

Phytoplankton Functional Groups Automatic Recognition using Convolutional Neural Networks

Robin Fuchs*, Melilotus Thyssen†, Gérald Gregori‡, Denys Pommeret §, Samuel Soubeyrand ¶

June 16, 2020

Abstract

Flow Cytometry has enabled to collect ~~very~~ significant phytoplankton datasets in quasi-real time. ~~However, these data have then to be post-processed in order to~~ count the number of cells belonging to each phytoplankton species or group of species sharing common characteristics, called phytoplankton functional groups (PFT). ~~This manual post-processing is~~ very time-consuming and may be a source of classification errors for which we provide here an estimation.

As a result, there were many recent propositions to make this classification automatic based on Machine Learning methods. However, most of these methods presents either a long training process, are unable to predict the class of very small cells or need to manually design some features. Our method is based on recent advances in Deep Convolutional Neural Networks and classify the phytoplankton functional groups thanks to cytometric curves provided by a ~~flow cytometer~~. We show on a novel long and ~~high frequency~~ time series that our model presents a high accuracy **that that seems not affected by infra-day cells cycles**. It also exhibits a significant per class precision despite of the extreme imbalanced nature of the data. Finally, the model seems also to be able to reuse features learnt on some particles to predict the class of particles coming from another location. Thus, it might be redeployed in other geographic contexts with minor adjustments.

*robin.fuchs@mio.osupytheas.fr

†melilotus.thyssen@mio.osupytheas.fr

‡gerald.gregori@mio.osupytheas.fr

§denys.pommeret@univ-amu.fr

¶samuel.soubeyrand@inra.fr

1 Introduction

The datasets in Oceanography are conspicuous for being large (with a high number of observations) and of high dimension (comprising a substantial number of variables). This makes oceanography a very suited field for statistical analysis. This is particularly the case of phytoplankton-related data. Phytoplankton, under a unique denomination, actually contains several thousand species. Some of these species share common characteristics such as for instance the size **or the division time** which makes it possible to design groups of species called phytoplankton functional groups (PFTs).


The study of the PFTs is of primary importance given that the contribution of phytoplankton to primary production, the amount of underwater dissolved CO_2 absorbed by phytoplankton cells per unit of time, is equivalent to all of the primary terrestrial production. This is the case even if phytoplankton cells represents less than 1% of the terrestrial autotrophic biomass [7]. This means that phytoplankton has a very rapid growth capacity (it can divide several times a day) and therefore highlights the need for high frequency observations to encompass the **morphologic diversity of those cells** and assess correctly the classification power of statistical models.

The development of flow cytometry has made possible a vast automated data acquisition on PFTs given that a cytometer can process up to 10,000 cells per second. From each cell in the sample, the cytometer is able to generate a set of curves which represents the optical profile of each cell. After reprocessing, the oceanographers manually determine the different functional groups existing in the data. These operations are however very time-consuming and their automation seems today necessary.

Automating classification tasks using statistical methods, *i.e.* designing a method to automatically assign a label to an observation based on its features, has received special interest in oceanography for 20 years. ~~Classification tasks were actually often referred as "clustering" tasks in the Oceanographic literature [add source], which can be confusing with regards to the actual meaning of "clustering" in statistics.~~ In the following, we will use the term classification which is more

in line with the statistical literature.

Concerning phytoplankton classification, most of the effort seems to have been spent on image processing and computer vision. One can for example cite the count of coccoliths using shallow neural networks in the seminal work by Beaufort and Dollfus (2004) [3] or more recently the works by Dunker & al. (2019) [5] or by González & al. (2019) [8] based on Residual Neural Networks and Transfer Learning [16] [check if Dunker is residual and transfer]. Automated recognition of groups of individuals from the **curves from flow cytometry** was less explored with some notable exceptions such as Malkasian (2011) [13] which classifies the species of phytoplankton by plunging these curves into a Fourier basis and calculating distances between them. Other works using Flow Cytometry data but not on phytoplankton cells also exist as in del Barrio and al. (2019) [1] where they create curves templates and classify the observations curves with respect to these templates using Wasserstein distance and optimal transport.

[Literature review too small ?] 

Contrary to these works, we aim to automate the cells count of functional groups and not of the species, which might be more challenging as the morphological diversity among a set of species is at least superior to the morphological diversity among each species. The raw data used for this classification are the **cytometric curves** of each cell going through the cytometer. Compared to phytoplankton images, the size on disk of the 5 curves per particle is approximately the same : 4.5kb per observation. However, classifying and then count the cells using cytometric curves presents a real advantage because it can deal with particles smaller than 20 microns. On the contrary, the image resolution is not sufficient to obtain proper images of such small particles. This is a real issue considering that the very vast majority of phytoplankton particles in the *in-situ* samples are smaller than this threshold. The second main advantage is the shorter training process because of the absence of transfer learning procedure [14], contrary to the images that require to fine-tune very heavy networks such as Residual Networks [9].



In this work we use recent advances in now standard Deep Convolutional Neural Networks [10] to automate PFTs classification. This is a challenging task because of the morphological diversity evoked earlier, to the format of raw data and to the dramatically imbalanced nature of datasets. These aspects also make manual classification challenging for oceanographers. Hence, we begin this article with an assessment of the error range coming from manual data classification (as those data are then considered as ground truth to train our model).

The first section thus presents an estimation of the manual clustering error that have been performed during a small experiment. We pursue by presenting the model specification and the data. Section 4 shows the performance of the model on three use cases. First, our model is compared with other benchmark models on a dataset made of several acquisitions from the FUMSECK data campaign. Then, the model is used on the new high-frequency long time series coming from the Endoume marine station (SSLAMM) located in Marseille in order to check that the predictions are not affected by seasonality nor infra-day cell cycles. Finally, we give insights about the robustness of our model by using the features learnt on a dataset and predicting observations from a different data source. Section 5 closes this work by analysis the limits of our work and giving axes for future research.

2 Manual clustering estimation

[THIS EXPERIMENT HAVE NOT BEEN PERFORMED YET, THE TEXT WILL CHANGE ACCORDINGLY]

In this section, we aim to assess the range of error commonly made while performing a manual clustering. The samples treated by the cytometer are very imbalanced between PFTs, the ratio between the most and less represented class being between 10^4 and 10^5 . Hence, the less represented particles can be "hidden" by the most represented ones on the 2D scatter plots used by cytometrists to identify the PFTs as the one presented in Figure 1 [have to be more precise on how cytometrists make classification ?]. This can create very sig-

nificant biases in the estimated count of these classes. For instance, in the data there are typically at most a few dozen of microphytoplankton particles while there are dozen of thousands of synechococcus particles. Hence, misclassifying 10 particles of microphytoplankton could result in a 30% error rate while misclassifying 10 particles of synechococcus would be negligible.

This issue is a type of statistical data-contamination and can have significant effects on the patterns learnt by the network. Without any estimation of this contamination, it is impossible to disentangle the errors coming from the data from the error coming from the training process. We could not find an estimation of this phenomenon and conducted a small study to give a raw assessment of the inter-individual manual classification errors.

In our experiment, we have asked five researchers to classify data coming from three samples sampled at different seasons and time of the day. They were given a list of the PFTs groups existing in those samples and an approximate repartition of the different groups in the data. Once they have performed the clustering, it is possible to measure the standard errors in the count of each functional group. The results are presented in Figure 2. Hence the closer to zero the standard error is, the more the classification is the same for all researchers.

| PFT | Airbubbles | Cryptophytes | Microphytoplankton |
|----------------|------------|--------------|--------------------|
| Mean count | | | |
| Standard Error | | | |

| PFT | Nanoeukaryotes | Noise | Picoeukaryotes |
|----------------|----------------|-------|----------------|
| Mean count | | | |
| Standard Error | | | |

| PFT | Prochlorococcus | Synechococcus |
|----------------|-----------------|---------------|
| Mean count | | |
| Standard Error | | |

3 Model and data description

In this section we detail the nature of the data, their pre-treatment and the specification of our network.

3.1 Data presentation

The general principle of the flow cytometer is as follows: a sample of water to be analyzed is pumped in the cytometer from the environment studied. The cytometer aligns the cells in suspension in the sample one by one thanks to the generation of a laminar net that generates an hydrodynamic focus and makes them pass in front of a laser beam. This method allows each cell or particle to pass in front of a coherent laser beam. The interception of this beam by the particles generates a set of optical profiles of diffusion and natural fluorescence when pigments are present. These optical profiles take the form of a set of curves. The frequency of acquisition of curves of the cytometer is 4 MHz which gives it a collection capacity of barely 10,000 cells per second. The datasets generated are therefore rapidly massive.

In our case the cytometer issues five curves: the curvature curve, the red (FLR) and orange fluorescence (FLO) curves and the forward (FWS) and Sideward scatter (SWS) (other cytometers might have a different number of curves) [have to give details about the curves ?]. Because of the high proportion of noise particles, a common practice is to perform two types of data acquisitions using two Red Fluorescence (FLR) thresholds: a low one here denoted FLR6 and a high one denoted FLR25. The lower threshold enables to count the particles that have the smallest total red fluorescence (which are also the smaller particles). The FLR25 enables to clear out the small particles in order to better count the biggest ones. Then the total count by class is simply the count of low fluorescent particles in the FLR6 file and of high fluorescent particles in the FLR25 file.

Hence, each observation is made of five curves which length is closely linked to the size of the particle (the bigger the particle the longer the sequence). In order for all sequences to be comparable, the curves have been interpolated using quadratic interpolation. Using truncated and padded with zeros sequences as in Natural Language Processing (NLP) has also been implemented and led to poorer performance.

We have chosen a fixed length of 120 values for each curve of all observations that corresponded approximately to the third quartile of the distri-

bution of particles curves lengths. The influence of this length choice on performance has not however been tested and could be in further research. Once the curves resized, one obtains for each observation five curves of length 120 or alternatively a 5×120 matrix which a representation is given in Appendix 2.

Concerning the origin of the data, two main sources of data have been used in this work: The FUMSECK Campaign data and the Endoume Marine Station (SSLAMM) data.

The FUMSECK campaign was a cruise that took place from April, 30, 2019 to May, 05, 2019 off the Ligurian Sea. [add small description]. During this campaign, 610 cytometer samples have collected representing barely ... particles.

The SSLAMM time series has started in September 2019 at the Endoume Marine Station in Marseille. Some sea water is continuously pumped at 10 meters from the coast and delivered into the laboratory buildings where analyses are conducted. A cytometer is making acquisitions into this water flow every two hours, hence generating 12 FLR6 and 12 FLR25 files per day. Such an infrastructure has no equivalent in Europe and to our knowledge in the World [is it true?]. Having data at such a high frequency over a period of several years could enable to answer very fine research questions that were until now impossible to investigate due to the lack of data. Cruises are often limited by weather conditions, being cancelled in case of extreme conditions. As the data acquisition process is not independent of the value taken by the quantity of interest (the multivariate distribution of the groups), this could result in significant biases in effects estimations. Moreover, the behavior of the phytoplankton cells during these extreme events is not fully understood, letting room for future research on these data.

In both cases, the same functional groups nomenclature have been used. It is composed of six phytoplankton classes: Prochlorococcus, Synechococcus, Picoeukaryotes, Nanoeukaryotes, Cryptophytes and Microphytoplankton. Two additional classes have been added: noise particles and airbbubbles. Noise particles are actually very heterogeneous: detritic particles *stricto sensu*, phytoplankton predators that present a fluores-

cence, phytoplankton cells in decomposition... etc. The airbubbles begin to appear in the cytometer when some manual interventions have to be performed. It is then more a class helpful for machine monitoring rather than for purely scientific purposis.

3.2 Prediction pipeline presentation

Convolutional networks have known a very fast development in image recognition and computer vision during the last ten years starting with the seminal works of Krizhevsky and al. (2012). [10]. The general idea of such a network is to learn a series of filters that detect some patterns in images and help to discriminate between the classes. More formally, these filters are tables of coefficients used to compute convolutional operations on the data output by the last layer. Compared to Dense layers the convolutional ones relies on the assumption that regions in the images conveys useful information and that close pixels often carry very redundant information. As a result, the total number of parameters of the model is reduced and the training of the model is kept tractable even for big three color channels (Red, Green, Blue) images. Once the features extracted by the convolutional networks, one can use Dense layers at the end of the network to perform the classification itself, which is what is conducted here.

A $l \times L$ color image is represented by $l \times L$ coefficients for each of the three RGB color channel i.e. $l \times L \times 3$ integer coefficients ranging from 0 to 255. In the case of black and white images it is a $l \times L \times 1$ or $l \times L$ table. In our case, our network is not applied to images but to 5×120 matrices of float coefficients which are formally speaking similar to black and white images. Hence, the same networks as for images can be used with minor modifications.

An alternative option rather than using the five stacked raw curves is to generate the images of each curve and to use these 5 images per observation as input. The two possible representations (matrix or curve) are again given in Appendix 2. Yet, taking the curves images actually dilutes the

signal held by the curves among a very substantial number of white pixels. The signal was then too hard to perceive for the network and resulted in network training failures.

Thus, we decided to keep the matrix representation that is the raw signal itself (up to a quadratic interpolation) rather than manually designed features. We expect this very rich signal to be highly efficient for classification purposes as information used by oceanographers is much more simplified and enable the manual classification. The model architecture is presented *model_{spe}*. Features are first extracted by three convolutional layers. Then, local averages of the coefficients are taken by a Global Average Pooling layer relying on the same idea that a part of the signal is redundant if taken at a too fine level. These "averaged features" are then treated by a series of dense layers. The dropout layers enable not to train every neuron of the layer. It avoids the network to become too specialize over a dataset, which is known as "model overfitting" in Machine Learning.

At the end of the dense layers a softmax layer is computing the probabilities that an observation belongs to each class and the loss of the model is computed.

The loss is measuring the gap existing between the class probabilities that the model outputs and the actual class of the observation. This gap represents an error that is then back-propagated to update the parameters of the network accordingly. The loss is a key component of the model and is the main way we have decided to treat the fact that the data were very imbalanced. We have also randomly sampled some observations of very represented classes to fasten the training time of the model. The use of more advanced under-sampling techniques such as Edited Nearest Neighbors or Tomek's links [2] is not straightforward due to the very particular functional form of the data. Projecting the data into simpler spaces and performing under-sampling methods in those spaces have given not better results and is not used in our final model training workflow.

The canonical loss for single-label multivariate classification is the negative log-likelihood or categorical cross-entropy (negLL). Its expression is

given by :

$$\text{negLL} = - \sum_{k=1}^K \sum_{i=1}^n (y_{i,k} * \log(\hat{y}_{i,k}))$$

with $i \in [1, n]$ the observation index, $k \in [1, K]$ the class index, $y_{i,k}$ equals 1 if observation i is in class k and $\hat{y}_{i,k}$ the probability that observation i is in class k predicted by the model.

This loss gives the same weights to all errors whatever the classes of the observations. Three refinements have been implemented in this sense here: weighted categorical loss entropy, focal loss [11] and class-balanced loss [4] in order to achieve a good overall accuracy but also a good per-class accuracy.

The weighted negLL (WnegLL) is a straightforward extension of the negLL where the loss incurred when badly predicting some classes is higher than badly predicting others. In our case, the less represented classes have been over-weighted. However, this approach is very sensitive to the way the weights are computed [add source]. The most common practice is to set those weights to $\frac{1}{n_k}$ or to $\frac{1}{\sqrt{n_k}}$, with n_k the number of observations of class k . Intuitively as our data are extremely imbalanced, using $\frac{1}{n}$ might lead to too small weights for the most represented data and we expect the second one to perform better. Additionally, according to common statistical standards, one should use different data for training, validating and testing the model. Tuning those weights have then to be performed on another dataset and not reuse those data for other purposes. [TO REWRITE].

comme tu veux, j'ai rien compris de toute façon ;)

Over-weighting very unrepresented data hinges on the hypothesis that rare classes are difficult to predict. This claim can be justified by the fact that bigger particles tend to present a wider range of morphological diversity and that they are also the least represented PFTs. However, in order not to rely on this hypothesis nor on the parametric form of the weights formula we have used the newly introduced focal loss (FL) its generalization, the Focal Class-Balanced loss (FCBL), which have the following expressions: [check formulations] [change \hat{y} for $p_{i,k}$]

$$FL = - \sum_{k=1}^K \alpha_k \sum_{i=1}^n y_{i,k} (1 - \hat{y}_{i,k})^\gamma \log(\hat{y}_{i,k})$$

and

$$FCBL = - \sum_{k=1}^K \frac{1 - \beta}{1 - \beta^{n_k}} \sum_{i=1}^n y_{i,k} (1 - \hat{y}_{i,k})^\gamma \log(\hat{y}_{i,k})$$

α is a class-dependent weight in the same spirit as the WnegLL weights. γ is a focusing parameter, it defines how little the contribution of easy-to-predict observations is. β controls how the re-weighting depends on class frequency: $\beta = 0$ corresponds to negLL and $\beta = 1$ correspond to WnegLL with the weights equals to $\frac{1}{n_k}$ are three hyper-parameters, n_k is the number of observations of class k .

From the expressions below, it appears that the focal loss decreases the contribution of easy well-classified observations, *i.e.* with $\hat{y}_{i,k}$ close to one, through the training thanks to the "modulating" factor $(1 - \hat{y}_{i,k})^\gamma$ but also with some weights α as in WnegLL. The FBCL automates in some way the choice of α , but also relies on a new parameter β . We are implementing the three losses, give the hyper-parameters value chosen in Appendix D and compare the performances obtained in the next section.

[Add the fact that there is no need for really balanced datasets]

Beyond the choice of the loss, an important choice is the one of the optimizer which deals with how the parameters of the network are updated with respect to the loss. We have here benchmarked two optimizers: Adam and its extensions Ranger. Ranger is the combination of two very recent publications: RectifiedAdam [12] and Lookahead [17]. In order not to be stuck in bad local maxima, it is a common practice to slowly update the parameters of the models at the beginning of the training, where really promising parameters regions are not for the moment identified. This adaptation rate of the parameters with respect to the loss is called the learning rate of the model and is hence often chosen to be small in the early stages of the training process [15]. Radam adapts the learning rate to avoid the learning rate

variance to grow too substantially, which according to the authors is often very detrimental to the learning process. On the other hand, Lookahead enables the network to get a better understanding of the loss topology. In order to do so, two sets of weights are designed by Lookahead: a faster set of weights that are often updated to "explore" the loss surface and a slower set of weights to ensure the stability of the learning process. The faster set of weights is updated using not all the data but only a set of several observations batches to get a raw idea of promising regions to explore. In the ranger case those fast weights are updated thanks to the Radam optimizer.

As for the losses and most of the parts of the neural networks, the behaviour of the optimizers are also ruled by a set of hyperparameters that need to be chosen by the user. The number of possible combinations is far too high for all the combinations to be tried and then pick the best network specification.

One popular approach relies on Bayesian Hyper-optimisation algorithms [add source] which are implemented in our case in the Python library Hyperas (Hyperopt for Keras). The idea of Hyper-optimisation methods is to consider hyperparameters as statistical random variables with a prior and to identify posterior regions that presents a low loss value. Hence, some draws are taken from the prior distributions, the model is evaluated and low loss regions are identified and focused on. It avoids to spend very significant computational efforts on non-promising regions of the hyper-parameters space as it is often the case using standard line search.

[end of this part to rewrite:] After model predictions a post-processing is performed. All the particles presenting low predicted probabilities are given the noise label rather than the label of the highest probability class. This is due to the fact that the noise particles are defined by the fact that they are not phytoplankton cells rather than as being a biologically consistent class. Conceptually by creating this noise class, we are here making a two stages procedure in a single step. The first stage would be to predict if the particle is a phytoplankton cell or not. If it is, then a second step would be performed to determine its class. [give more details]. Doing this post-

processing enables to account for the first step of this procedure.

In the following, the data were split in three parts: the training the set, the validation and the test set. The model was trained on the training set, the choice of the parameters done according to the performance of the validation sets and the final choice between the losses performed on the test set.

4 Results on in-situ data

This section presents three cases of applications of our model. First, the model is benchmarked again other models on the FUMSECK campaign data in order to illustrate its predictive power. Then, predictions are made upon samples acquired at the Endoume marine station in order to show the invariance of our network to seasonality and infra-day shape changes of the cells. Finally, the model is trained on FUMSECK data to predict Endoume samples to illustrate its generalisation power.

4.1 Model benchmark on FUMSECK data

In this section, we train the ~~our~~ model over 44.500 observations and benchmark it against other supervised models in order to illustrate its performances. The algorithms compared are Light GBM (LGBM), Feed Forward Neural Network (FFNN), the k-Nearest Neighbors (kNN) and Support Vector Machines (SVM). LGBM has been chosen because it is in practice very used by Machine Learning practitioners in several fields of application and has won several Kaggle challenges (as it was the case of Random Forests models earlier on). The last three models have been picked as they were the one implemented in the Rclus-Tool package [add source], which is a package implementing Machine methods applied to flow cytometry data.

However, these models could not process the raw signal as it is the case in our model and there is a need to manually compute some features. The presented results have then to be considered

by keeping in mind that the choice of the features created from the signal highly influence the performances of the models. We rely on the 70 features [number to precise] created by default by the CytoClus ©software as they are widely used by oceanographers. The feature list is given in Appendix E.

The metrics reported for each class and each algorithm are the precision and the recall. The precision is the proportion of particles actually belonging to class k among all those identified as belonging to class k by the algorithm. The recall is the proportion of particles effectively belonging to class i among all the particles of class k existing in the dataset.

There is a precision / recall arbitration and obtaining precision and recall close to 1 constitutes the horizon of any supervised algorithm. For example, an algorithm that would predict "noise" for all particles would have a recall of 1 and relatively poor precision for the category "noise".

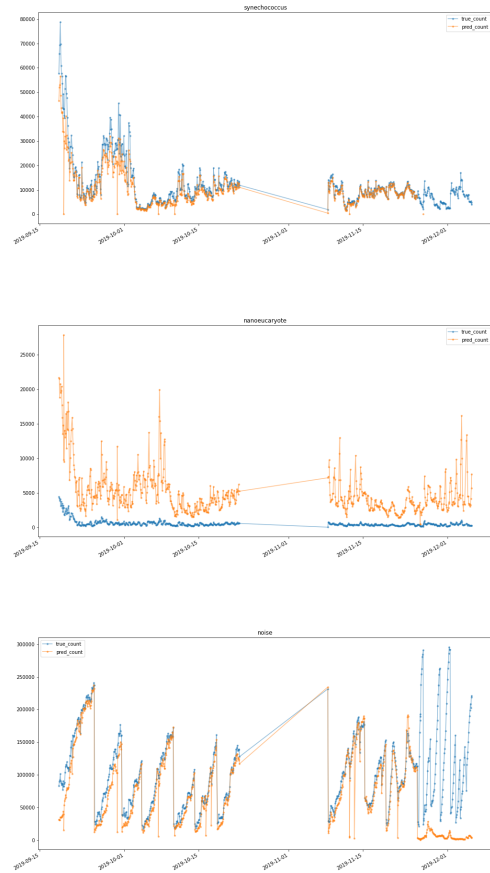
| PFT | Airbubbles | | Cryptophytes | | Microphytoplankton | |
|--------------|------------|--------|--------------|--------|--------------------|--------|
| Model/Metric | Precision | Recall | Precision | Recall | Precision | Recall |
| CNN | | | | | | |
| LGBM | | | | | | |
| FFNN | | | | | | |
| kNN | | | | | | |
| SVM | | | | | | |

| PFT | Nanoeukaryotes | | Noise | | Picoeukaryotes | |
|--------------|----------------|--------|-----------|--------|----------------|--------|
| Model/Metric | Precision | Recall | Precision | Recall | Precision | Recall |
| CNN | | | | | | |
| LGBM | | | | | | |
| FFNN | | | | | | |
| kNN | | | | | | |
| SVM | | | | | | |

| PFT | Prochlorococcus | | Synechococcus | |
|--------------|-----------------|--------|---------------|--------|
| Model/Metric | Precision | Recall | Precision | Recall |
| CNN | | | | |
| LGBM | | | | |
| FFNN | | | | |
| kNN | | | | |
| SVM | | | | |

4.2 Prediction of Endoume Time Series data

This section illustrates the ability of our model to be deployed in production and to provide accurate estimates of the count of each class on a daily basis. Figure 4.2 presents the time series obtained with manual counts and automatic count for the ~~synechococcus, the noise and the nanoeukaryote~~ particles.



[labels too small and the end of the time series is not predicted]

~~The noise and synechococcus particles are well counted whereas the model tends to largely overcount the nanoeukaryotes. Looking at the confusion matrix, it reveals that a part of the noise is actually taken to be nanoeukaryotes [to check].~~ What is striking is that the quality of the predictions does not seem to vary with the hour of the day nor the month on which there are performed. It means that our predictions are not influenced by the cell divisions that might occur at an infra-day frequency.

[detail how the training set was built]

[Sortir la série temporelle de diffusion et fluorescence → conversion automatique en taille, montre bien la diversité morphologique].

4.3 Estimation of the generalization power of the model

Finally, we provide an illustration of how general the features learnt by the model are by choosing two different data sources for training and testing the model.

The model is here trained on FUMSECK data [add source]. The FUMSECK campaign was a campaign that took place in 2019 in the Ligurian Sea but far from the coast, contrary to the Endoume acquisitions that are conducted a few meters away from the coast. Once trained on this data 1, the model is used to predict samples taken at the Endoume station. The next figure presents a 2D cytogram of the Total Red Fluorescence (area under the curve) as a function of the Total Orange Fluorescence. This representation is often used by cytometrists to separate ... from ...

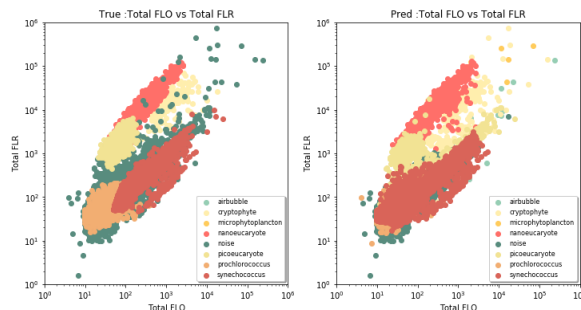


Figure 1: Manual vs. automated count

[Image to replace with the new one and then add interpretation]

5 Discussion

Our work aimed at providing a better understanding of the magnitude of manual classification biases. It confirms that less represented groups suffer more from these biases than the best represented ones [TO CHECK]. This highlights the need for data reviewed by several researchers/technicians in order to obtain good ground-truth data for model training. Such multi-reviewed datasets are more and more popular in the Machine Learning community, the best example being the ImageNet repository [6]. As a result we call for the creation of an equivalent repository

for cytometric curves PFT recognition.

This is all the more so necessary that the prediction error obtained by our network lies in the same error range as manual classification. Hence, better data may be even more useful than exploring better model specifications in order to achieve better performance.

Through this work we propose a full PFT prediction pipeline able to make quasi-real time PFT identification at the cell level. The total training time of our model is of less than a minute for a training set of nearly 45,000 observations on Google Collab GPU [give more details about machine hardware] in contrast to several hours or days of Residual Networks transfer learning on images as in González & al. (2019) [8]. The data pre-processing and in particular the interpolation of the curves is actually the slowest part of our automated pipeline (between 1 and 2 minutes) whereas the prediction themselves take only a few seconds. This is explained by the fact that the curves are for the moment interpolated in a sequential manner, observation per observation. More efficient methods have to be implemented to reduce this computational bottleneck.

Thanks to new software developments, the pipeline will soon be able to feed the predictions back into the cytometric software CytoClus © and enable the oceanographers to manually modify the automated classification boundaries of the PFTs that seem erroneous to them. In this respect, our pipeline could also be used as a turnkey pre-clustering tool made to speed up the manual clustering tasks.

Concerning the network itself improvements are possible. Our methodology is based on a "classify and count" approach which is strongly criticized by González & al. (2019) [8]. Indeed, our pipeline attributes each cell to a PFT and then count the number of cells in each PFT. González & al. (2019) [8] present a reformulation of the cells count problem, called "quantification problem" in the literature, and show that training a series of one-versus-all simple predictors on features extracted from the network is better suited for this task. This will be investigated in future research.

Finally, this work is a preliminary work in order to study the behaviour of the PFT dynamics on Endoume data. Indeed, with a FLR6 and FLR25 acquisitions every two hours, the data load is hardly treatable by a single person and needs to be automated. With our model in production, future research will dwell on the behaviour of the PFTs with respect to environmental variables such as temperature, salinity or nutrients.

6 Acknowledgments

Cytobuoy for software development of the DAL
Funders of the SSLAMM
ENS Thesis grant ?
ENDOUME DIVERS

References

- [1] Eustasio del Barrio et al. “optimalFlow: Optimal-transport approach to flow cytometry gating and population matching”. In: *arXiv preprint arXiv:1907.08006* (2019).
- [2] Gustavo EAPA Batista, Ronaldo C Prati, and Maria Carolina Monard. “A study of the behavior of several methods for balancing machine learning training data”. In: *ACM SIGKDD explorations newsletter* 6.1 (2004), pp. 20–29.
- [3] L Beaufort and D Dollfus. “Automatic recognition of coccoliths by dynamical neural networks”. In: *Marine Micropaleontology* 51.1-2 (2004), pp. 57–73.
- [4] Yin Cui et al. “Class-balanced loss based on effective number of samples”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 9268–9277.
- [5] Susanne Dunker. “Hidden Secrets Behind Dots: Improved Phytoplankton Taxonomic Resolution Using High-Throughput Imaging Flow Cytometry”. In: *Cytometry Part A* 95.8 (2019), pp. 854–868.
- [6] Li Fei-Fei. “ImageNet: crowdsourcing, benchmarking & other cool things”. In: *CMU VASC Seminar*. Vol. 16. 2010, pp. 18–25.
- [7] Christopher B Field et al. “Primary production of the biosphere: integrating terrestrial and oceanic components”. In: *science* 281.5374 (1998), pp. 237–240.
- [8] Pablo González et al. “Automatic plankton quantification using deep features”. In: *Journal of Plankton Research* 41.4 (2019), pp. 449–463.
- [9] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.
- [10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.
- [11] Tsung-Yi Lin et al. “Focal loss for dense object detection”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2980–2988.
- [12] Liyuan Liu et al. “On the variance of the adaptive learning rate and beyond”. In: *arXiv preprint arXiv:1908.03265* (2019).
- [13] Anthony Malkassian et al. “Functional analysis and classification of phytoplankton based on data from an automated flow cytometer”. In: *Cytometry part A* 79.4 (2011), pp. 263–275.
- [14] Sinno Jialin Pan and Qiang Yang. “A survey on transfer learning”. In: *IEEE Transactions on knowledge and data engineering* 22.10 (2009), pp. 1345–1359.
- [15] Martin Popel and Ondřej Bojar. “Training tips for the transformer model”. In: *The Prague Bulletin of Mathematical Linguistics* 110.1 (2018), pp. 43–70.
- [16] Jason Yosinski et al. “How transferable are features in deep neural networks?” In: *Advances in neural information processing systems*. 2014, pp. 3320–3328.
- [17] Michael Zhang et al. “Lookahead Optimizer: k steps forward, 1 step back”. In: *Advances in Neural Information Processing Systems*. 2019, pp. 9593–9604.

A Representations of an observation

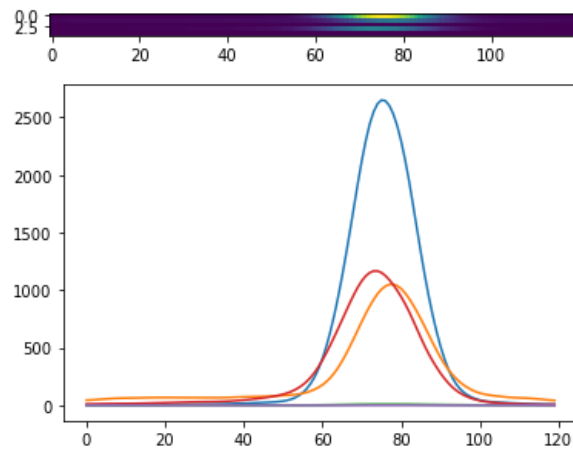


Figure 2: Matrix and curves representation of the five curves of an observation

B Model specification used

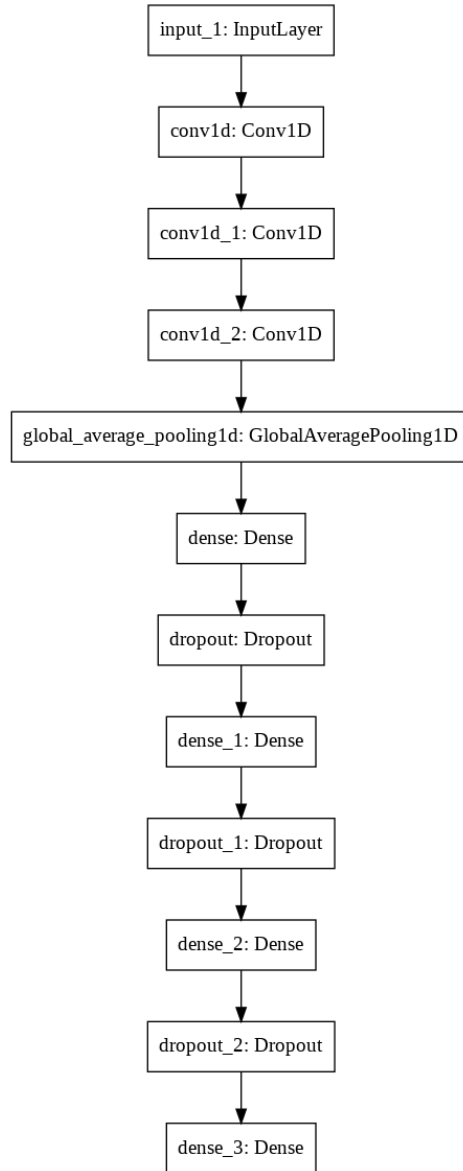


Figure 3: Model specification

C Decision boundaries learnt

D Hyperparameters chosen

This section presents the final architecture choice used for all the results presented below. The best performance was obtained for the focal loss and the Ranger optimizer. The following table presents the main hyperparameters of our model.

| Hyperparameters | α | batch size | Dropout | γ | Optimizer | Radam learning rate | Lookahead slow step size | Lookahead Sync period |
|-----------------|----------|------------|----------|----------|-----------|---------------------|--------------------------|-----------------------|
| Value | 6.980E-4 | 256 | 5.093E-3 | 2.046 | Ranger | 3.810E-3 | 2.074E-1 | 10 |

E Listmode features