tables 10–12 and only process the remaining 20% with the CNN_Trans, without any loss in system performance. This holds true for all datasets except CrossD. For instance, if we classify 80% of the patterns using the ensemble E_AV(10) (see Table 12) and the remaining 20% are passed to the full ensemble CNN_Trans, the performance of this two-step approach is nearly identical to that of classifying all patterns using the CNN_Trans, but with a significant reduction in computational time. However, the challenge arises with the CrossD dataset, which involves cross-domain data. In this case, achieving the same performance of CNN_Trans requires using $BEiT2_L$, the performance of the ensemble of CNN, with a rejection rate of 20%, is too low for this goal.

3.6. Results on the SYKE dataset

Finally, we include in this subsection the results on the SYKE dataset (Kraft et al., 2022). Due to its size, we employed a reduced set of networks to manage computational complexity, yet achieving state-of-the-art (SOTA) performance on this dataset. The authors of that dataset use a different performance indicator for assessing the performance, the weighted F-measure, since it is a class-imbalanced dataset.

The SYKE dataset follows a different protocol, including an "open" species not present in the training set. Thus, we established a threshold to assign patterns to specific species: let us define $\theta_1(x)$ as the highest score among the different species given a pattern x; $\theta_2(x)$ is the second highest score of that pattern; $\theta(x)=\theta_1(x)-\theta_2(x)$, if $\theta(x)>\tau$, the plankton is assigned to the predicted species, otherwise, it is assigned to the open class. In all the tests we have used $\tau = 0.9$, it works well for all the topologies.

We evaluated performance following the protocol outlined in Kraft et al. (2022) but considering also the open species for calculating the weighted F-measure. In Table 13 we present the performance of our ensemble E_AV(10), for CNNs, across three different network topologies, as well as the fusion (Ensemble CNN) of these three approaches. Moreover, for a fair comparison we have calculated also the weighted F-measure excluding the open species (as in Kraft et al. (2022)), these results can be compared with those reported in Kraft et al. (2022) where a weighted F-measure of 0.820 is reported. The ensemble of CNNs obtains a weighted F-measure of 0.861; $all_l(7)$ obtains a weighted F-measure of 0.864; CNN_Trans gets the best performance: 0.870.

4. Discussion

Our study extends prior deep learning approaches for plankton classification by leveraging ensemble strategies to improve classification robustness. While previous works have demonstrated the effectiveness of CNNs and Transformers in plankton identification, our approach enhances model adaptability and accuracy through the combination of diverse architectures.

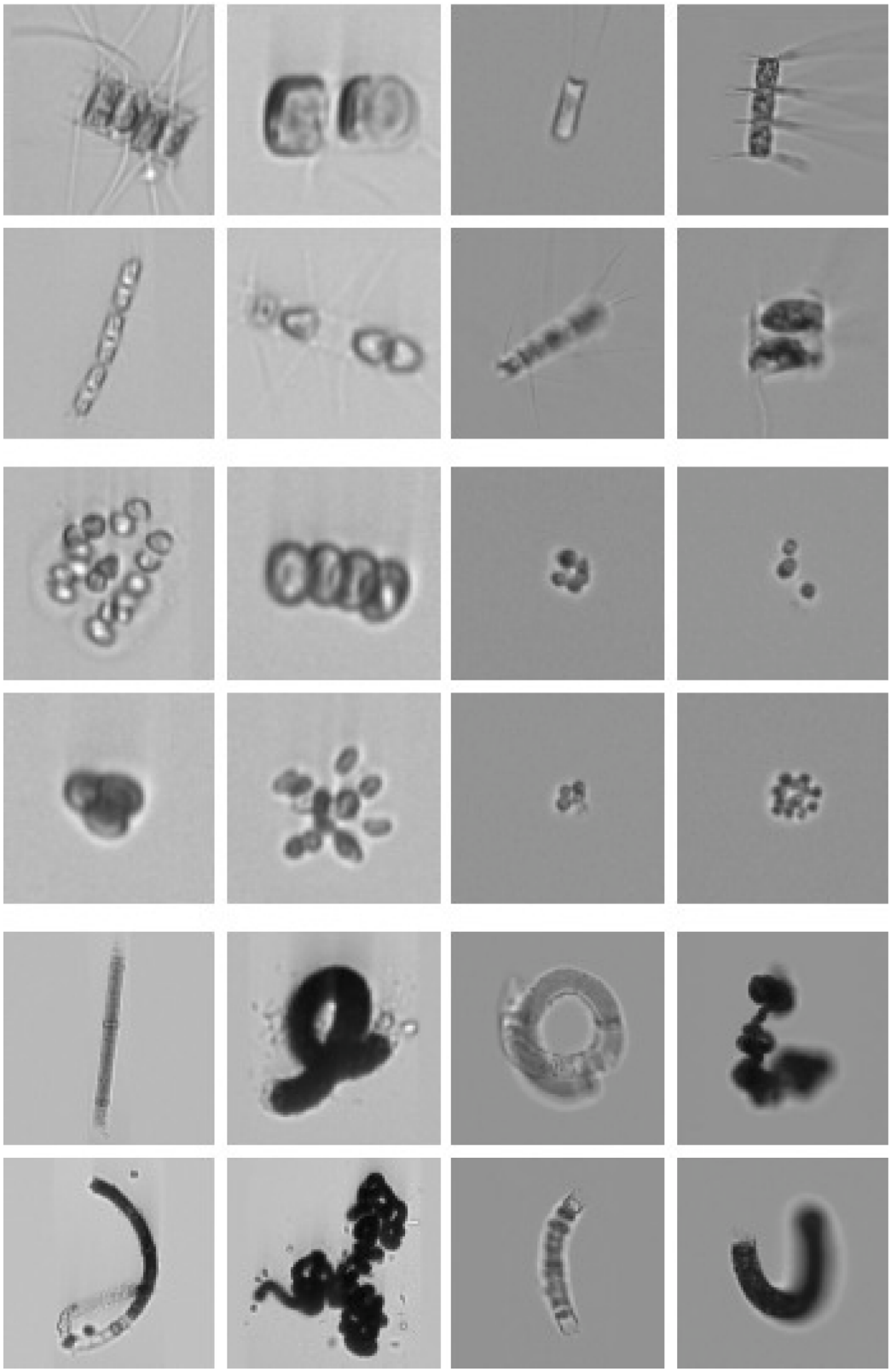


Fig. 6. The first two columns contain test images, while the last two columns contain training images. Each class is represented by every two consecutive rows. There are three classes in total: Chaetoceros sp.; Chlorococcales; Nodularia spumigena.

Table 10
Accuracy varying the rejection %, $BEiT2_L(1)$.

Rejection%	WHOI22	Kaggle-38	ZooScan20	LakeZoo	WHOI15	CrossD
0	95.09	94.55	88.10	96.82	94.86	82.03
20	99.28	99.10	95.00	99.91	99.17	89.37
40	99.75	99.65	97.39	99.94	99.74	93.24
60	99.85	99.84	99.01	99.91	99.87	94.42

Table 11
Accuracy varying the rejection %, ResNet50 - E_AV +E_A_All(60).

Rejection%	WHOI22	Kaggle-38	ZooScan20	LakeZoo	WHOI15	CrossD
0	95.33	94.20	88.49	97.01	94.10	53.59
20	99.28	99.10	95.63	99.77	98.93	60.19
40	99.65	99.58	98.41	99.88	99.49	68.36
60	99.92	99.90	99.60	99.91	99.85	79.68

Table 12
Accuracy varying the rejection %, ResNet50 - E_AV(10).

Rejection%	WHOI22	Kaggle-38	ZooScan20	LakeZoo	WHOI15	CrossD
0	95.27	93.92	88.39	97.01	93.71	61.45
20	99.09	98.87	95.56	99.82	98.85	68.72
40	99.75	99.73	98.23	99.88	99.67	76.02
60	99.92	99.97	99.47	100	99.87	84.26

Table 13
Performance on SYKE dataset (Weighted F-measure).

Topology	Weighted F-measure
E_AV(10) ResNet	0.831
E_AV(10) EfficientNet	0.825
E_AV(10) MobileNet	0.847
$Swin_L(7)$	0.845
$ViT_L(7)$	0.822
$BEiT2_L(7)$	0.817
Ensemble CNN (3 topologies)	0.854
$all_l(7)$	0.853
CNN_Trans	**0.860**

4.1. Comparative analysis of algorithms

To further contextualize our findings, we provide a comparative analysis between CNN-based and transformer-based architectures. CNNs excel in extracting local spatial features, while transformers are better suited for capturing long-range dependencies. Our results indicate that ensembles leveraging both architectures achieve superior performance. Details about computational efficiency and dataset adaptability is also discussed in the next section.

This study also focuses on the impact of different optimization strategies, confirming that varying Adam-based optimization techniques contributes to improve ensemble classification accuracy. This suggests that ensemble learning benefits from diverse optimization strategies, that mitigate local minima convergence issues.

Our results further confirm the superiority of ensemble methods over individual architectures. Despite the high performance of single Transformer models, our experiments show that combining multiple models leads to significant improvements in accuracy and robustness. This is consistent with previous studies on CNN-based ensembles, that demonstrated to be more robust against dataset variability and overfitting.

The ensemble approach enhances performance by integrating complementary strengths of CNNs and Transformers. While CNNs excel in local feature extraction and spatial pattern recognition, Transformers capture long-range dependencies and contextual relationships. Combining both paradigms, the ensemble provides a more comprehensive feature representation, leading to more stable and reliable classification across multiple datasets. These findings reinforce the importance of ensemble learning in plankton classification and, more broadly, in ecological monitoring applications.

4.2. Trade-offs in model selection

Choosing the right deep learning model for plankton classification involves balancing accuracy, computational efficiency, and dataset adaptability. CNNs, particularly lightweight architectures, offer fast inference times and lower computational costs, making them ideal for real-time and edge-computing applications. However, their reliance on localized feature extraction may limit their adaptability to diverse datasets.

In contrast, Vision Transformers provide improved accuracy by capturing long-range dependencies, but their computational costs make them less suitable for real-time applications or resource-constrained environments.

Dataset variability substantially complicates model selection, as plankton images exhibit significant differences in imaging conditions, species imbalance, and morphological diversity. Our experiments demonstrate that ensembles combining diverse architectures improve feature extraction and robustness against dataset changes. Future work should explore adaptive learning strategies to further optimize model generalization across different plankton communities.

While ensemble learning improves classification robustness and generalization, it also increases computational requirements compared to individual models. Table 14 compares the number of parameters and parameter memory in the models considered, highlighting the trade-off between model complexity and performance.

As shown in Table 14, Transformers methods significantly increase the total number of parameters, making them computationally more demanding than CNN models. This highlights a key limitation of our approach: ensembles require substantial GPU resources, limiting their feasibility for real-time or edge computing applications. One potential solution is the use of model distillation techniques, where a smaller network is trained to replicate the performance of an ensemble, reducing inference costs while maintaining high classification accuracy. Future work will explore such strategies to improve the deployability of ensemble-based plankton classification models.

4.3. Interpretation of embedding distributions via t-SNE

Fig. 7 shows a 2D t-SNE projection of the feature embeddings extracted from the ensemble model for a the Kaggle-38 dataset. Each point corresponds to a sample, color-coded by its ground-truth label. The projection offers qualitative insights into the model's ability to

Table 14
Comparison of the number of parameters for individual models and ensembles. The ensemble approach increases computational cost but improves classification robustness.

Model	Number of parameters (Millions)	Parameter memory (MB)
ResNet50	25.6M	98 MB
EfficientNetB0	5.3M	20 MB
MobileNetV2	3.4M	14 MB
ViT_L	305.7M	1166.0 MB
$Swin_L$	195.3M	727.4 MB
$BEiT2_L$	303.5	1157.6 MB

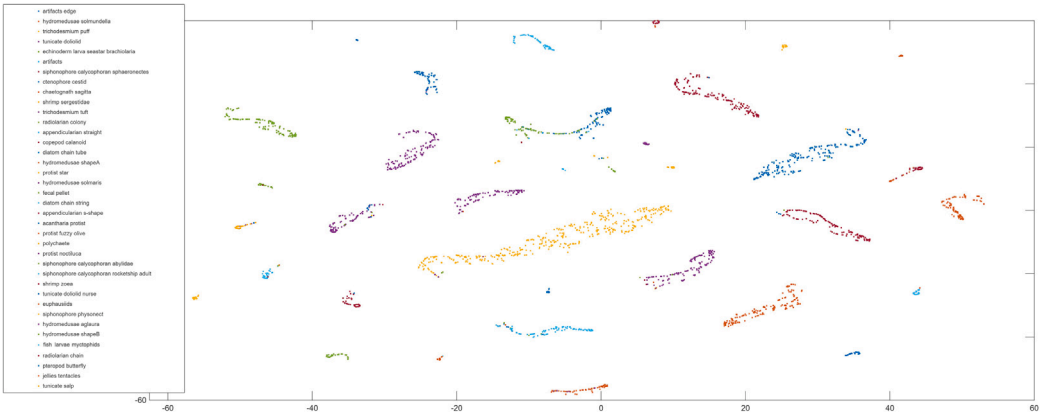


Fig. 7. t-SNE projection of the samples from the Kaggle-38 dataset.

learn discriminative representations and reveals distinct spatial arrangements that reflect both inter-class separability and intra-class cohesion.

Notably, some classes form very tight and isolated clusters, indicating strong internal consistency and high discriminative power of the learned features. Examples include Shrimp sergestidae and Jellies tentacles, whose points are compact and well-separated from other clusters, suggesting that these classes are highly distinctive in terms of visual features.

In contrast, classes such as Trichodesmium puff exhibit a broad and dispersed embedding footprint, reflecting significant intra-class variability. Despite this large spread, the cluster remains distinct from others, indicating that the model successfully captures the class identity despite structural diversity among its instances.

Some degree of proximity or partial overlap is also evident among certain classes, which may reflect morphological similarities and potential sources of confusion. For instance, Fecal pellet and Diatom chain tube – already highlighted in previous figures – appear close in the embedding space. A similar spatial adjacency is observed between Protist fuzzy olive and Protist noctiluca, as well as between Tunicate doliolid and Tunicate doliolid nurse. These cases might benefit from refined feature modeling or hierarchical classification schemes to better handle subtle inter-class boundaries.

Overall, the t-SNE projection provides visual confirmation of the model's capacity to learn meaningful and structured representations, with a generally favorable trade-off between intra-class compactness and inter-class separation. However, the analysis also highlights the presence of ambiguous class pairs that may warrant further investigation.

4.4. Discussion on Adam variants

In this study, we evaluated various optimization techniques aimed at efficiently locating a favorable minimum by leveraging their distinct approaches. Consequently, combining these methods into an ensemble improves performance.

Our tested strategies operate by assessing the absolute disparity between the current parameter gradients and their moving average squared. This approach enhances resilience against fluctuations in gradient differences across iterations compared to methods like Adam and DiffGrad.

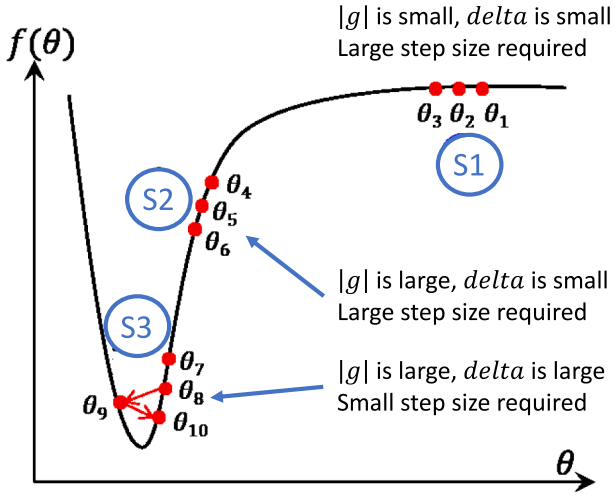


Fig. 8. A typical scenario in optimization that demonstrates the importance of adjusting parameters adaptively during the optimization process.

As outlined in the literature, an optimal parameter optimization method should adhere to the guidelines illustrated in Fig. 8 (Dubey et al., 2022), where Θ is the parameters tensor, $delta$ is the difference of the parameters between two training iterations, g are the gradients.

In the flat region (S1), an optimal optimizer should take a substantial step to move away from the flat area. In the large gradient — small delta area (S2), it is crucial to use a large step size for faster convergence. In the steep and narrow valley (S3), where the minimum resides, both gradients and $delta$ are significant, necessitating a small step size to accurately pinpoint the minimum and minimize oscillations.

The Adam variants approach possess these properties:

- The Hyp optimizer builds upon DGrad/DiffGrad concepts, aiming for a gradient that changes smoothly as it approaches the minimum (referenced as S3). Unlike DGrad/DiffGrad, which employs a sigmoid function confining values of ξ_t between 0.5 and 1, Hyp utilizes a function (see Eq. (12)) that confines values of ξ_t between 0 and 1. One drawback of Hyp is its tendency to update parameters minimally when gradient variations approach zero, potentially leading to slower convergence compared to DGrad/DiffGrad in the S1 region, but the values do not saturate quickly to 1 as when (9) is used, i.e. formula (9) quickly saturates at 1.
- MinD operates as a minimalistic rule, aiming to decrease and smooth out steps to mitigate the risk of overlooking a minimum point. However, this method may lead to slower convergence in the S1 region.
- The AIO optimizer adjusts the step size based on both the magnitude and direction of the gradient vector, as described in (15). By taking the minimum between two angular coefficients, the optimizer becomes more resilient to changes in the curvature of the shape. Furthermore, we incorporate curvature information as a weighting factor to introduce second-order momentum into the update rule, detailed in (20) and (23).
  In region S1, where gradients are small, the step size is sufficiently large due to the small denominator in (23). It is important to note that $delta$ affects only the numerator of (23), with $delta < 0$ for $g_t > 0$ and $delta > 0$ for $g_t < 0$.
  In region S2, where $|g_t|$ is large, the value of (20) remains significant despite a small $delta$. The inclusion of $delta$ amplifies (20), resulting in larger steps compared to Adam or Hyp.
  In region S3, where both $|g_t|$ and $delta$ are large, the optimizer takes larger steps due to their relationship highlighted earlier, in summary, AIO outperforms Hyp/DGrad in regions S1 and S2 but performs worse in region S3.

The implications of our findings go beyond classification accuracy; they highlight the potential of ensemble deep learning models for real-world ecological applications, including biodiversity monitoring and ecosystem health assessment.

4.5. Contextualization in ecological informatics

Our work aligns with recent advancements in ecological informatics, where AI-driven models are increasingly used for marine and freshwater ecosystem monitoring. We compare our approach with previous studies that employ deep learning for plankton classification, demonstrating that our ensemble-based methodology provides a competitive advantage in accuracy and robustness.

Furthermore, a promising direction for future research is the integration of additional ecological parameters, such as fluorescence data or nutrient concentration, to refine functional plankton classification. By incorporating ecological parameters with AI-driven classification, future models could bridge the gap between taxonomy-based and function-based plankton monitoring, leading to more actionable insights for marine and freshwater ecosystem management.

The results in this study highlight that ensemble-based approaches not only achieve state-of-the-art accuracy but also improve classification consistency across different datasets. This is a crucial aspect for ecological monitoring applications, where variations in imaging conditions and species distributions can significantly impact model performance. Further research should investigate additional ensemble techniques, such as weighted voting or meta-learning strategies, to further optimize classification reliability in real-world scenarios.

Despite the strong performance of our ensemble models, a key challenge in this field is the reliance on benchmark datasets, some of which contain a relatively small number of samples. While these datasets are widely used and enable fair comparisons with existing methods, their limited size may not fully capture the complexity and variability of real-world plankton images. This constraint affects all approaches addressing this task and highlights the need for more extensive and diverse datasets in the future.

5. Conclusions

Plankton image analysis is a challenging task due to several inherent factors. First, a large number of different species can be counted, which means that plankton images can contain a vast array of different objects and species, each belonging to a different category. This diversity requires extensive training data to accurately classify and identify each species. Second, there is great intra-species variance. Elements belonging to the same species can show a substantially different appearance due to various factors such as age, size, orientation, and even biological variations. For example, two plankton specimens of the same species might have different color patterns or be seen from different angles, making them look distinct. Third, there might be small extra-species differences, with elements from different species having similar appearance, challenging the classifier.

In this paper, we proposed computer vision systems for plankton classification based on the integration of several CNN topologies trained with different Adam optimization methods. We also compared such systems with Transformer networks based on multiple architectures. The proposed ensemble, comprising both CNN and Transformer models, demonstrated superior performance compared to existing methods on several benchmarks.

While our approach significantly improves classification accuracy and robustness, it also presents some limitations. Ensemble methods inherently increase computational complexity, requiring substantial GPU resources for training and inference. This limits their feasibility for real-time and edge computing applications, particularly in scenarios where computational resources are constrained. Additionally, while ensembles enhance model generalization, their training process is time-intensive, which poses challenges for adapting to newly introduced plankton species.

Future research will focus on several key areas to further refine our approach. One promising direction is the exploration of model distillation techniques, where a smaller, more efficient model is trained to replicate the performance of an ensemble, reducing inference costs while maintaining high classification accuracy. Additionally, we aim to investigate adaptive learning strategies that can improve generalization across diverse plankton datasets and facilitate continual learning as new species are identified. Further efforts will include optimizing ensemble design, refining data augmentation techniques, and integrating external ecological parameters to enhance functional classification.

While inference times are no longer a significant concern, the real challenge arises when new species must be regularly added. In such cases, the long training times become a limitation, making it necessary to adopt continual learning methods to efficiently manage the evolving model. This will allow models to dynamically adapt to new species while retaining previously learned classifications.

All the code used in our experiments will be freely available in our Github repository[7] (Nanni, 2024) in order to reproduce the experiments reported and for future comparisons.

CRediT authorship contribution statement

Loris Nanni: Writing – review & editing, Writing – original draft, Validation, Supervision, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Alessandra Lumini:** Writing – review & editing, Writing – original draft, Validation, Supervision, Software, Conceptualization. **Leonardo Barcellona:** Writing – review & editing, Writing – original draft, Software. **Stefano Ghidoni:** Writing – review & editing, Writing – original draft.

[7] Available from: https://github.com/LorisNanni/Convolutional-neural-networks-and-vision-transformers-for-Plankton-Classification.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We would like to acknowledge the support that NVIDIA provided us through the GPU Grant Program. We used a donated TitanX GPU to train deep networks used in this work.

Data availability

The source code of our method is openly available, and the datasets used are publicly accessible from previous literature.

References

Bachimanchi, H., Pinder, M.I.M., Robert, C., Wit, P.D., Havenhand, J., Kinnby, A., Midtvedt, D., Selander, E., Volpe, G., 2023. Deep-learning-powered data analysis in plankton ecology. URL https://arxiv.org/abs/2309.08500. arXiv:2309.08500.

Batrakhanov, D., Eerola, T., Kraft, K., Haraguchi, L., Lensu, L., Suikkanen, S., Camarena-Gómez, M.T., Seppälä, J., Kälviäinen, H., 2024. Daplankton: Benchmark dataset for multi-instrument plankton recognition via fine-grained domain adaptation. In: 2024 IEEE International Conference on Image Processing. ICIP, pp. 158–164. http://dx.doi.org/10.1109/ICIP51287.2024.10648228.

Bbouzidi, S., Hcini, G., Jdey, I., Drira, F., 2024. Convolutional neural networks and vision transformers for fashion mnist classification: A literature review. URL https://arxiv.org/abs/2406.03478. arXiv:2406.03478.

Chong, J.W.R., Khoo, K.S., Chew, K.W., Ting, H.Y., Iwamoto, K., Ruan, R., Ma, Z., Show, P.L., 2024a. Artificial intelligence-driven microalgae autotrophic batch cultivation: A comparative study of machine and deep learning-based image classification models. Algal Res. 79, 103400. http://dx.doi.org/10.1016/j.algal.2024.103400, URL https://www.sciencedirect.com/science/article/pii/S2211926424000122.

Chong, J.W.R., Khoo, K.S., Chew, K.W., Ting, H.Y., Iwamoto, K., Show, P.L., 2024b. Digitalised prediction of blue pigment content from spirulina platensis: Next-generation microalgae bio-molecule detection. Algal Res. 83, 103642. http://dx.doi.org/10.1016/j.algal.2024.103642, URL https://www.sciencedirect.com/science/article/pii/S2211926424002546.

Ciranni, M., Murino, V., Odone, F., Pastore, V.P., 2024a. Computer vision and deep learning meet plankton: Milestones and future directions. Image Vis. Comput. 143, 104934. http://dx.doi.org/10.1016/j.imavis.2024.104934, URL https://www.sciencedirect.com/science/article/pii/S0262885624000374.

Ciranni, M., Odone, F., Pastore, V.P., 2024b. Anomaly detection in feature space for detecting changes in phytoplankton populations. Front. Mar. Sci. 10, http://dx.doi.org/10.3389/fmars.2023.1283265, URL https://www.frontiersin.org/journals/marine-science/articles/10.3389/fmars.2023.1283265.

Dai, Z., Liu, H., Le, Q.V., Tan, M., 2021. Coatnet: Marrying convolution and attention for all data sizes. Adv. Neural Inf. Process. Syst. 34, 3965–3977.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al., 2020. An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929.

Dubey, S.R., Basha, S.H.S., Singh, S.K., Chaudhuri, B.B., 2022. AdaInject: Injection based adaptive gradient descent optimizers for convolutional neural networks. arXiv:2109.12504.

Dubey, S.R., Chakraborty, S., Roy, S.K., Mukherjee, S., Singh, S.K., Chaudhuri, B.B., 2019. Diffgrad: an optimization method for convolutional neural networks. IEEE Trans. Neural Networks Learn. Syst. 31 (11), 4500–4511.

Eerola, T., Batrakhanov, D., Barazandeh, N.V., Kraft, K., Haraguchi, L., Lensu, L., Suikkanen, S., Seppälä, J., Tamminen, T., Kälviäinen, H., 2024. Survey of automatic plankton image recognition: challenges, existing solutions and future perspectives. Artif. Intell. Rev. 57 (5), 114.

Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press, http://www.deeplearningbook.org.

Gorsky, G., Ohman, M.D., Picheral, M., Gasparini, S., Stemmann, L., Romagnan, J.-B., Cawood, A., Pesant, S., García-Comas, C., Prejger, F., 2010. Digital zooplankton image analysis using the ZooScan integrated system. J. Plankton Res. 32 (3), 285–303. http://dx.doi.org/10.1093/plankt/fbp124, arXiv:https://academic.oup.com/plankt/article-pdf/32/3/285/4394627/fbp124.pdf.

Hallfors, G., 2004. Checklist of Baltic Sea Phytoplankton Species (Including Some Heterotrophic Protistan Groups). Technical Report, Helsinki Commission, Baltic Marine Environment Protection Commission, Helsinki, Finland.

Hays, G.C., Richardson, A.J., Robinson, C., 2005. Climate change and marine plankton. Trends Ecol. Evolut. 20 (6), 337–344.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778.

Henson, S.A., Cael, B.B., Allen, S.R., Dutkiewicz, S., 2021. Future phytoplankton diversity in a changing climate. Nat. Commun. 12 (1), 5372. http://dx.doi.org/10.1038/s41467-021-25699-w.

Khaldi, A., Khaldi, R., 2024. Recognition of harmful phytoplankton from microscopic images using deep learning. URL https://arxiv.org/abs/2409.12900, arXiv:2409.12900.

Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Kraft, K., Velhonoja, O., Eerola, T., Suikkanen, S., Tamminen, T., Haraguchi, L., Ylöstalo, P., Kielosto, S., Johansson, M., Lensu, L., Kälviäinen, H., Haario, H., Seppälä, J., 2022. Towards operational phytoplankton recognition with automated high-throughput imaging, near-real-time data processing, and convolutional neural networks. Front. Mar. Sci. 9, http://dx.doi.org/10.3389/fmars.2022.867695, URL https://www.frontiersin.org/articles/10.3389/fmars.2022.867695.

Kyathanahally, S.P., Hardeman, T., Merz, E., Bulas, T., Reyes, M., Isles, P., Pomati, F., Baity-Jesi, M., 2021. Deep learning classification of lake zooplankton. Front. Microbiol. 3226.

Larkum, A., Orth, R., Duarte, C., 2006. Seagrasses: Biology, ecology and conservation. http://dx.doi.org/10.1007/978-1-4020-2983-7.

Lee, H., Park, M., Kim, J., 2016. Plankton classification on imbalanced large scale database via convolutional neural networks with transfer learning. In: 2016 IEEE International Conference on Image Processing. ICIP, IEEE, pp. 3713–3717.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B., 2021. Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 10012–10022.

Lumini, A., Nanni, L., 2019. Deep learning and transfer learning features for plankton classification. Ecol. Informatics 51, 33–43.

Lumini, A., Nanni, L., Maguolo, G., 2020. Deep learning for plankton and coral classification. Appl. Comput. Informatics ahead-of-print.

Mann, H., Whitney, D., et al., 1947. On a test of whether one of two random variables is stochastically larger than the other. Ann. Math. Stat. 18 (1), 50–60.

Maracani, A., Pastore, V.P., Natale, L., Rosasco, L., Odone, F., 2023. In-domain versus out-of-domain transfer learning in plankton image classification. Sci. Rep. 13 (1), 10443. http://dx.doi.org/10.1038/s41598-023-37627-7.

Nanni, L., 2024. Convolutional neural networks and vision transformers for plankton classification. https://github.com/LorisNanni/Convolutional-neural-networks-and-vision-transformers-for-Plankton-Classification.

Nanni, L., Brahnam, S., Paci, M., Ghidoni, S., 2022a. Comparison of different convolutional neural network activation functions and methods for building ensembles for small to midsize medical data sets. Sensors 22 (16), http://dx.doi.org/10.3390/s22166129, URL https://www.mdpi.com/1424-8220/22/16/6129.

Nanni, L., Loreggia, A., Barcellona, L., Ghidoni, S., 2023. Building ensemble of deep networks: Convolutional networks and transformers. IEEE Access 11, 124962–124974. http://dx.doi.org/10.1109/ACCESS.2023.3330442.

Nanni, L., Manfè, A., Maguolo, G., Lumini, A., Brahnam, S., 2022b. High performing ensemble of convolutional neural networks for insect pest image detection. Ecol. Informatics 67, 101515.

Olson, R., Sosik, H., 2007. A submersible imaging-in-flow instrument to analyze nano- and microplankton: Imaging FlowCytobot. Limnol. Ocean. Methods 5, 195–203. http://dx.doi.org/10.4319/lom.2007.5.195.

Peng, B., Xu, Y., Huang, J., Hsiao, W., Xu, Z., Xie, Z., Xie, S., Urtasun, R., Lu, T., 2022. Beit v2: Masked image modeling with vector-quantized visual tokenizers. arXiv preprint arXiv:2208.06366.

Pitois, S.G., Blackwell, R.E., Close, H., Eftekhari, N., Giering, S.L.C., Masoudi, M., Payne, E., Ribeiro, J., Scott, J., 2025. RAPID: real-time automated plankton identification dashboard using edge AI at sea. Front. Mar. Sci. 11, http://dx.doi.org/10.3389/fmars.2024.1513463, URL https://www.frontiersin.org/journals/marine-science/articles/10.3389/fmars.2024.1513463.

Py, O., Hong, H., Zhongzhi, S., 2016. Plankton classification with deep convolutional neural networks. In: 2016 IEEE Information Technology, Networking, Electronic and Automation Control Conference. IEEE, pp. 132–136.

Ratnarajah, L., Abu-Alhaija, R., Atkinson, A., Batten, S., Bax, N.J., Bernard, K.S., Canonico, G., Cornils, A., Everett, J.D., Grigoratou, M., Ishak, N.H.A., Johns, D., Lombard, F., Muxagata, E., Ostle, C., Pitois, S., Richardson, A.J., Schmidt, K., Stemmann, L., Swadling, K.M., Yang, G., Yebra, L., 2023. Monitoring and modelling marine zooplankton in a changing climate. Nat. Commun. 14 (1), 564. http://dx.doi.org/10.1038/s41467-023-36241-5.

Roy, S.K., Paoletti, M.E., Haut, J.M., Dubey, S.R., Kar, P., Plaza, A., Chaudhuri, B.B., 2021. AngularGrad: A new optimization technique for angular convergence of convolutional neural networks. arXiv preprint arXiv:2105.10190.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C., 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4510–4520.

Sosik, H., Olson, R., 2007. Automated taxonomic classification of phytoplankton sampled with imaging-in-flow cytometry. Limnol. Ocean. Methods 5, 204–216. http://dx.doi.org/10.4319/lom.2007.5.204.

Tan, M., Le, Q., 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. In: International Conference on Machine Learning. PMLR, pp. 6105–6114.

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jégou, H., 2021. Training data-efficient image transformers & distillation through attention. In: International Conference on Machine Learning. PMLR, pp. 10347–10357.

Wang, C., Yu, Z., Zheng, H., Wang, N., Zheng, B., 2017. CGAN-plankton: Towards large-scale imbalanced class generation and fine-grained classification. In: 2017 IEEE International Conference on Image Processing. ICIP, IEEE, pp. 855–859.

Wightman, R., 2019. Pytorch image models. http://dx.doi.org/10.5281/zenodo.4414861, https://github.com/rwightman/pytorch-image-models.

Yuan, C., He, Z., Ning, C., Wang, W., Zhao, J., Yuan, G., Li, C., 2024. Research on in situ observation method of plankton based on convolutional neural network. J. Mar. Sci. Eng. 12 (10), http://dx.doi.org/10.3390/jmse12101702, URL https://www.mdpi.com/2077-1312/12/10/1702.

Zhang, Y., Shen, F., Li, R., Li, M., Li, Z., Chen, S., Sun, X., 2024. AIGD-PFT: The first AI-driven global daily gap-free 4 km phytoplankton functional type products from 1998 to 2023. Earth Syst. Sci. Data Discuss. 2024, 1–35. http://dx.doi.org/10.5194/essd-2024-122, URL https://essd.copernicus.org/preprints/essd-2024-122/.

Zhao, F., Lin, F., Seah, H., 2010. Binary SIPPER plankton image classification using random subspace. Neurocomputing 73, 1853–1860. http://dx.doi.org/10.1016/j.neucom.2009.12.033.

Zheng, H., Wang, R., Yu, Z., Wang, N., Gu, Z., Zheng, B., 2017. Automatic plankton image classification combining multiple view features via multiple kernel learning. BMC Bioinformatics 18, 1–18. http://dx.doi.org/10.1186/s12859-017-1954-8.