Automatic plankton quantification using deep features

Pablo González[a], Alberto Castaño[a], Emily E. Peacock[b], Jorge Díez[a], Juan José del Coz[a], Heidi M. Sosik[b]

[a]*Artificial Intelligence Center, University of Oviedo at Gijón, Spain*

[b]*Biology Department, Woods Hole Oceanographic Institution, Woods Hole, MA 02543*

Abstract

The study of marine plankton data is vital to monitor the health of the world's oceans. In recent decades, automatic plankton recognition systems have proved useful to address the vast amount of data collected by specially engineered in situ digital imaging systems. At the beginning, these systems were developed and put into operation using traditional automatic classification techniques, which were fed with hand-designed local image descriptors (such as Fourier features), obtaining quite successful results. In the past few years, there have been many advances in the computer vision community with the rebirth of neural networks. In this paper, we leverage how descriptors computed using Convolutional Neural Networks (CNNs) trained with out-of-domain data are useful to replace hand-designed descriptors in the task of estimating the prevalence of each plankton class in a water sample. To achieve this goal, we have designed a broad set of experiments that show how effective these deep features are when working in combination with state-of-the-art quantification algorithms.

Keywords: Abundance estimation, quantification, deep learning, convolutional neural networks, phytoplankton

1. Introduction

Phytoplankton play a vital role in marine ecosystems. Since the creation of automatic plankton imaging systems, many efforts have been devoted to the development of automatic techniques for processing all the data captured to optimize conclusions from temporally dense data sets (Benfield et al., 2007).

In the last few years, attention has turned to Convolutional Neural Networks (CNNs) that push the limit of computer vision techniques, but before that, systems designed to automatically classify plankton were trained with hand-designed descriptors. These descriptors were a reduced representation of each image, that were used for training and testing machine learning algorithms such as Random Forest or

Email addresses: gonzalezgpablo@uniovi.es (Pablo González), castanoalberto@uniovi.es (Alberto Castaño), epeacock@whoi.edu (Emily E. Peacock), jdiez@uniovi.es (Jorge Díez), juanjo@uniovi.es (Juan José del Coz), hsosik@whoi.edu (Heidi M. Sosik)

Support Vector Machines (SVM). These well studied descriptors included shape and texture features, such as Fourier descriptors (Kuhl and Giardina, 1982), Haralick features (Haralick and Shanmugam, 1973), invariant moments (Hu, 1962), etc. When using CNNs, these descriptors no longer need to be computed for each image. Convolutional layers in the CNN serve as feature extractors, gaining in complexity as we move forward into the network, while pooling layers are designed to reduce the spatial resolution of the feature maps, obtaining then invariance to translations and distortions. The network itself learns a representation of the images when adjusting the weights of its different layers. This approach has been shown to be superior to hand-designed descriptors (Sharif Razavian et al., 2014) and it was applied by most of the teams participating in the National Data Science Bowl (NDSB) competition[1], where participants had to classify plankton images, and 81.5% accuracy was reached across 121 different categories.

While CNNs can be used to build a classifier for a specific dataset, as the teams of the NDSB competition did, they can also be used to extract a numerical representation of a given image (as an alternative to hand-designed descriptors). After feeding a CNN with a plankton image and evaluating all the activations of the network, activations in fully connected layers are compressed representations of the image and can be used as image descriptors. These descriptors, called deep features, have been applied successfully to many computer vision problems (Oquab et al., 2014; Chatfield et al., 2014).

One of the main problems with CNNs is that they are computationally expensive. They need special hardware to be trained (powerful GPUs) and the time needed for training a big CNN with a respectable amount of data is usually counted in weeks. One possible solution for this issue is to use a CNN already trained with a set of images belonging to a different domain, a technique known as transfer learning (Pan and Yang, 2010). With this approach, a pre-trained CNN can be used to compute deep features of plankton images. To improve the results, the network can be fine-tuned with labeled plankton images, so the network weights are adjusted better to the plankton domain.

Transfer learning is a technique that has been around for a few years and that has increased in importance since the growth in popularity of CNNs. It has been applied to the WHOI-plankton dataset (Sosik et al., 2015) with promising results in terms of classification accuracy (Orenstein et al., 2015). Recently, increasingly more powerful CNNs have been developed with larger numbers of layers (He et al., 2016), leading to astonishing results over the ImageNet dataset (Deng et al., 2009). These very deep pre-trained networks are usually openly available, presenting us with the opportunity to test their performance in challenging problems such as the WHOI-plankton dataset where the objective is to estimate the prevalence of plankton taxa in a water sample.

The task of predicting the prevalence of each taxon in a given sample has often been tackled with image classification techniques. The most basic approach uses a classifier to assign a class to each plankton

[1] https://www.kaggle.com/c/datasciencebowl

image and then counts them. We shall call this approach "Classify & Count". Although this method has some efficacy, it is suboptimal and can be improved with methods specifically designed for quantification (González et al., 2017c), such that the aggregated underlying distribution is considered rather than individual classifications.

We are interested in estimating the prevalence of each class in an unknown water sample. To that end, we have used quantification algorithms with deep features as their input, and we have analyzed their performance with a rigorously designed validation methodology (González et al., 2017a) where the sample is the minimum test unit.

In recent years, different deep learning algorithms have been applied for plankton classification, see Moniruzzaman et al. (2017) for a small survey of some of these applications. However, to the best of our knowledge, only one other paper (Beijbom et al., 2015) has studied the use of deep learning for plankton abundance estimation using quantification algorithms before. In the majority of the papers published on the topic, authors use CNNs as classifiers rather than as quantifiers. For instance, Py et al. (2016) and Luo et al. (2018) describe two systems based on CNNs able to automatically classify 121 and 108 types of plankton, respectively. Dunker et al. (2018) and Lloret et al. (2018) use CNN classifiers for identifying phytoplankton species. Dai et al. (2016a) present a similar approach in the design of a zooplankton classifier. Other authors combine CNNs with different machine learning techniques, including active learning (Bochinski et al., 2018), hybrid systems (Dai et al., 2016b), parallel networks (Wang et al., 2018), imbalance learning (Lee et al., 2016) or different forms of information fusion (Cui et al., 2018; Lumini and Nanni, 2019). It should be noted that the improvements on plankton classification described in these papers may not be directly transferable to quantification systems, as classification and quantification are two different tasks. Importantly, capturing the changes in the distribution between training data and test samples (González et al., 2017b) is crucial when dealing with quantification. All proper quantification algorithms have some mechanism to detect and deal with such changes (see Section 2.4). Furthermore, classification and quantification use different target performance measures. While classification requires performance metrics that measure classification accuracy at the individual image level (e.g., how likely it is that an image of a given taxon will be classified correctly), quantification focuses on sample-level errors instead (e.g., how precise is the estimated concentration of a given taxon). The correlation between both performances is lower than expected, see González et al. (2017a) for further details. Thus, plankton quantification should be properly studied through a well-designed set of experiments, different from those commonly used in plankton classification papers.

Beijbom et al. (2015) apply four quantification algorithms (Forman, 2008; Saerens et al., 2002) based on CNNs classifiers to automatically estimate the abundance of 33 classes over 21 test samples. The present paper expands such study in several directions:

1. Applying standard CNNs, with and without fine-tuning, as feature extractors. The goal is to analyze whether fine-tuning helps to significantly improve quantification performance.

2. Employing CNNs to obtain deep features, rather than as classifiers. Since training CNN classifiers may be complex for some users, this paper proposes the use of deep features provided by already trained CNNs in combination with easy-to-train quantification algorithms.
3. Comparing deep features with hand-crafted features (e.g. shape and texture features) that were the standard until the emergence of deep learning algorithms.
4. Performing more exhaustive experiments. In Beijbom et al. (2015) only 21 test samples were used, a number that we consider to be too low for analyzing quantification performance, and those classes with less than 1000 examples were removed. Our study comprises 764 test samples considering all the 49 classes present in those samples. Additionally, the computational cost of the compared approaches is also analyzed.

Our aim is to show that an approach that combines standard CNNs with basic quantification algorithms outperforms traditional machine learning methods over the WHOI-plankton dataset.

For the sake of reproducibility, all relevant source code used to run experiments has been made available for download[2], along with the full results drawn from all the experiments that were not included in this paper[3].

2. Material and Methods

2.1. Dataset

The WHOI-Plankton dataset (Sosik et al., 2015) was used for all of the experiments. This dataset is publicly available and it has been used by a few papers on this topic (Beijbom et al., 2015; Lee et al., 2016; Orenstein and Beijbom, 2017). The WHOI-Plankton data was collected with a multi-year series of Imaging FlowCytobot (IFCB) (Olson and Sosik, 2007) deployments at the Martha's Vineyard Coastal Observatory (MVCO), which is a facility operated by Woods Hole Oceanographic Institution (WHOI). The MVCO site is a component of the Northeast U.S. Shelf Long-Term Ecological Research (NES-LTER) program where the IFCB time series contributes critical information to characterize and understand the roles of plankton in ecosystem function. At MVCO, IFCB automatically draws in a 5-ml sample of seawater every 20 minutes. The seawater sample is pumped through a cytometric system, where particles that contain chlorophyll and are in the approximate size range 10 to 150 μm are imaged. Regions of interest (ROIs) containing plankton targets are extracted from the camera frame in realtime during a sample run. These ROIs are stored onboard the IFCB and transmitted to shore over an Ethernet connection. At MVCO, IFCB has captured nearly 1 billion images since 2006.

[2] https://github.com/pglez82/IFCB_quantification
[3] https://pglez82.github.io/IFCB_quantification

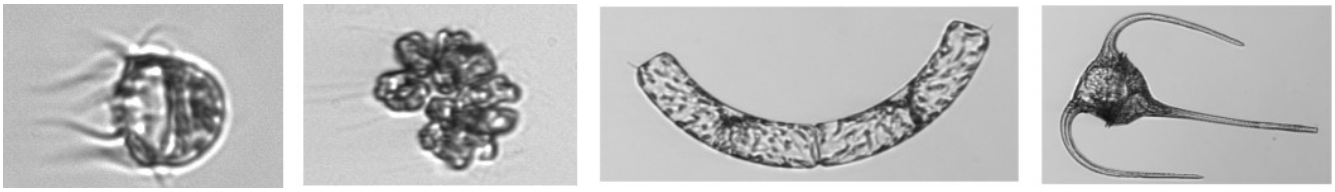


Figure 1: Examples of ROIs from the WHOI annotated dataset.

From this huge quantity of images, NES-LTER researchers at WHOI have annotated 3.5 million ROIs belonging to more than 5000 different samples. They have manually sorted images into 103 different classes, which can be grouped together into 51 more generic categories. Notably, in these experiments we have used these image categories as determined by NES-LTER researchers working with the image data to address unresolved ecological questions. An example of IFCB images from MVCO can be seen in Figure 1. The size of the images varies depending on the size of the organism, typically ranging from 100 to 100,000 pixels. All images are gray-scale.

The WHOI-Plankton dataset has a highly unbalanced distribution in which over 90% of images belong to only 5 classes and the most prevalent class (miscellaneous nanoplankton) represents 75% of all ROIs in the dataset.

It is important to highlight that every sample has a different plankton distribution, due to temporal variations in the natural community.

2.2. Data preprocessing

The WHOI-plankton dataset images are distributed in more than 5000 samples. From all these samples only the fully annotated ones (all the individuals in the sample have been annotated) were considered for the experiment in order to properly test quantification methods. The resulting dataset contains 3.4 million images organized in 964 samples collected between 2006 and 2014. The categories considered were the ones suggested in Sosik et al. (2015), which comprises 51 different classes. From these 964 samples, the dataset was split into training and test sets taking the first 200 samples (in temporal sequence) as the training set and the rest as the test set. Because two classes contained no examples in the first 200 samples, this split resulted in a 49-class training set and resulting 49-class quantifiers evaluated in this work.

This experimental setting follows the guidelines suggested in (González et al., 2017a) trying to simulate a realistic case in which: 1) training data is collected during a sufficiently long period of time, 2) a model is learned using these training data and 3) finally such model is deployed to automatically process subsequent samples. Notice that only 20% of all available data has been chosen for the training set but these first 200 samples represent all data collected from 2006 to 2008, a length of time which should assure the classes of interest are represented. It is likely that using a larger number of samples in the training set could result in improved performance for the whole system, but the trade-off is increased

time required to learn each model. Our choice balances adequate performance with training time that is fast enough to run all the experiments reported in this paper (see Table 4).

All plankton images have been resized to match the inputs of the CNNs. In this case, images had to be 224x224 pixels. As most of the IFCB images were not square, each one was resized keeping its original aspect ratio and so that its longest dimension is 224 pixels; then the resulting scaled ROI was placed in the middle of the image and the two lateral gaps were filled with the value of the average pixel, computed from 30.000 plankton images randomly selected from the data set.

Hand-coded features were downloaded from the publicly available MVCO IFCB Dashboard website[4], where standard computations are provided for the entire dataset (Sosik and Olson, 2007; Sosik et al., 2016). For each image a vector with 227 features was downloaded, including shape and texture features. The computation process for these features is carefully documented in Sosik (2017). From now on, we will refer to this feature set as *normal features* (NF) for which performance will be compared against the features computed using Convolutional neural networks (CNNs), also known as *deep features*.

2.3. Deep features

Convolutional networks (CNNs) have recently enjoyed great success in large-scale image recognition tasks. This has been made possible by the existence of large public image repositories, such as ImageNet (Deng et al., 2009), and the increase of computing capacity. CNNs used by the computer vision community have been growing deeper and deeper since AlexNet (Krizhevsky et al., 2012) was proposed in 2012 with only 8 layers. Nowadays, deep residual networks (resnets) (He et al., 2016) contain more than one hundred layers. The resnet architecture solves the notorious vanishing gradient problem (Hochreiter et al., 2001) that emerges when training very deep networks by introducing short cut connections that skip one or more layers. Notably, residual networks were successful in winning the ImageNet ILSVRC 2015 competition with an incredible error rate of 3.6% (humans generally hover around a 5-10% error rate).

When a CNN is trained on images, like those in ImageNet, to perform image classification, it automatically learns features that will vary in complexity depending on the layer depth. On the first layers, features similar to Gabor filters that act as edge and contour detectors are learned. These features are not specific to a particular dataset. Deeper layers in the network learn more complex features, usually from a combination of features from early layers, that resemble shapes or forms, that are also more specific to the dataset in hand. Nonetheless, this specificity is not a problem since the ImageNet dataset is sufficiently varied. When a new image (in our case, a plankton image) is presented to the CNN, this set of

[4] http://ifcb-data.whoi.edu/mvco

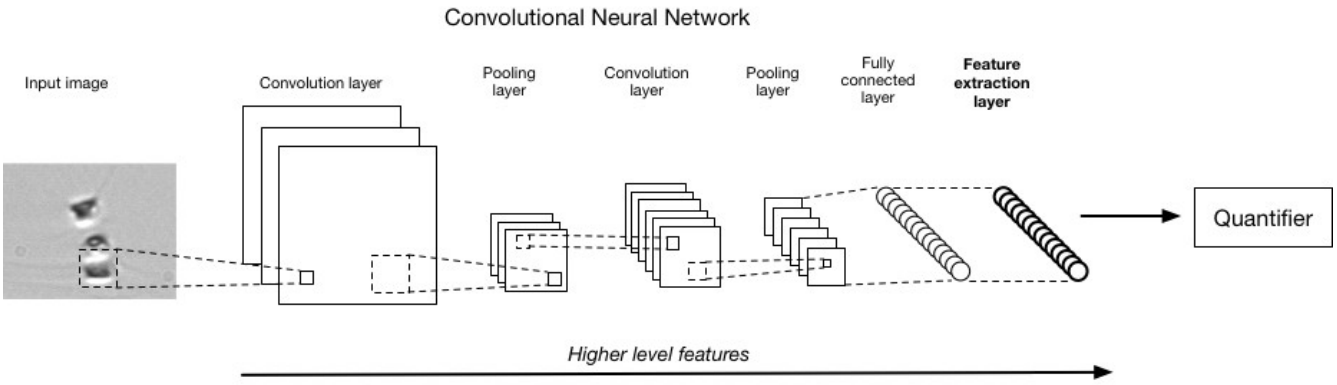


Figure 2: CNN architecture used for deep feature extraction. Layers on the right contain higher level features. The network input is a raw image resized to fit the input layer.

learned features will be computed in each of the network layers. These deep features, can then be used as the input for the different quantification algorithms.

For the problem at hand, we have used resnets pre-trained on the ImageNet dataset, even though other network architectures could be considered (see for example, Huang et al. 2016). ImageNet contains images totally different from plankton images containing only macroscopic images such as animals, landscapes, etc. We tested different pre-trained versions of this network, varying the number of layers from 18 (RN-18) to 101 (RN-101). Our goal was to determine the degree of complexity necessary to obtain satisfactory results. To compute deep features for each plankton image, activations were pulled from the final fully connected layer of a network during a forward pass of each IFCB plankton image (see Figure 2). The number of deep features obtained for each image depends on the network used, varying between 512 for RN-18 and RN-34 to 2048 for RN-50 and RN-101.

Even though off-the-shelf CNN deep features have good discriminative power (Sharif Razavian et al., 2014), results can be improved by fine-tuning the networks to actual plankton images. To fine-tune the CNNs and adapt them to plankton images, we replaced the last fully connected layer of the CNNs (which is designed for classifying ImageNet) with an output layer matching the number of classes in our dataset. The network was then trained with the labeled images from the training set in order to adapt its weights to plankton images. In our experiments, we used 30 epochs, being an epoch a full pass of the whole training dataset (half a million images), with a learning rate of 0.01, which was decreased by an order of magnitude after completing the first 15 epochs.

All fine-tuning and deep feature computing was done with the R deep learning package MXNet (Chen et al., 2015) on 2xNVIDIA K80 GPUs. Times needed for fine-tuning the networks and computing the deep features are shown in Table 4.

2.4. *Quantification algorithms*

In the wide variety of problems that machine learning faces, there are tasks in which the individual class predictions are not as important as predicting the proportion of each class in a concrete sample or set of examples. This problem is called quantification (Forman, 2008) in machine learning and data mining communities. Quantification is a learning problem on its own because it requires specific approaches, and not just using classification methods. In fact, many experiments (Barranquero et al., 2013, 2015; González et al., 2017c) have shown that off-the-shelf classifiers are often suboptimal when applied directly to quantification tasks. For that reason, several quantification algorithms have been proposed during the past few years (Firat, 2016; Narasimhan et al., 2016; Pérez-Gállego et al., 2017, 2019). A review of quantification learning can be found in González et al. (2017b).

Given a dataset $D = \{(x_1, y_1), ..., (x_n, y_n)\}$, in which x_i is a representation of an individual example in the input space $\mathcal{X}$ and $y_i \in \mathcal{Y} = \{c_1, ..., c_l\}$ is the corresponding class label, the goal in supervised classification is learning a model:

$$h : \mathcal{X} \longrightarrow \{c_1, ..., c_l\}, \tag{1}$$

able to assign a class label for a new unseen example. In quantification, the learning task is totally different from a formal point of view; it can be defined as follows:

$$\bar{h} : Sample \longrightarrow [0, 1]^l. \tag{2}$$

In this case the model $\bar{h}$ returns a l-dimensional vector in which each element, $\hat{p}_j$, represents the predicted prevalence for class j for the input sample, such that

$$\begin{aligned} &\sum_{j=1}^{l} \hat{p}_j = 1, \\ s.t. \quad &0 \leq \hat{p}_j \leq 1, \forall j = 1, \ldots, l. \end{aligned} \tag{3}$$

That is, $\bar{h}$ predicts the class probability distribution of a sample. Despite the fact that most quantifiers have been designed for binary problems ($l = 2$), multiclass quantification ($l > 2$) can be solved combining the results of l binary quantifiers. In this paper, we use the well-known one-vs-all approach that learns a collection of binary quantifiers:

$$\bar{h}_j : Sample \longrightarrow [0, 1]. \tag{4}$$

Each $\bar{h}_j$ just returns the proportion of examples of class j in the sample. The initial predictions of all the binary models, $\{ \hat{p}_j^0 \mid j = 1, \ldots, l\}$, are finally normalized in order to satisfy (3):

$$\hat{p}_j = \frac{\hat{p}_j^0}{\sum_{j=1}^{l} \hat{p}_j^0}. \tag{5}$$

In this study, we have evaluated a set of quantification algorithms developed in the literature for application to binary quantifiers. These approaches were mainly selected because their implementation is very

simple and any practitioner with a little knowledge of machine learning could implement these algorithms. We briefly describe each of them here.

The Classify & Count (CC) approach follows the most intuitive way to tackle a quantification problem: build a classifier and count the examples falling into each class. We shall consider this method as a baseline as it is not formally a quantification method, even though it is used in the evaluation of many automatic plankton recognition systems (González et al., 2017a). The problem with the CC method is that its performance degrades when there are significant changes in class distributions (González et al., 2017b).

The Adjusted Count (AC) algorithm, proposed by Forman (Forman, 2008), is theoretically well founded and based on making a correction to the prevalence estimated by the CC method, $\hat{p}_j^{CC}$, using the classifier true positive rate (tpr) and false positive rate (fpr) for the target class j:

$$\hat{p}_j^{AC} = \frac{\hat{p}_j^{CC} - fpr}{tpr - fpr} \tag{6}$$

which would lead to a perfect prediction given that the estimation of tpr and fpr is perfect and $P(x|y)$ is constant [see further details in Forman (2008)]. Even when these two conditions are not completely fulfilled, AC usually works better than CC (see Section 3). This approach has been previously used in the plankton domain for correcting abundance estimates with a high degree of success (Sosik and Olson, 2007).

The methods referred to as Probabilistic Classify & Count (PCC) and Probabilistic Adjusted Count (PAC) (Bella et al., 2010) work with an underlying probabilistic classifier instead of a crisp one. The prevalence is then computed as the average of the probability of belonging to class j for all the examples in a test sample T:

$$\hat{p}_j^{PCC} = \frac{1}{|T|} \sum_{x \in T} P(y = c_j|x) \tag{7}$$

In the PAC method, this result is adjusted in an analogous way as in the AC method.

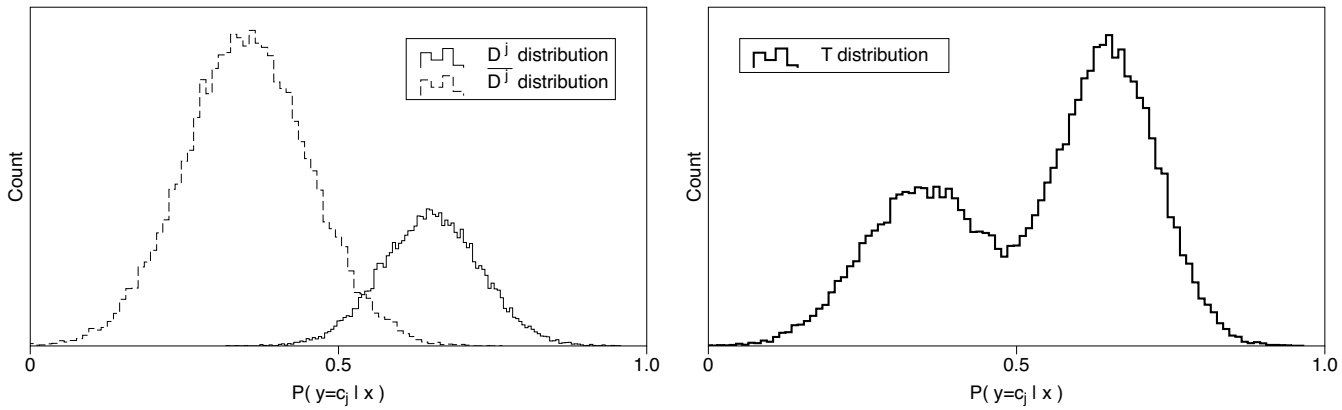
$$\hat{p}_j^{PAC} = \frac{\hat{p}_j^{PCC} - FP^{pa}}{TP^{pa} - FP^{pa}} \tag{8}$$

in which TP^{pa} (*TP probability average*) and FP^{pa} (*FP probability average*), are estimated from the training dataset, and are defined as:

$$TP^{pa} = \frac{\sum_{x \in D^j} P(y = c_j|x)}{|D^j|} \quad \text{and} \quad FP^{pa} = \frac{\sum_{x \in \overline{D^j}} P(y = c_j|x)}{|\overline{D^j}|} \tag{9}$$

where D^j is the set of training examples in class j and $\overline{D^j}$ is the rest of the training examples in D.

The HDy method (González-Castro et al., 2013) is based on matching probability distributions where the Hellinger Distance (HD) is the metric to compute the difference between such distributions. It uses



(a) D^j and $\overline{D^j}$ distributions

(b) Test sample distribution

Figure 3: HDy computes first the probability density functions of (a) the examples from D^j and $\overline{D^j}$. Eq.(10) combines both distributions for different values of $\hat{p}$, to approximate (b) the distribution of the test set sample.

the outputs of a binary classifier to represent the distributions for D^j, $\overline{D^j}$ and the test set T (with binning to approximate the integral in the definition of HD), see Figure 3. The idea is to combine the distributions D^j and $\overline{D^j}$, by means of their prevalences, to approximate the observed distribution in T:

$$\hat{p}_j^{HDy} = \min_{\hat{p}_j \in [0,1]} \sqrt{\sum_{k=1}^{bins} \left(\sqrt{\frac{|T_k|}{|T|}} - \sqrt{\frac{|D_k^j|}{|D^j|} \cdot \hat{p}_j + \frac{|\overline{D^j}_k|}{|\overline{D^j}|} \cdot (1 - \hat{p}_j)} \right)^2} . \quad (10)$$

For instance, in the example depicted in Figure 3, the prevalence of class j is 0.4 in the training set (a) and 0.6 in T (b). To match the distribution of T, we need to increase $\hat{p}_j$ to give more importance to D^j distribution. A simple linear search in which $\hat{p}_j$ moves over the range [0,1] in small steps is used to select the predicted prevalence for class j that minimizes the HD.

These quantification algorithms have been implemented in Python and are publicly available as a Python module called PyQuan[5]. PyQuan is able to tackle multi-class quantification problems with n binary quantifiers using a one-vs-all approach. As explained above, a binary quantifier is trained for each class j considering examples of this class as the positive class and the rest as the negative. We trained only one model per class and used it for each of the quantification algorithms described above (CC, AC, PCC, PAC and HDy). This guarantees that the differences in performance between them are only due to the way in which each method employs the predictions made by the binary classifiers. We use linear regression as the underlying binary classifier as it is simple and fast enough to be trained with this dataset in reasonable time and it provides probabilistic outputs, which are needed for the PCC and PAC methods. The regularization parameter C was adjusted for each model with a grid search over the values $(0.1, 1, 10)$.

To run the quantification algorithms, we used a machine with 2 Haswell 2680v3 processors, 24 cores,

[5] https://github.com/albertorepo/quantification

and 120Gb of RAM. All experiment times are shown in Table 4.

2.5. Performance measures

To compare the different methods and descriptors used in this paper, we need to define the evaluation method and performance measures. Given the characteristics of this dataset and its inherent properties (see Section 2.1), we chose a validation method that ensures that results are transferable to production like conditions. That is, testing should be carried out with different samples presenting different plankton distributions, covering the actual variations due to seasonalilty or any other factors. The dataset comprises 964 samples, where the first 200 (ordered from oldest to newest) are used for training, having 764 remaining samples for the test set. In this work, the evaluation guidelines proposed in González et al. (2017a) have been followed. Thus, given the high number of samples, the evaluation method chosen has been a hold-out by sample, where the model to evaluate is used to predict the distribution of all the samples in the test set. An important precondition for properly evaluating quantification algorithms is that samples must be complete, meaning that all examples present in a sample have to be annotated and placed in one class and no example can be manually discarded.

Performance measures included in this paper are Mean Absolute Error (MAE) and Mean Relative Absolute Error (MRAE). Given the true prevalences $\{p_{j,s} : s = 1, \ldots, m\}$ of class c_j over m labelled samples, $\{T_1, \ldots, T_m\}$, and the predicted prevalences $\{\hat{p}_{j,s} : s = 1, \ldots, m\}$, these performance measures can be defined as:

- Mean Absolute Error: $MAE(c_j) = \frac{1}{m} \sum_{s=1}^{m} |p_{j,s} - \hat{p}_{j,s}|$
- Mean Relative Absolute Error: $MRAE(c_j) = \frac{1}{m} \sum_{s=1}^{m} \frac{\epsilon + |p_{j,s} - \hat{p}_{j,s}|}{\epsilon + p_{j,s}}$, where ϵ is a small constant that prevents the function from being undefined when $p_{j,s} = 0$.

We do not compare classification accuracy for individual images for several reasons: 1) our goal is to tackle the abundance problem in which predictions at the individual level are not relevant, 2) in fact, some of the compared algorithms (e.g. AC, PAC and HDy) do not provide such individual classifications, and 3) it has been shown that the correlation between classification accuracy and quantification accuracy is much lower than expected; see González et al. (2017a) for a complete analysis on this issue.

3. Results

All experiment results are fully available online[6] as an interactive web application, allowing the user to compare between different feature sets and quantification methods, for each class in the dataset.

[6] https://pglez82.github.io/IFCB_quantification/

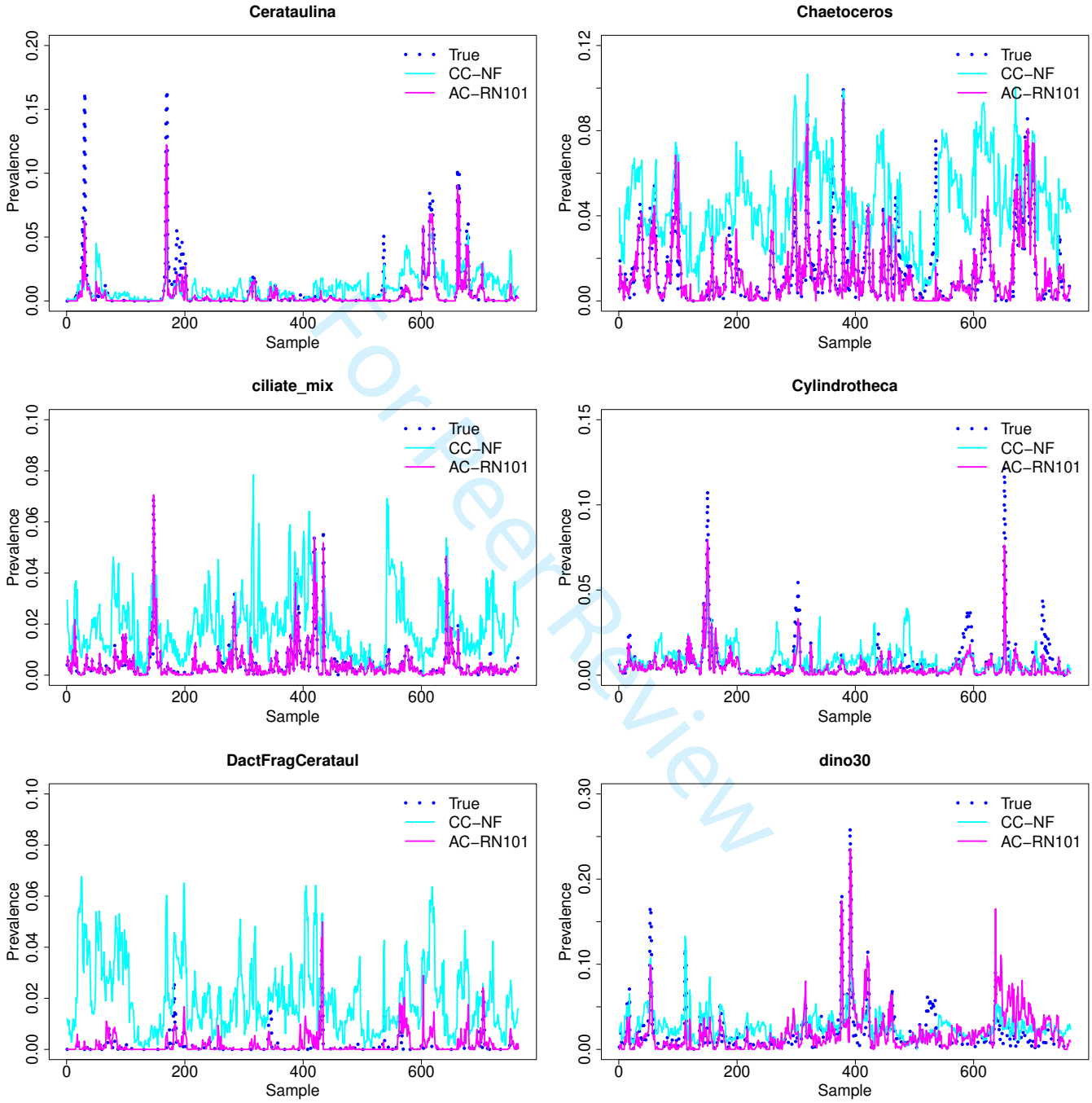


Figure 4: (Part 1) Results for each sample comparing true prevalence and model outputs using NF (normal features) with CC method and RN (fine-tuned RN-101 features) using AC method.

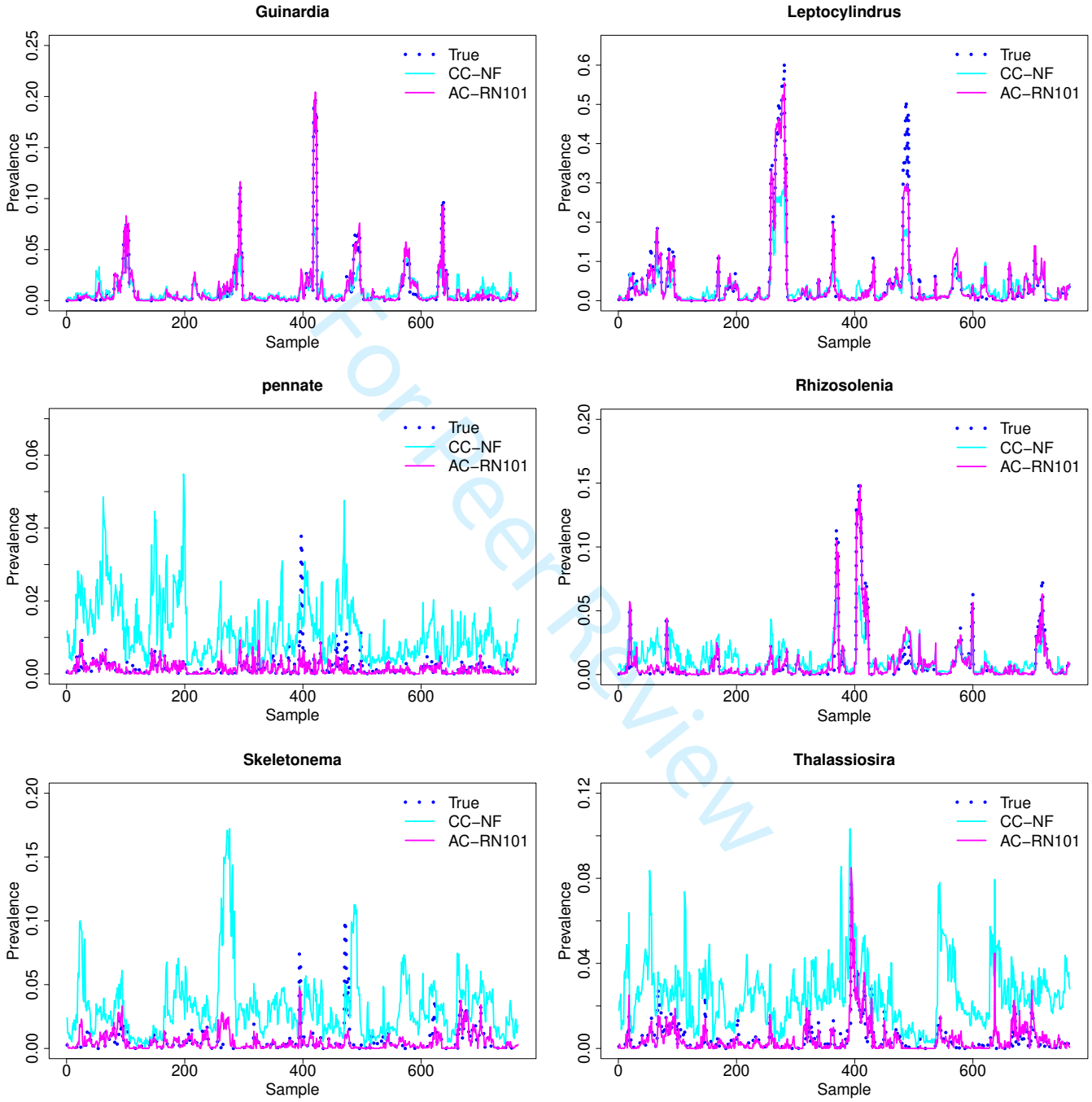


Figure 5: (Part 2) Results for each sample comparing True prevalence and model outputs using NF (normal features) with CC method and RN (fine-tuned RN-101 features) using AC method.

Class	NF CC (10^{-4})	NF AC (10^{-4})	RN-101 CC (10^{-4})	RN-101 AC (10^{-4})
Cerataulina	89.25 / 4.39	61.97 / 4.61	28.44 / 2.87	**27.52** / 2.79
Chaetoceros	334.28 / 6.17	136.26 / 5.47	42.49 / 1.86	**38.81** / 1.97
ciliate_mix	148.76 / 3.96	71.13 / 4.29	6.56 / 0.40	**6.47** / 0.36
Cylindrotheca	58.14 / 2.82	63.26 / 3.00	23.18 / 1.98	**22.50** / 1.93
DactFragCerataul	192.36 / 4.97	75.18 / 5.29	20.19 / 1.06	**14.69** / 1.04
dino30	148.90 / 4.82	128.98 / 4.78	138.26 / 5.05	**121.86** / 5.20
Guinardia	62.17 / 4.42	50.89 / 3.97	20.88 / 1.27	**20.48** / 1.28
Leptocylindrus	217.77 / 16.64	184.05 / 12.26	91.86 / 7.48	**87.28** / 7.17
pennate	101.97 / 2.83	30.42 / 2.28	**8.05** / 0.79	8.35 / 0.79
Rhizosolenia	82.21 / 3.95	61.50 / 3.55	20.99 / 1.41	**20.00** / 1.39
Skeletonema	272.94 / 9.43	229.31 / 13.01	**28.52** / 2.65	28.57 / 2.66
Thalassiosira	210.93 / 4.94	75.49 / 5.07	19.93 / 0.98	**19.32** / 0.99

Table 1: Absolute Errors (AE) Mean / Standard Error for 12 classes over all test samples (full table can be found in the supplemental material). Results for CC and AC methods using normal features (NF) and RN-101 features. Lowest errors per class shown in bold.

The annotated dataset described in Section 2.1 was used for the experiments. Hand-coded descriptors (”normal” features, NF) downloaded from the MVCO IFCB Dashboard (see Section 2.2) were used as a baseline to compare with results from descriptors obtained with CNNs. The CNN-derived descriptors were computed with residual deep networks (see Section 2.3). These deep features were used for training and testing the quantification algorithms in the same way as the hand-coded descriptors. All experiment times have been logged with the aim of giving a general view of the computing time needed to apply these methods (see Table 4). With the class prevalences calculated for each quantification method, Absolute Errors and Relative Absolute Errors were computed (see Table 1).

We found that deep features perform better than normal features resulting in a lower absolute error for all classes. This difference is illustrated in Figures 4 and 5. For a given class, it is important to observe how the true prevalence varies from sample to sample, sometimes very abruptly. Predictions made with deep features get a superior level of adjustment compared to traditional features. There are some classes like mix_elongated or ciliate_mix where predicted prevalences from deep features are almost perfect. It is interesting to note how true prevalences vary over samples. For instance in the class *Leptocylindrus*, true prevalence goes from less than 0.1 to 0.61 in sample 280. Similar changes can be observed for most classes. These variations make this problem challenging and suitable for quantification techniques.

Differences between the CC and AC method are very small when we deal with very low absolute errors. For RN-101 features, AC is almost equivalent to CC. The mean absolute error by class for CC is 0.0035 where the same value for AC is 0.0031. This difference is greater when features do not work as well. For instance, with normal features, error decreases from 0.0149 with CC to 0.0.0084 with AC, a 43% decrease in absolute error. On the one hand, when the CC method already gets very good results, the margin for improvement is too low to be noticeable. On the other hand, when dealing with a complex

	NF	RN-18*	RN-18	RN-34*	RN-34	RN-50	RN-101
CC	0.0149	0.0111	0.0103	0.0110	0.0070	0.0036	0.0035
AC	0.0084	0.0208	0.0074	0.0211	0.0268	0.0031	0.0031
PCC	0.0171	0.0133	0.0118	0.0130	0.0082	0.0045	0.0044
PAC	0.0089	0.0077	0.0082	0.0072	0.0058	0.0035	0.0034
HDy	0.0075	0.0063	0.0071	0.0057	0.0055	0.0054	0.0062

Table 2: Mean Absolute Error (AE) by class for all the CNN tested. CNNs with * have not been fine-tuned to plankton images.

	NF	RN-18*	RN-18	RN-34*	RN-34	RN-50	RN-101
CC	17.64	9.15	9.69	8.53	4.45	2.13	2.08
AC	9.99	67.03	7.87	69.85	94.16	1.88	1.87
PCC	20.34	11.96	12.18	11.11	6.67	3.71	3.37
PAC	10.24	7.07	8.95	5.95	4.09	2.50	2.41
HDy	7.9	5.71	8.19	5.16	4.67	11.31	13.97

Table 3: Mean Relative Absolute Error (MRAE) by class for all the CNN tested. CNNs with * have not been fine-tuned to plankton images.

quantification problem like this one, the conditions for a perfect adjustment are only met to a certain degree (tpr and fpr estimations are not perfect and $P(x|y)$ varies across the dataset).

For other quantification methods, it is interesting to see that adjustments usually work better than the CC method. For instance, AC improves the results in four out of seven experiments. Taking a closer look at experiments where AC has underperformed CC (RN-18*, RN-34* and RN-34), the problem is caused by a few classes (such as *Stephanopyxis*) with very few training examples and nearly zero prevalence over all samples. With such a low number of examples for a class, it is possible for AC to compute a tpr and fpr almost zero. In this case, it is easy to see how a very small denominator in Equation 6 can lead to a high error in the adjustment. Since Table 2 and Table 3 errors are averaged by class, giving the same importance to every class in the dataset, the errors can be misleading without considering each class individually.

Similar conclusions can be drawn looking at the Mean Relative Absolute Errors (see Table 3). The same problem is observed with class *Stephanopyxis*, where the relative error is very high for the AC method in three experiments. In the rest of the experiments, values are equivalent to those in the Absolute Error table and show how well RN-50 and RN-101 with AC work, with an error lower than 2%.

Another interesting conclusion is that PAC outperforms PCC in all seven experiments (both in absolute and relative errors). The adjustment in PAC works similarly to AC (see Equation 8), but seems more resistant than AC to the problems due to very infrequent classes, mainly because PAC does not use a threshold when deciding if an example belongs or not to a certain class, as this method works with the

	NF	RN-18	RN-34	RN-50	RN-101
Fine-tuning CNN	X	11	17	36	65
Compute deep features	X	27	30	32	45
Quantification	72	96	96	224	226

Table 4: Comparative of times (in hours) needed for making the experiments using 2 GPUs NVIDIA Tesla K80 for GPU tasks (fine-tuning and compute deep features) and 2 processors Haswell 2680v3, 24 cores with 120Gb RAM memory for CPU tasks (quantification).

raw probabilities returned by the classifier. Also, HDy appears as a very solid method, giving very good results even with normal features.

Mean absolute errors by class (see Table 2) show errors for shallower residual networks. It is important to note than even the smaller network with 18 layers and without fine-tuning (RN-18*) outperforms normal features. This result leads to the conclusion that the process of fine-tuning is desirable but not required. Previous studies (Sharif Razavian et al., 2014) have shown how off-the-shelf deep features work better than hand-coded features and this claim is confirmed by our work. Nonetheless, it is important to note that fine-tuning is a process that is done only once in the model building phase and it is not very computationally expensive (65 hours of GPU computing for the biggest network tested: RN-101). Fine-tuning leads to an improvement over 7% in absolute error for RN-18 and a 36% improvement for RN-34.

It is important to notice how errors decrease with deeper networks. The largest difference takes place from 34 layers to 50 layers, where absolute error for the CC method decreases from 0.0070 to 0.0036 (49% improvement). From there, even doubling the number of network layers (from 50 to 101) only results in a 2% decrease in absolute error. This improvement also has a drawback in computation time. In addition, the number of deep features computed with RN-50 is four times higher than for RN-34 (2048 vs. 512). Table 4 shows how time increases from 96 hours to more than 200 hours for building the model and applying quantification algorithms. Part of this time increment has its origin in the memory requirements to fit a dataset four times bigger. With a machine with 120Gb of RAM, we were able to build up to twelve binary models at the same time for 512 deep features, but only four parallel models for 2048 deep features.

4. Discussion

We have evaluated how well deep features perform when trying to estimate the abundances of plankton species in a water sample. Conforming with current computer vision literature, deep features have proven far more powerful than traditional hand-designed descriptors. Even the smallest networks, pre-trained with out-of-domain data, are able to compete against traditional features.

Deep features are applicable to most problems in the computer vision field. Nowadays they should be considered above hand-designed descriptors given their robust performance, as shown by this and many other recent studies. Even if a dataset is not big, and fine-tuning is not an option, pre-trained CNNs should still remain as a viable alternative.

The power of the improved quantification accuracy achieved with deep features is most evident when we consider the implications for understanding important ecological problems. A major objective of IFCB deployments at MVCO is to characterize taxon-specific bloom occurrences and temporal changes in community structure in the plankton. The number of images in the multi-year dataset (nearly 1 billion) makes full expert validation of image classifications prohibitive. Traditional hand-descriptor based classification has been employed with some success, but even low overall false positive rates can interfere with the ability to separate critical species and provide adequate estimates of bloom trajectories through time.

We highlight the relative strengths of quantification with deep features with several contrasting plankton taxa in the MVCO records, fully analyzed to reflect absolute concentration in the environment (count estimate scaled to seawater volume) and known sample date and time (Figure 6). The chain-forming diatom species *Guinardia delicatula* commonly dominates the phytoplankton biomass at MVCO, with large wintertime blooms occurring in many years. While traditional features and a random forest classification approach have previously been used to study bloom dynamics in this species (Peacock et al., 2014), quantification from AC coupled with RN-101 provides fewer cases of incorrectly predicted small peaks during non-bloom periods. For the less abundant diatom, *Ditylum brightwellii* the improvement is even more evident, with AC-RN101 estimates almost entirely removing the false bloom events and overestimates that plague quantification with CC and traditional feature-based classification. Appropriately interpreted random forest classification has also been useful for studying temporal dynamics in the ciliated micrograzer *Laboea strobila* (Brownlee et al., 2016) but, as for the low concentration diatom, quantification with deep features provides striking fidelity even during period of very low concentration ($\ll 1ml^{-1}$). Notably, quantification with deep features also works extremely well for some challenging cases, such as heterogeneous groupings of small ciliated protozoan taxa with a range of morphologies and relatively small cell types including the nanoflagellate *Pyraminomnas longicauda*, that have proven difficult to distinguish reliably from other nanoplankton on the basis of traditional features.

New CNN architectures are emerging rapidly, with innovations that are expected to lead to even better results going forward(Huang et al., 2016). The availability of these models pre-trained with a dataset such as ImageNet makes it relatively easy for researchers from different domains to take advantage of transfer learning and apply these models to their problems. Even a low-end computer, equipped with an inexpensive GPU, would be able to compute deep features for an automatic plankton system in real time. For instance, with a GPU GTX 1080, and a 100-layer resnet, deep features for all the images in an IFCB sample (we have taken 3500 images as the average sample size), can be computed in less than

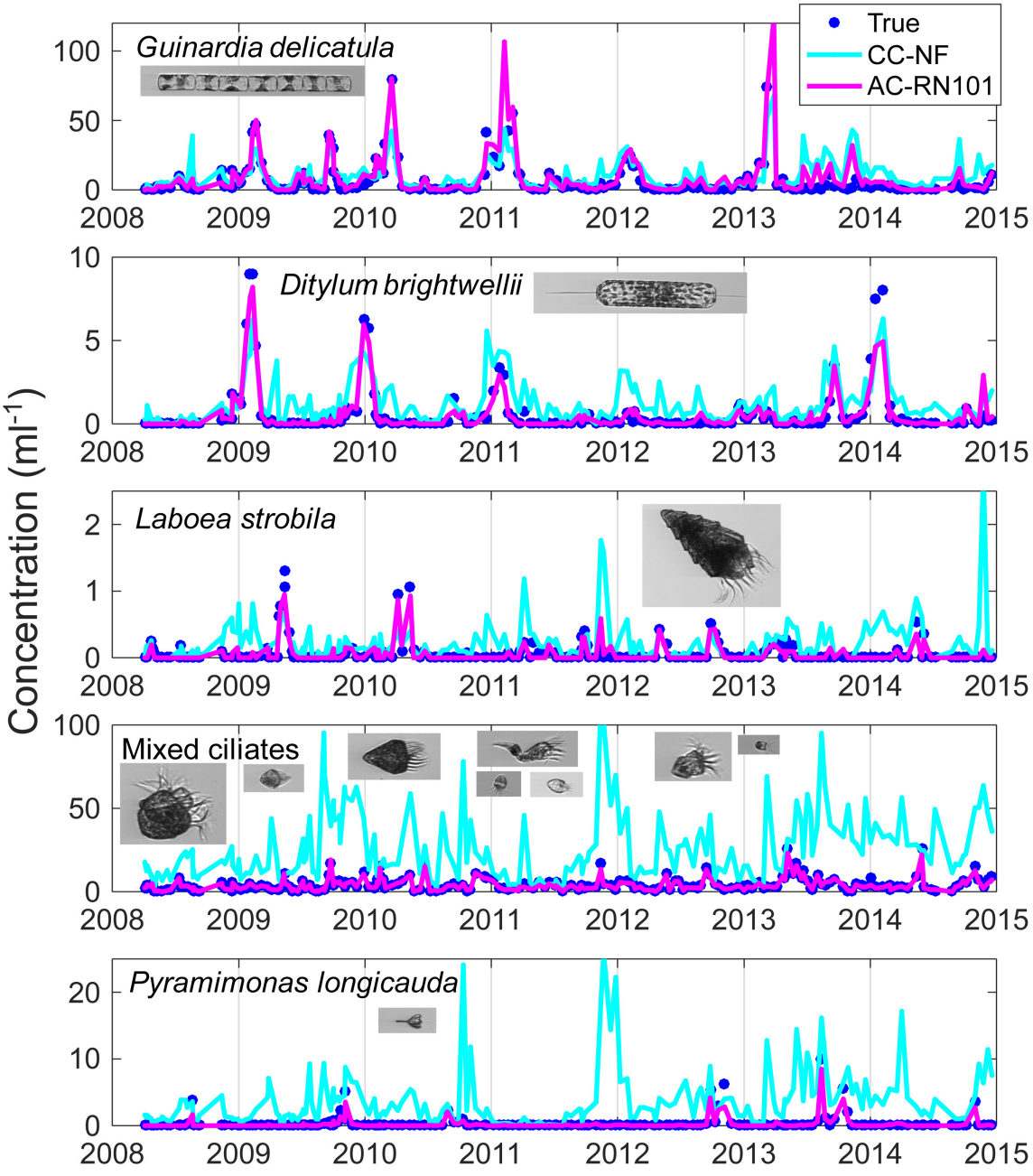


Figure 6: Daily resolved estimates of plankton concentration in the ocean at MVCO for several taxa that exhibit a range of concentrations and patterns of temporal variability during a 7-year period after collection of the images used for classifier training. True concentration (manually verified by experts) is shown along with estimates contrasting the simple CC method with traditional features (CC-NF) with the AC method with fully trained 101-layer network (AC-RN101). *Guinardia delicatula* and *Ditylum brightwellii* are diatoms. *Laboea strobila* is a distinctive species of ciliated protozoa, while "Mixed ciliates" corresponds to a heterogeneous grouping of smaller-sized ciliates of unknown identity. *Pyramimonas longicauda* is a small ($< 10\mu m$) flagellate.

five minutes.

The methods described in this work are fully applicable to other plankton capture systems capable of obtaining and processing water samples containing plankton data. Parameters such as the period between samples, the number of ROIs in each sample or the plankton classes to be identified, are not determining as long as the system is trained and validated with a sufficiently high number of samples. This number is a parameter that should be analyzed carefully while validating the system and that will depend on the complexity of the data to which the system is exposed.

In this work, quantification algorithms have been tested against the traditional Classify and Count (CC) method. Our results show that the improvement of quantification methods as AC, PAC or HDy over CC is typically small. Nonetheless, results also indicate that these quantification methods make the biggest difference when the underlying classifier performs less well, as the adjustments made are bigger and have a greater impact on the final result.

The set of experiments conducted in this work were carried out following a very thorough method (González et al., 2017a). Quantification algorithms were tested in the most similar way to actual working conditions. Also, the error measures used, are appropriate for abundance estimation problems, and allow us to detect potential problems in the built system. It is very interesting to note that all numerical and graphical data generated during the experiments are available online, favouring the detailed analysis of the system performance and its refinement.

Finally, it is important to highlight the importance of the effort of making public and available for download a dataset as the WHOI-Plankton dataset. Researchers from the Woods Hole Oceanographic Institution have made public not only the annotated data used in this paper, but also all data captured by IFCB since 2006. On the one hand, the use of publicly available dataset is important to guarantee experimental reproducibility in studies such as the one described in this paper. On the other hand, the fact of having all this raw data accessible will enable future exploration of different approaches such as autoencoders (Hinton and Salakhutdinov, 2006). The idea behind autoencoders is to build a neural network where the input and output layers are fed with the pixel values of the images. Thus, the network learns how to reconstruct each image in the dataset and the activations in the internal layers can be considered as a compressed representation of the image. Future work with the full IFCB image dataset will make it possible to assess the utility of this unsupervised method that can exploit the huge amount of unlabelled data available.

Acknowledgments

Our thanks to Angel López-Urrutia for his comments and insights during this research. Contributions from HMS and EEP were supported in part by the Simons Foundation, by the National Oceanic and Atmospheric Administration (NOAA) through the Cooperative Institute for the North Atlantic Region

(CINAR) under Cooperative Agreement NA14OAR4320158, and by the National Science Foundation (NSF; Grants CCF-1539256, OCE-1655686).

References

Barranquero, J., Díez, J., and del Coz, J. J. (2015). Quantification-oriented learning based on reliable classifiers. *Pattern Recognition*, 48(2):591–604.

Barranquero, J., González, P., Díez, J., and del Coz, J. J. (2013). On the study of nearest neighbour algorithms for prevalence estimation in binary problems. *Pattern Recognition*, 46(2):472—482.

Beijbom, O., Hoffman, J., Yao, E., Darrell, T., Rodriguez-Ramirez, A., Gonzalez-Rivero, M., and Guldberg, O. H. (2015). Quantification in-the-wild: data-sets and baselines. *arXiv preprint arXiv:1510.04811*.

Bella, A., Ferri, C., Hernández-Orallo, J., and Ramirez-Quintana, M. J. (2010). Quantification via probability estimators. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 737–742. IEEE.

Benfield, M. C., Grosjean, P., Culverhouse, P. F., Irigoien, X., Sieracki, M. E., Lopez-Urrutia, A., Dam, H. G., Hu, Q., Davis, C. S., and Hansen, A. (2007). RAPID: research on automated plankton identification. *Oceanography*, 20(2):172–187.

Bochinski, E., Bacha, G., Eiselein, V., Walles, T. J., Nejstgaard, J. C., and Sikora, T. (2018). Deep active learning for in situ plankton classification. In *International Conference on Pattern Recognition*, pages 5–15. Springer.

Brownlee, E. F., Olson, R. J., and Sosik, H. M. (2016). Microzooplankton community structure investigated with imaging flow cytometry and automated live-cell staining. *Marine Ecology Progress Series*, 550:65–81.

Chatfield, K., Simonyan, K., Vedaldi, A., and Zisserman, A. (2014). Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*.

Chen, T., Li, M., Li, Y., Lin, M., Wang, N., Wang, M., Xiao, T., Xu, B., Zhang, C., and Zhang, Z. (2015). Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv preprint arXiv:1512.01274*.

Cui, J., Wei, B., Wang, C., Yu, Z., Zheng, H., Zheng, B., and Yang, H. (2018). Texture and shape information fusion of convolutional neural network for plankton image classification. In *2018 OCEANS-MTS/IEEE Kobe Techno-Oceans (OTO)*, pages 1–5. IEEE.

Dai, J., Wang, R., Zheng, H., Ji, G., and Qiao, X. (2016a). Zooplanktonet: Deep convolutional network for zooplankton classification. In *OCEANS 2016-Shanghai*, pages 1–6. IEEE.

Dai, J., Yu, Z., Zheng, H., Zheng, B., and Wang, N. (2016b). A Hybrid Convolutional Neural Network for Plankton Classification. In *Computer Vision – ACCV 2016 Workshops*, Lecture Notes in Computer Science, pages 102–114. Springer, Cham.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE.

Dunker, S., Boho, D., Wäldchen, J., and Mäder, P. (2018). Combining high-throughput imaging flow cytometry and deep learning for efficient species and life-cycle stage identification of phytoplankton. *BMC ecology*, 18(1):51.

Firat, A. (2016). Unified framework for quantification. *arXiv preprint arXiv:1606.00868*.

Forman, G. (2008). Quantifying counts and costs via classification. *Data Mining and Knowledge Discovery*, 17(2):164–206.

González, P., Álvarez, E., Díez, J., López-Urrutia, Á., and del Coz, J. J. (2017a). Validation methods for plankton image classification systems. *Limnology and Oceanography: Methods*, 15(3):221–237.

González, P., Castaño, A., Chawla, N. V., and del Coz, J. J. (2017b). A Review on Quantification Learning. *ACM Computing Surveys (CSUR)*, 50(5):74.

González, P., Díez, J., Chawla, N., and del Coz, J. J. (2017c). Why is quantification an interesting learning problem? *Progress in Artificial Intelligence*, 6(1):53–58.

González-Castro, V., Alaiz-Rodríguez, R., and Alegre, E. (2013). Class distribution estimation based on the Hellinger distance. *Information Sciences*, 218:146–164.

Haralick, R. M. and Shanmugam, K. (1973). Textural features for image classification. *IEEE Transactions on systems, man, and cybernetics*, (6):610–621.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Hinton, G. E. and Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507.

Hochreiter, S., Bengio, Y., Frasconi, P., Schmidhuber, J., et al. (2001). Gradient flow in recurrent nets: the difficulty of learning long-term dependencies.

Hu, M.-K. (1962). Visual pattern recognition by moment invariants. *IRE transactions on information theory*, 8(2):179–187.

Huang, G., Liu, Z., and Weinberger, K. Q. (2016). Densely connected convolutional networks. *CoRR*, abs/1608.06993.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Kuhl, F. P. and Giardina, C. R. (1982). Elliptic Fourier features of a closed contour. *Computer graphics and image processing*, 18(3):236–258.

Lee, H., Park, M., and 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)*, pages 3713–3717.

Lloret, L., Heredia, I., Aguilar, F., Debusschere, E., Deneudt, K., and Hernandez, F. (2018). Convolutional neural networks for phytoplankton identification and classification. *Biodiversity Information Science and Standards*, 2:e25762.

Lumini, A. and Nanni, L. (2019). Ocean ecosystems plankton classification. In *Recent Advances in Computer Vision*, pages 261–280. Springer.

Luo, J. Y., Irisson, J.-O., Graham, B., Guigand, C., Sarafraz, A., Mader, C., and Cowen, R. K. (2018). Automated plankton image analysis using convolutional neural networks. *Limnology and Oceanography: Methods*, 16(12):814–827.

Moniruzzaman, M., Islam, S. M. S., Bennamoun, M., and Lavery, P. (2017). Deep learning on underwater marine object detection: A survey. In *International Conference on Advanced Concepts for Intelligent Vision Systems*, pages 150–160. Springer.

Narasimhan, H., Li, S., Kar, P., Chawla, S., and Sebastiani, F. (2016). Stochastic optimization techniques for quantification performance measures. *stat*, 1050:13.

Olson, R. J. and Sosik, H. M. (2007). A submersible imaging-in-flow instrument to analyze nano-and microplankton: Imaging FlowCytobot. *Limnology and Oceanography: Methods*, 5(6):195–203.

Oquab, M., Bottou, L., Laptev, I., and Sivic, J. (2014). Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1717–1724.

Orenstein, E. C. and Beijbom, O. (2017). Transfer Learning and Deep Feature Extraction for Planktonic Image Data Sets. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1082–1088.

Orenstein, E. C., Beijbom, O., Peacock, E. E., and Sosik, H. M. (2015). Whoi-plankton-a large scale fine grained visual recognition benchmark dataset for plankton classification. *arXiv preprint arXiv:1510.00745*.

Pan, S. J. and Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359.

Peacock, E. E., Olson, R. J., and Sosik, H. M. (2014). Parasitic infection of the diatom *Guinardia delicatula*, a recurrent and ecologically important phenomenon on the new england shelf. *Marine Ecology Progress Series*, 503:1–10.

Pérez-Gállego, P., Castaño, A., Quevedo, J. R., and del Coz, J. J. (2019). Dynamic ensemble selection for quantification tasks. *Information Fusion*, 45:1 – 15.

Pérez-Gállego, P., Quevedo, J. R., and del Coz, J. J. (2017). Using ensembles for problems with characterizable changes in data distribution: A case study on quantification. *Information Fusion*, 34:87–100.

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

Saerens, M., Latinne, P., and Decaestecker, C. (2002). Adjusting the outputs of a classifier to new a priori probabilities: A simple procedure. *Neural Computation*, 14(1):21–41.

Sharif Razavian, A., Azizpour, H., Sullivan, J., and Carlsson, S. (2014). CNN features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 806–813.

Sosik, H. M. (2017). Ifcb-analysis wiki. `https://github.com/hsosik/ifcb-analysis/wiki`.

Sosik, H. M., Futrelle, J., Brownlee, E. F., Peacock, E., Crockford, T., and Olson, R. J. (2016). hsosik/ifcb-analysis: Ifcb-analysis software system, initial formal release at v2 feature stage.

Sosik, H. M. and Olson, R. J. (2007). Automated taxonomic classification of phytoplankton sampled with imaging-in-flow cytometry. *Limnology and Oceanography: Methods*, 5(6):204–216.

Sosik, H. M., Peacock, E. E., and Brownlee, E. F. (2015). Annotated plankton images data set for developing and evaluating classification methods.

Wang, C., Zheng, X., Guo, C., Yu, Z., Yu, J., Zheng, H., and Zheng, B. (2018). Transferred parallel convolutional neural network for large imbalanced plankton database classification. In *2018 OCEANS-MTS/IEEE Kobe Techno-Oceans (OTO)*, pages 1–5. IEEE.

Appendix A. Plankton Dataset Labels

Short quantifier label	WHOI-Plankton dataset labels	Taxonomic range in class, other description
Asterionellopsis	*Asterionellopsis*	*Asterionellopsis spp.*
Centraulina	*Centraulina pelagica*	*Centraulina pelagica*
Ceratium	*Ceratium*	*Ceratium spp.*
Chaetoceros	*Chaetoceros, Chaetoceros didymus*	*Chaetoceros spp., Chaetoceros didymus*
Corethron	*Corethron hystrix*	*Corethron hystrix*
Coscinodiscus	*Coscinodiscus*	*Coscinodiscus spp.*
Cylindrotheca	*Cylindrotheca*	*Cylindrotheca spp.*
DactFragCerataul	*Dactyliosolen fragilissimus*	*Dactyliosolen fragilissimus*
Dactyliosolen	*Dactyliosolen blavyanus*	*Dactyliosolen blavyanus*
Dictyocha	*Dictyocha*	*Dictyocha spp.*
Dinobryon	*Dinobryon*	*Dinobryon spp.*
Dinophysis	*Dinophysis*	*Dinophysis spp.*
Ditylum	*Ditylum brightwellii*	*Ditylum brightwellii*
Ephemera	*Ephemera*	*Ephemera spp.*
Eucampia	*Eucampia*	*Eucampia spp.*
Euglena	Euglenia (subclass)	Euglenia (subclass)
Guinardia	*Guinardia delicatula*	*Guinardia delicatula*
Guinardia_flaccida	*Guinardia flaccida*	*Guinardia flaccida*
Guinardia_striata	*Guinardia striata*	*Guinardia striata*
Gyrodinium	*Gyrodinium, Amphidinium, Katodinium, Torodinium, Proterythropsis*	*Gyrodinium spp., Amphidinium spp., Katodinium spp., Torodinium spp., Proterythropsis spp.*
Laboea	*Laboea strobila*	*Laboea strobila*
Lauderia	*Lauderia*	*Lauderia spp.*
Leptocylindrus	*Leptocylindrus*	*Leptocylindrus spp.*
Licmophora	*Licmophora*	*Licmophora spp.*
Myrionecta	*Mesodinium_sp*	*Mesodinium spp.*
Odontella	*Odontella*	*Odontella spp.*
Paralia	*Paralia*	*Paralia spp.*

Phaeocystis	*Phaeocystis, Parvicorbicula socialis*	*Phaeocystis globosa, Parvicorbicula socialis*
Pleurosigma	*Pleurosigma*	*Pleurosigma spp.*
Prorocentrum	*Prorocentrum*	*Prorocentrum spp.*
Pseudonitzschia	*Pseudonitzschia*	*Pseudonitzschia spp.*
Pyramimonas	*Pyramimonas longicauda*	*Pyramimonas longicauda*
Rhizosolenia	*Rhizosolenia*	*Rhizosolenia spp.*
Skeletonema	*Skeletonema*	*Skeletonema spp.*
Stephanopyxis	*Stephanopyxis*	*Stephanopyxis spp.*
Thalassionema	*Thalassionema*	*Thalassionema spp.*
Thalassiosira	*Thalassiosira*	*Thalassiosira spp.*, other similar centric diatom species
Thalassiosira_dirty	*Thalassiosira* with external detritus	Thalassiosira spp.with external detritus
bad	contains only camera field background	contains only camera field background
ciliate_mix	*Didinium*, *Euplotes*, *Leegaardiella ovalis*, *Pleuronema*, *Strobilidium*, *Tiarnia*, *Tontonia*, and unidentified ciliates	*Didinium spp.*, *Euplotes spp.*, *Leegaardiella ovalis*, *Pleuronema spp.*, *Strobilidium spp.*, *Tiarnia spp.*, *Tontonia spp.*, and unidentified ciliates
clusterflagellate	*Corymbellus*	*Corymbellus spp.*
detritus	detritus	detritus
dino30	ameoba, *Akashiwo*, *Hetercapsa triquetra*, *Karenia*, *Protoperidinium*, *Vicicitus globosus*, unidentified dinoflagellates	*Akashiwo spp.*, *Hetercapsa triquetra*, *Karenia spp.*, *Protoperidinium spp.*, *Vicicitus globosus*, unidentified dinoflagellates and amoeba
kiteflagellates	*Chrysochromulina lanceolata*	*Chrysochromulina lanceolata*
mix	*Cryptophyta*, *Pyramimonas*, *Chrysochromulina*, *Heterocapsa rotundata*, unidentified nanoplankton	*Cryptophyta, Pyramimonas spp., Chrysochromulina spp., Heterocapsa rotundata, unidentified nanoplankton*
mix_elongated	miscellaneous diatom fragments	miscellaneous diatom fragments

na	dino10, other, diatome_flagellate, other_interaction, *Leptocylindrus mediterraneus*, pennates on diatoms, *Delphineis*, *Bacillaria*, *Bidulphia*, *Cochlodinium*, *Emiliania huxleyi*, *Pseudochattonella farcimen*, bead, bubble, pollen, spore, zooplankton	Other rare and/or unidentified taxa
pennate	miscellaneous pennate diatoms	miscellaneous pennate diatoms
tintinnid	Tintinnida	Tintinnida

Appendix B. Mean absolute error by class

Class	NF CC (10^{-4})	NF AC (10^{-4})	RN-101 CC (10^{-4})	RN-101 AC (10^{-4})
Asterionellopsis	56.97 / 1.60	43.16 / 2.50	2.64 / 0.25	**2.61** / 0.26
bad	148.35 / 15.04	162.01 / 18.44	**40.43** / 5.16	40.78 / 5.31
Cerataulina	89.25 / 4.39	61.97 / 4.61	28.44 / 2.87	**27.52** / 2.79
Ceratium	0.97 / 0.07	0.95 / 0.07	0.32 / 0.05	**0.31** / 0.05
Chaetoceros	334.28 / 6.17	136.26 / 5.47	42.49 / 1.86	**38.81** / 1.97
ciliate_mix	148.76 / 3.96	71.13 / 4.29	6.56 / 0.40	**6.47** / 0.36
clusterflagellate	80.33 / 3.14	67.71 / 5.56	0.61 / 0.09	**0.57** / 0.09
Corethron	75.28 / 2.32	39.16 / 3.01	2.04 / 0.21	**1.89** / 0.19
Coscinodiscus	2.42 / 0.17	2.61 / 0.29	**0.82** / 0.11	0.87 / 0.12
Cylindrotheca	58.14 / 2.82	63.26 / 3.00	23.18 / 1.98	**22.50** / 1.93
DactFragCerataul	192.36 / 4.97	75.18 / 5.29	20.19 / 1.06	**14.69** / 1.04
Dactyliosolen	72.64 / 2.14	60.30 / 3.11	**9.49** / 0.85	9.61 / 0.87
detritus	555.91 / 20.27	652.58 / 24.07	409.26 / 23.09	**387.54** / 23.79
Dictyocha	20.72 / 1.02	14.98 / 1.22	1.13 / 0.19	**1.13** / 0.20
dino30	148.90 / 4.82	128.98 / 4.78	138.26 / 5.05	**121.86** / 5.20
Dinobryon	27.72 / 0.92	18.52 / 1.04	3.54 / 0.33	**3.30** / 0.32
Dinophysis	18.10 / 0.83	15.44 / 1.49	**1.59** / 0.16	1.66 / 0.17
Ditylum	7.91 / 0.43	6.11 / 0.52	1.86 / 0.22	**1.67** / 0.20
Ephemera	3.99 / 0.20	2.29 / 0.23	**0.57** / 0.11	0.58 / 0.11
Eucampia	29.93 / 1.02	25.28 / 1.66	**2.01** / 0.26	2.09 / 0.27
Euglena	17.63 / 0.60	4.60 / 0.51	4.08 / 0.37	**3.91** / 0.37
Guinardia	62.17 / 4.42	50.89 / 3.97	20.88 / 1.27	**20.48** / 1.28
Guinardia_flaccida	4.56 / 0.23	3.49 / 0.30	**0.72** / 0.08	0.72 / 0.08
Guinardia_striata	12.56 / 1.28	8.64 / 1.22	4.25 / 1.24	**4.18** / 1.24
Gyrodinium	33.73 / 1.60	25.01 / 2.14	4.89 / 0.34	**4.81** / 0.34
kiteflagellates	14.35 / 0.79	16.26 / 1.13	**0.75** / 0.21	0.75 / 0.22
Laboea	2.02 / 0.15	1.85 / 0.17	0.50 / 0.09	**0.47** / 0.08
Lauderia	2.22 / 0.30	7.61 / 1.14	**0.19** / 0.04	0.22 / 0.05
Leptocylindrus	217.77 / 16.64	184.05 / 12.26	91.86 / 7.48	**87.28** / 7.17
Licmophora	3.61 / 0.16	2.45 / 0.22	0.42 / 0.06	**0.42** / 0.06
mix	3135.64 / 29.01	1296.29 / 39.10	583.80 / 23.05	**459.72** / 22.77
mix_elongated	305.75 / 7.28	177.42 / 9.49	119.55 / 6.54	**111.25** / 6.81
Myrionecta	65.59 / 1.92	25.91 / 2.15	**2.33** / 0.17	2.38 / 0.16
na	484.27 / 7.93	125.58 / 9.29	19.33 / 0.88	**18.90** / 0.90
Odontella	0.68 / 0.06	1.13 / 0.14	**0.10** / 0.02	0.12 / 0.03
Paralia	22.52 / 0.93	17.57 / 1.49	**0.60** / 0.06	0.60 / 0.06
pennate	101.97 / 2.83	30.42 / 2.28	**8.05** / 0.79	8.35 / 0.79
Phaeocystis	13.62 / 0.58	20.35 / 1.28	1.98 / 0.32	**1.94** / 0.31
Pleurosigma	2.43 / 0.17	2.40 / 0.21	0.76 / 0.09	**0.75** / 0.09
Prorocentrum	15.38 / 0.58	10.92 / 0.66	**4.35** / 0.49	4.36 / 0.48
Pseudonitzschia	69.37 / 2.54	28.30 / 2.34	**11.66** / 0.74	11.80 / 0.75
Pyramimonas	24.49 / 0.82	15.42 / 1.14	1.12 / 0.27	**1.04** / 0.25
Rhizosolenia	82.21 / 3.95	61.50 / 3.55	20.99 / 1.41	**20.00** / 1.39
Skeletonema	272.94 / 9.43	229.31 / 13.01	**28.52** / 2.65	28.57 / 2.66
Stephanopyxis	0.38 / 0.04	0.51 / 0.09	**0.04** / 0.01	0.04 / 0.01
Thalassionema	10.25 / 0.37	5.55 / 0.37	**1.48** / 0.11	1.48 / 0.11
Thalassiosira	210.93 / 4.94	75.49 / 5.07	19.93 / 0.98	**19.32** / 0.99
Thalassiosira_dirty	48.70 / 2.29	25.04 / 2.61	6.13 / 0.83	**5.45** / 0.79
tintinnid	8.71 / 0.43	6.09 / 0.57	**1.75** / 0.17	1.77 / 0.17

Table B.1: Absolute Errors (AE) Mean / Standard Error by class over all test samples. Results for CC and AC methods using normal features (NF) and RN-101 features. Lowest errors per class shown in bold.