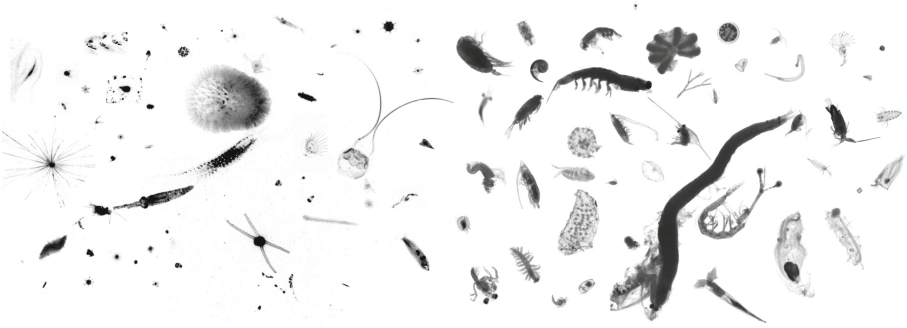
# Low-Shot Learning of Plankton Categories

Simon-Martin Schröder[1(✉)], Rainer Kiko[2], Jean-Olivier Irisson[3], and Reinhard Koch[1]

[1] Department of Computer Science, Kiel University, Kiel, Germany
{sms,rk}@informatik.uni-kiel.de
[2] GEOMAR Helmholtz-Centre for Ocean Research, Kiel, Germany
rkiko@geomar.de
[3] Sorbonne Université, CNRS, Laboratoire d'Océanographie de Villefanche, LOV, Villefranche-sur-mer, France
irisson@obs-vlfr.fr

**Abstract.** The size of current plankton image datasets renders manual classification virtually infeasible. The training of models for machine classification is complicated by the fact that a large number of classes consist of only a few examples. We employ the recently introduced weight imprinting technique in order to use the available training data to train accurate classifiers in absence of enough examples for some classes.

The model architecture used in this work succeeds in the identification of plankton using machine learning with its unique challenges, i.e. a limited number of training examples and a severely skewed class size distribution. Weight imprinting enables a neural network to recognize small classes immediately without re-training. This permits the mining of examples for novel classes.



(a) UVP5 (b) ZooScan

**Fig. 1.** Example images from both datasets.


T. Brox et al. (Eds.): GCPR 2018, LNCS 11269, pp. 391–404, 2019.
https://doi.org/10.1007/978-3-030-12939-2_27

## 1 Introduction

Planktonic organisms – drifters in the ocean – cover a large size range from nanometer-sized bacteria to meter-sized jellyfishes. While some of these organisms such as the planktonic copepods can be observed nearly everywhere, others occupy only small niches. Past observations allow an overview of the most abundant groups but we can expect that the number of classes will keep increasing with increasing sampling effort.

Current imaging systems (e.g. UVP5, ZooScan, ISIIS, FlowCytoBot [6,13,21,25]) that target the micro to macroplankton size range (approx. 10 µm to 10 cm) yield large amounts of image data every day. The size of the resulting datasets renders manual classification virtually infeasible. Therefore, accurate machine classification is a critical step in the processing of these data. Usually, the result is later verified by human experts. Even the annotation of pre-classified data is still labor-intensive [7,12], which is why maximally accurate models are crucial.

This work is part of a larger undertaking with the aim of continually monitoring newly acquired data for classes that have been overlooked so far. The observation of new kinds of objects means that the machine classification models need to be updated to incorporate these novel classes. In addition, plankton image datasets typically consist of few classes with many examples and many classes with only a few examples. A major problem is therefore the scarcity of training data for a large number of classes.

Here we tackle the question of how available labeled data can be used to train accurate machine classifiers when some class sizes in the training data set are very small, which is known as *low-shot learning*. We employ a recently presented method for low-shot learning called *weight imprinting* [27] that is able to incorporate new classes into a model without re-training it from scratch.

The contribution of this present paper is a rigorous evaluation of whether weight imprinting works satisfactorily for two plankton image datasets. We also examine the necessity of the architectural choices made in [27].

Our hypothesis is that once we have trained a classifier, we can use it to find more examples for underrepresented and novel classes within a large set of unlabeled data. In this current work, we therefore focus on the smaller classes instead of maximizing overall accuracy.

The remaining part of this paper is structured as follows. In Sect. 2 we introduce two plankton image datasets. Then we review the related work in Sect. 3. Section 4 reproduces the most important aspects of the weight imprinting technique. In Sect. 5 we apply weight imprinting to both plankton datasets. Subsequently, we report and discuss our results in Sect. 6 and draw a conclusion in Sect. 7.

## 2 Datasets

We evaluate the approach on two datasets extracted from the plankton image database EcoTaxa [24]. The objects were sampled on numerous cruises in many

parts of the world's oceans. The first dataset (UVP5) consists of 588,121 pelagic underwater images acquired with the UVP5 [25]. The images were sorted by experts into 65 classes. The dataset is available from the authors upon reasonable request. The second dataset (ZooScan) [10] consists of 1,433,282 wet net samples digitized with the ZooScan system [13] and sorted into 93 classes. We use a subset of 1,146,684 images for training and validation.

Both datasets are severely imbalanced, as shown in Fig. 2 for the UVP5 dataset. The 10% most populated classes contain more than 77% of all objects and the class sizes span multiple orders of magnitude. Figure 1 shows some exemplary objects from both datasets.

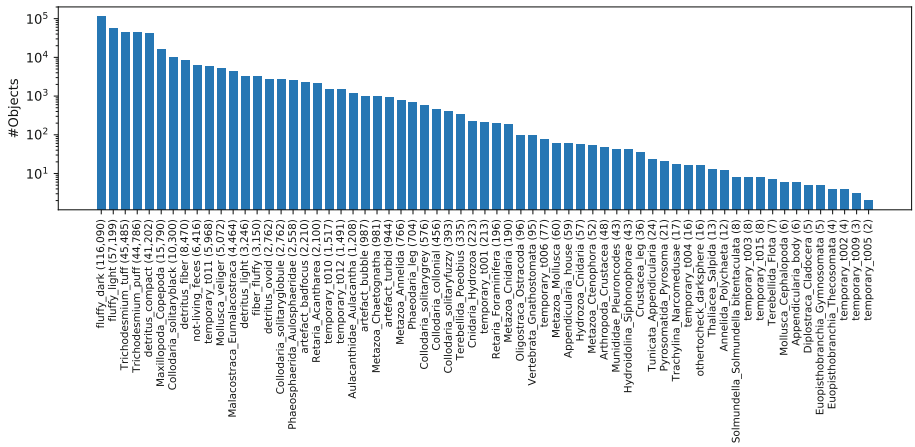



**Fig. 2.** UVP5 dataset: classes ordered by their size in the training set. The class sizes span five orders of magnitude.

## 3 Related Work

**One-Shot and Low-Shot Learning.** One-shot and low-shot learning is concerned with training a model with only one or a few training examples for each class.

Low-shot learning using neural networks usually incorporates two phases [15]. In the *representation learning* phase, the learner finds a suitable feature space, usually guided by a set of base classes with abundant examples. In the *low-shot learning* phase, a classifier is trained that incorporates both base and low-shot classes. Different approaches emphasize different aspects of the process [3]: the discriminative approach is concerned with learning powerful features, the generative approach enlarges the training set by augmentation or generation and the network structural approach utilizes new types of classifiers.

Weight imprinting [27], label diffusion [9], and metric learning [20] belong to the third category. They provide low-shot learning without having to retrain the whole model from scratch.

**Classification of Plankton Images.** Classification of plankton images is traditionally performed using shallow models, like Support Vector Machines or Random Forests, trained with handcrafted local features measured on the image (e.g. size, grey level distribution, etc.) [1,8,11,13,28].

Since Kaggle's National Data Science Bowl competition to sort data from ISIIS [6], there has been a slow transition towards deep models [4,14,18,22,26].

In the representation learning phase, we rely on the observations of [22] regarding the classification of plankton images with deep learning models, i.e. that the initialization with pre-trained weights outperforms random initialization.

## 4 Weight Imprinting

In this section, we outline the most important aspects of weight imprinting as introduced by [27].

The technique follows the two-phase paradigm of [15]: The set of all classes $C$ is partitioned into *base classes* $C_0$ with enough training data and the smaller *low-shot classes* $C_+$, i.e. $C = C_0 \cup C_+$.

In the representation learning phase, a convolutional neural network (CNN) is trained to distinguish the base classes with enough training data $C_0$. In the low-shot learning phase, the classifier is then updated with calculated weights (see Sect. 4.2 for details) to also to distinguish the smaller low-shot classes $C_+$. Finally, the whole model can be fine-tuned to further increase its predictive power.

### 4.1 Neural Network Model

The model consists of two stages: A feature extractor network $f : I \to \mathbb{R}^d$ maps an input image $x \in I$ to an $L_2$-normalized $d$-dimensional feature vector $\hat{y}$. The second stage is a modified softmax classifier $g : \mathbb{R}^d \to [0,1]^{|C|}$ that maps the feature activations to a discrete probability distribution of $|C|$ classes.

$$g_i(y) = \frac{\exp(s \cdot \hat{w}_i^\mathsf{T} \hat{y})}{\sum_{j \in C} \exp(s \cdot \hat{w}_j^\mathsf{T} \hat{y})} \tag{1}$$

$\hat{w}_i$ is the weight vector corresponding to class $i$ and is normalized to unit length as well. The scalar product $\hat{w}_i^\mathsf{T} \hat{y}$ is the angle or cosine similarity [19] between the feature vector and the weight vector. A weight vector $\hat{w}_i$ therefore acts as a template for class $i$. $s$ is a learnable scale factor that allows the probabilities to match the one-hot encoding of classes [29].

### 4.2 Low-Shot Learning

To learn a new class $c_+ \in C_+$, the weight matrix is extended by a column $w_+$. It follows from the above characterization of weight vectors $\hat{w}_i$ and image feature vectors $\hat{y}$ that they are interchangeable. Therefore, $w_+$ can be calculated directly from the feature vectors of the examples of class $c_+$. In the simplest case, if only one single training example $x_+$ belongs to $c_+$, the weight vector is equal to its feature vector, i.e. $w_+ = f(x_+)$. In the general case, if $c_+$ consists of multiple examples ($|c_+| \geq 1$), the weight vector is calculated as the arithmetic average of the image features, i.e. $w_+ = \frac{1}{|c_+|} \sum_{x_+ \in c_+} f(x_+)$, and renormalized. The scalar product $\hat{w}_+^\mathsf{T} \hat{y}$ then acts as a nearest mean classifier [20] using the cosine similarity. The underlying assumption is that the distribution of the examples is unimodal for each new class $c_+$.

## 5 Experiments

This section describes the experiments to evaluate the ability of weight imprinting to incorporate new classes into a plankton classification model without completely re-training from scratch.

### 5.1 Network Architecture

The model architecture is summarized in Table 1. The feature extractor network is based on the ResNet18 architecture, as it has a favorable accuracy-speed trade-off [2]. It is initialized with weights pre-trained on the ImageNet dataset [16]. The grayscale plankton images are converted to color images to fit the pre-trained model. We use 512-dimensional embeddings as we observed that the 64-dimensional embeddings from the original paper delayed the convergence of the training and did not bring an advantage. The last three layers implement the operations required for the weight imprinting as described in Sect. 4.1 and are initialized randomly.

**Table 1.** Network architecture.

| Layer | #Parameters | Output shape |
|---|---|---|
| Input | | $128 \times 128 \times 3$ |
| ResNet18 | 11 M | $4 \times 4 \times 512$ |
| Global Average Pooling | | 512 |
| Linear Layer (Embedding) | 262 k | 512 |
| Normalization | | 512 |
| Linear Layer $(\hat{w}_i)_{i \in C}$ | $512 \cdot \lvert C \rvert$ | $\lvert C \rvert$ |
| Scale | 1 | $\lvert C \rvert$ |

The whole model is fine-tuned to the task at hand, following the common practice [5]. Each experiment is carried out with three-fold cross-validation. To counter the class imbalance in the dataset we randomly subsample 1000 samples from the larger classes for each training epoch independently. Early stopping is used to avoid overfitting to the training split. To treat all classes equally, we weight the validation loss by the inverse class size. The initial learning rate is set to $1 \times 10^{-4}$ and decreased whenever the validation loss plateaus until it reaches $1 \times 10^{-8}$. The Adam algorithm [17], an extension to stochastic gradient descent, is used for parameter optimization. The batch size is set to 128 images. The images are cropped to their tight bounding box and padded to a square with a minimum edge length of 128 px. Images larger than 128 px are shrunken to this size. The grey values are scaled to the $[0, 1]$ range. We perform training-time augmentation using random rotations in 90° steps, random horizontal and vertical flips and Gaussian noise with $\sigma = 0.001$. The models are trained using the PyTorch deep learning library [23] on an NVIDIA GeForce GTX 1070 GPU.

### 5.2 Baselines

As a baseline (BL), we train the whole network on all classes $C = C_0 \cup C_+$ jointly and without altering the weights in a separate step. Larger classes are subsampled, smaller classes oversampled. We also conduct an ablation experiment to examine the necessity of the architectural choices made in [27] by comparing the original architecture to two modified versions where we removed the weight normalization (BL-W) and subsequently the feature normalization as well (BL-WF).

### 5.3 Representation Learning

In the first phase of model training, the model learns the base classes $C_0$. For the UVP5 dataset, we selected the classes with more than 1000 members in the training split (22 classes out of 65) as base classes. For the ZooScan dataset, the classes with more than 1500 members in the training split (44 classes out of 93) were selected as base classes.

The network is trained until the validation loss does not decrease for 25 epochs. No oversampling is performed because the minimum class size (1000 or 1500) is equal to or larger than the number of images needed per epoch (1000).

### 5.4 Low-Shot Learning

Now, the low-shot weights of the base network are set to their calculated values. For this purpose, we apply the feature extractor part of the network to every image. For each low-shot class $c_i \in C_+$, we calculate the mean feature vector, re-normalize it, and write it into the entry $\hat{w}_i$ of the weight matrix in the classifier part of the network corresponding to the respective class. We compare the described mean imprinting (MI) to random imprinting (RI), where $\hat{w}_i$ is initialized randomly.

### 5.5 Fine-Tuning

Finally, the imprinted networks are fine-tuned (MI+FT, RI+FT). Classes with less than 1000 examples are oversampled to counteract the class imbalance.

## 6 Results and Discussion

Tables 2 and 3 list the evaluation results for the UVP5 and ZooScan datasets, respectively. Both datasets behave comparably regarding the success of weight imprinting. We show the arithmetic average and standard deviation of three validation splits. Macro precision and macro recall are reported for base and low-shot classes separately. We do not report accuracy as it is mainly influenced by the base classes that contain most of the examples. Bold entries are global maxima, italic entries are maximal among their respective group.

Figures 3 and 4 show how precision and recall of individual classes depend on the size of the respective class. The vertical line divides low-shot classes (left) and base classes (right). To show the general tendency, the graphs contain a LOWESS fit for each set of classes. The figures compare MI, MI+FT, and BL. They are best viewed in color.

Table 4 lists the amount of time required to train a model to convergence in each condition, respectively.

**Table 2.** Macro precision and recall (UVP5).

| | Base classes | | Low-shot classes | |
|---|---|---|---|---|
| | Macro precision | Macro recall | Macro precision | Macro recall |
| MI | 0.569 ± 0.019 | 0.466 ± 0.051 | 0.030 ± 0.001 | 0.149 ± 0.023 |
| MI+FT | *0.581 ± 0.001* | *0.644 ± 0.002* | *0.121 ± 0.035* | *0.296 ± 0.035* |
| RI | 0.538 ± 0.024 | *0.675 ± 0.015* | | 0 |
| RI+FT | **0.593 ± 0.008** | 0.642 ± 0.005 | *0.163 ± 0.026* | *0.299 ± 0.022* |
| BL | *0.539 ± 0.004* | **0.689 ± 0.003** | **0.366 ± 0.049** | 0.265 ± 0.022 |
| BL-W | 0.489 ± 0.006 | 0.634 ± 0.015 | 0.212 ± 0.033 | 0.289 ± 0.014 |
| BL-WF | 0.487 ± 0.017 | 0.555 ± 0.087 | 0.157 ± 0.010 | **0.302 ± 0.006** |

### 6.1 Low-Shot Learning

For both datasets, mean imprinting (MI) leads to a better precision for the base classes than random imprinting (RI) and to a nonzero recall and precision of the low-shot classes. However, recall for the base classes is impaired. This was expected because as the low-shot classes are initialized with plausible locations in the feature space, they will draw off objects from the base classes.

**Table 3.** Macro precision and recall with standard deviation (ZooScan).

| | Base classes | | Low-shot classes | |
|---|---|---|---|---|
| | Macro precision | Macro recall | Macro precision | Macro recall |
| MI | 0.701 ± 0.001 | 0.716 ± 0.006 | 0.066 ± 0.008 | 0.205 ± 0.010 |
| MI+FT | *0.724 ± 0.005* | *0.771 ± 0.002* | *0.174 ± 0.010* | *0.650 ± 0.010* |
| RI | 0.663 ± 0.006 | **0.823 ± 0.003** | | 0 |
| RI+FT | **0.731 ± 0.005** | 0.775 ± 0.003 | *0.187 ± 0.003* | *0.664 ± 0.003* |
| BL | *0.627 ± 0.008* | *0.812 ± 0.005* | **0.459 ± 0.019** | 0.614 ± 0.002 |
| BL-W | 0.544 ± 0.010 | 0.793 ± 0.007 | 0.350 ± 0.025 | 0.675 ± 0.021 |
| BL-WF | 0.502 ± 0.019 | 0.747 ± 0.028 | 0.242 ± 0.013 | **0.709 ± 0.012** |

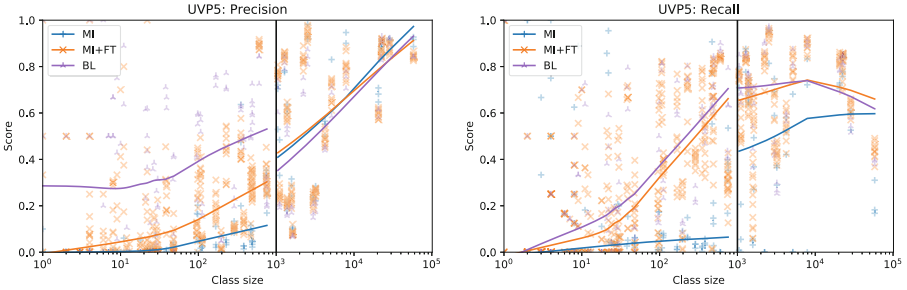



**Fig. 3.** UVP5: Comparison of per-class precision and recall for mean imprinting (MI), fine-tuning (MI+FT), and training from scratch (BL). (Color figure online)

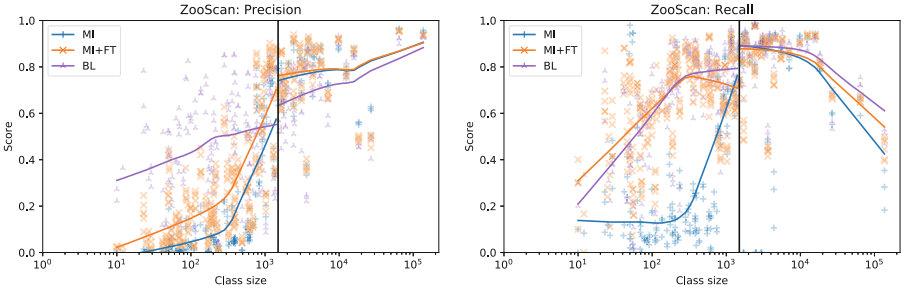



**Fig. 4.** ZooScan: Comparison of per-class precision and recall for mean imprinting (MI), fine-tuning (MI+FT), and training from scratch (BL). (Color figure online)

**Table 4.** Training times (s).

| | UVP5 | ZooScan |
|---|---|---|
| MI | 0.476 ± 0.026 | 2.120 ± 0.500 |
| MI+FT | 6588.105 ± 989.882 | 10 550.683 ± 961.637 |
| BL | 10 261.286 ± 425.061 | 22 182.181 ± 665.596 |

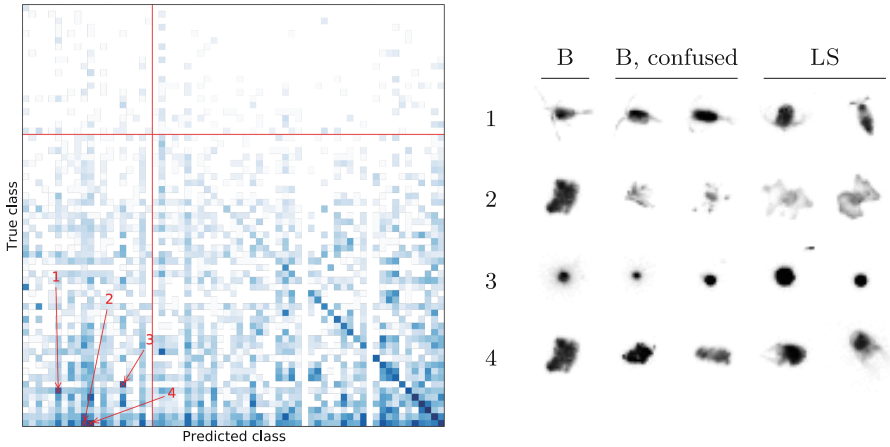



**Fig. 5.** Mean imprinting: confusion matrix and failure cases (UVP5, validation set). Four of the entries with the most objects from a base class falsely predicted as a low-shot class are numbered. Representative images of the true base class (B) and the predicted low-shot class (LS), as well as two confused images (B falsely predicted as LS) are shown for each entry. 1: Copepoda vs. Cladocera, 2: fluffy dark vs. t003, 3: Collodaria solitary black vs. other dark sphere, 4: fluffy dark vs. t015. *See the electronical version for a magnified view.* (Color figure online)

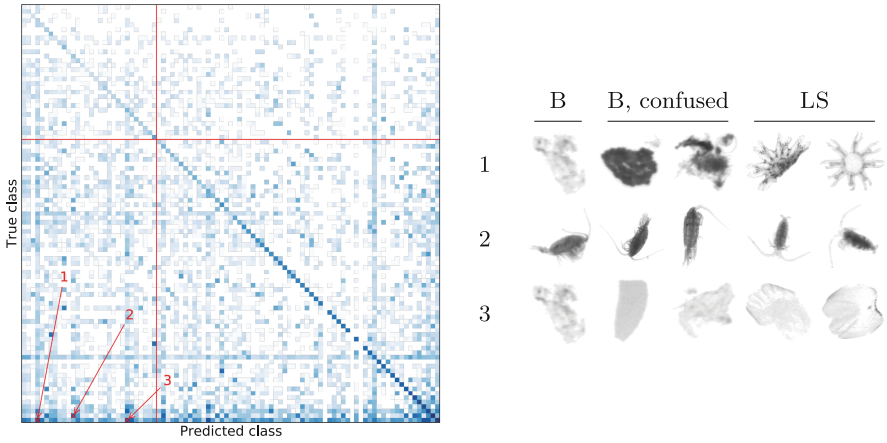



**Fig. 6.** Mean imprinting: confusion matrix and failure cases (ZooScan, validation set). Three of the entries with the most objects from a base class falsely predicted as a low-shot class are numbered. Representative images of the true base class (B) and the predicted low-shot class (LS), as well as two confused images (B falsely predicted as LS) are shown for each entry. 1: Detritus vs. Ephyra, 2: Calanoida vs. Calocalanus pavo, 3: Detritus vs. scale. *See the electronical version for a magnified view.* (Color figure online)

As apparent in Table 4, the time to calculate the weights is negligible. This is a huge advantage of weight-imprinting compared to other training methods when speed is important, e.g. when a user iteratively trains a model.

Figures 5 and 6 show the confusion matrices for both datasets averaged over all three splits and examples images for the three or four most often misclassifications of existing classes as being novel. The entries of the matrices are ordered by ascending class size and the low-shot classes are separated by red lines.

In the UVP5 dataset, images of a few of the more abundant base classes contaminate the predictions of low-shot classes (lower left rectangle in Fig. 5). In the ZooScan this behavior is less pronounced and the confusion matrix has a stronger diagonal and looks more uniform (Fig. 6), i.e. the model is more accurate.

The images representative of true and falsely predicted classes are very similar in most cases. In fact, several of the smaller classes were defined only recently and accordingly, many of their belonging images were assigned to more general classes in earlier days. We therefore suspect that some of the "errors" made by the models originate from an incoherent labeling of the data. Another type of error is the confusion of classes that differ only by a minuscule feature, for example the missing tail of the Cladocera in Fig. 5 or the regular shape of the Ephyra in Fig. 6.

### 6.2 Fine-Tuning

Fine-tuning (MI+FT) always improves over mean imprinting alone (MI). This is in agreement with [27]. However, fine-tuning a mean imprinted model does not yield better results than fine-tuning a randomly initialized model (RI+FT). In fact, both perform in a very similar way, while most scores are slightly higher for random initialization. Here, our results differ from [27]. The reason is presumably overfitting of the weight imprinted models to too few samples in the training split. Finetuning a mean imprinted model roughly halves the training time compared to the baseline.

### 6.3 Baselines

When comparing the baseline models amongst each other, it is apparent that the best scores are achieved with weight normalization and feature normalization (BL). Taking away weight normalization (BL-W) and subsequently feature normalization (BL-WF), the scores get worse. One exception is the recall of the low-shot classes which is maximal for minimal restrictions (BL-WF). We assume that feature and weight normalization have a regularizing effect that leads to a better generalization of the model.

Regarding the low-shot classes, a model trained from scratch (BL) leads to a significantly higher precision and recall than an imprinted model (MI) and to a better or equal recall than a fine-tuned model (MI+FT).

For the base classes, all three conditions behave roughly similarly, although the precision of BL is slightly worse than of MI and MI+FT. This can be explained

by the fact that during the representation learning phase of MI and MI+FT, the models learn features that are explicitly adapted to the base classes. Fine-tuning does not seem to destroy this advantage. Conversely, when training on all classes jointly (BL), the features are adapted to all classes equally (and not primarily to the base classes). This also explains the better scores of BL concerning the low-shot classes.

The negative slope of the recall of the base classes in the ZooScan dataset can be explained by the decreasing sampling rate of the base classes, where down to 1% (for the largest classes) of the available examples are used in each training epoch. Therefore, the model may not be able to capture the full variability of the data.

Training a model from scratch takes roughly twice as long as finetuning a mean imprinted model.

### 6.4 Dataset Maintenance

When collecting plankton images, it is important to constantly monitor the newly acquired data for previously overlooked classes and to update the models accordingly. When a dataset is extended by a novel class, more examples of this novel class have to be found among a large number of images. These examples can be found in new unlabeled data and may also hide in existing labeled data, as in some cases they might have been assigned to a more general class before the introduction of the novel class.

Figures 3 and 4 show how precision and recall are linked to the class size for all models (BL, MI, and MI+FT). Although the imprinting of weights (MI) does not achieve the best scores, its huge advantage is the instant integration of novel data.

Correspondingly, the model may serve as a tool to quickly build up a collection of images in order to train a more powerful classifier later. As weight imprinting allows instant (i.e. in less than one second) re-training for every additional training example without much overhead (see Sect. 4.2), it can be used during a process of iterative example mining: The model is initialized using one or few examples of a class. It is then used to classify unlabeled data and a human operator validates these classifications. The positive examples are used to improve the classifier. In this way, the suggestions will get increasingly better in every iteration.

For the suggestions to be useful, they have to contain positive examples at all, which manifests in a non-zero recall. In the ZooScan dataset, the recall is stable at an average of approx. 0.2 for even the smallest classes. In the UVP5 dataset, the recall is low for most classes. However, there are notable outliers with a recall of up to 1.0. Further research is needed to figure out what makes these outliers special.

As a result, weight imprinting is useful in the mining of additional examples for small and novel classes in both plankton image datasets. The actual implementation of the described procedure will be a task for the future.

The incoherent labeling of novel low-shot classes and existing base classes apparent in the evaluation underlines the usefulness of weight imprinting to keep a dataset consistently labeled when the set of classes evolves.

## 7 Conclusion

Deep learning models often fail when it comes to training classifiers with very little training data. This is the case for a considerable number of classes in both plankton image datasets. We therefore employed *weight imprinting* [27], a low-shot learning method. When applying this method to plankton image datasets, it allows a model to incorporate underrepresented classes without overly impairing the performance on the base classes.

Weight imprinting resulted in an acceptable recall for the low-shot classes immediately without re-training the neural network. This permits the extension of a training set by examples of a novel class as soon as a single example of this class is observed.

Even when training from scratch, the underlying network architecture performs better than a model without normalization of features and weights in most cases. Apart from a reduced training time, we observed no significant advantage of an imprinted and fine-tuned model over a model trained from scratch.

In summary, the techniques in question, i.e. normalization of features and weights and weight imprinting, are able to advance the identification of plankton using machine learning with its unique challenges, i.e. a limited number of training examples, a severely skewed class size distribution and ever-emerging novel classes.

**Acknowledgements.** Rainer Kiko was supported by the Deutsche Forschungsgemeinschaft (DFG) as part of the Collaborative Research Centre (SFB) 754 "Climate-Biogeochemistry Interactions in the Tropical Ocean." Rainer Kiko, Reinhard Koch and Simon-Martin Schröder were furthermore supported by grants CP1650 and CP1733 of the Cluster of Excellence 80 "The Future Ocean." "The Future Ocean" is funded within the framework of the Excellence Initiative by the Deutsche Forschungsgemeinschaft (DFG) on behalf of the German federal and state governments. Jean-Olivier Irisson was supported by CNRS LEFE-MANU through project DL-PIC.

## References

1. Blaschko, M.B., et al.: Automatic in situ identification of plankton. In: 2005 Seventh IEEE Workshops on Applications of Computer Vision (WACV/MOTION 2005), vol. 1, pp. 79–86. IEEE (2005)
2. Canziani, A., Paszke, A., Culurciello, E.: An Analysis of Deep Neural Network Models for Practical Applications (2016). http://arxiv.org/abs/1605.07678
3. Choe, J., Park, S., Kim, K., Park, J.H., Kim, D., Shim, H.: Face generation for low-shot learning using generative adversarial networks. In: ICCVW 2017, pp. 1940–1948 (2017)

4. Christiansen, S., et al.: Particulate matter flux interception in oceanic mesoscale eddies by the polychaete Poeobius sp. Limnol. Oceanogr. **63**, 2093–2109 (2018)
5. Chu, B., Madhavan, V., Beijbom, O., Hoffman, J., Darrell, T.: Best practices for fine-tuning visual classifiers to new domains. In: Hua, G., Jégou, H. (eds.) ECCV 2016. LNCS, vol. 9915, pp. 435–442. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-49409-8_34
6. Cowen, R.K., Guigand, C.M.: In situ ichthyoplankton imaging system (ISIIS): system design and preliminary results. Limnol. Oceanogr.: Methods **6**, 126–132 (2008)
7. Culverhouse, P.F., Macleod, N., Williams, R., Benfield, M.C., Lopes, R.M., Picheral, M.: An empirical assessment of the consistency of taxonomic identifications. Marine Biol. Res. **10**, 73–84 (2014)
8. Culverhouse, P.F., et al.: Automatic classification of field-collected dinoflagellates by artificial neural network. Marine Ecol. Progress Ser. **139**(1/3), 281–287 (1996)
9. Douze, M., Szlam, A., Hariharan, B., Jégou, H.: Low-shot learning with large-scale diffusion (2017). https://arxiv.org/pdf/1706.02332.pdf
10. Elineau, A., et al.: ZooScanNet: plankton images captured with the ZooScan (2018). http://doi.org/10.17882/55741
11. Ellen, J., Li, H., Ohman, M.D.: Quantifying California current plankton samples with efficient machine learning techniques. In: OCEANS 2015 - MTS/IEEE Washington, pp. 1–9. IEEE (2015)
12. Faillettaz, R., Picheral, M., Luo, J.Y., Guigand, C., Cowen, R.K., Irisson, J.O.: Imperfect automatic image classification successfully describes plankton distribution patterns. Methods Oceanogr. **15–16**, 60–77 (2016)
13. Gorsky, G.: Digital zooplankton image analysis using the ZooScan integrated system. J. Plankton Res. **32**(3), 285–303 (2010)
14. Graham, B., van der Maaten, L.: Submanifold Sparse Convolutional Networks (2017). http://arxiv.org/abs/1706.01307
15. Hariharan, B., Girshick, R.: Low-shot visual recognition by shrinking and hallucinating features. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 3037–3046. IEEE (2017)
16. Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Fei-Fei, L.: ImageNet: a large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255. IEEE (2009)
17. Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization (2014). http://arxiv.org/abs/1412.6980
18. Lee, H., Park, M., Kim, J.: Plankton classification on imbalanced large scale database via convolutional neural networks with transfer learning. In: 2016 IEEE International Conference on Image Processing (ICIP), pp. 3713–3717. IEEE (2016)
19. Luo, C., Zhan, J., Wang, L., Yang, Q.: Cosine Normalization: Using Cosine Similarity Instead of Dot Product in Neural Networks (2017). http://arxiv.org/abs/1702.05870
20. Mensink, T., Verbeek, J., Perronnin, F., Csurka, G.: Distance-based image classification: generalizing to new classes at near-zero cost. IEEE Trans. Pattern Anal. Mach. Intell. **35**(11), 2624–2637 (2013)
21. Olson, R.J., Sosik, H.M.: A submersible imaging-in-flow instrument to analyze nano-and microplankton: Imaging FlowCytobot. Limnol. Oceanogr.: Methods **5**(6), 195–203 (2007)
22. Orenstein, E.C., Beijbom, O.: Transfer learning and deep feature extraction for planktonic image data sets. In: 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 1082–1088. IEEE (2017)

23. Paszke, A., et al.: Automatic differentiation in PyTorch. In: Advances in Neural Information Processing Systems (NIPS), vol. 30, pp. 1–4 (2017)
24. Picheral, M., Colin, S., Irisson, J.O.: EcoTaxa (2017). http://ecotaxa.obs-vlfr.fr/
25. Picheral, M., Guidi, L., Stemmann, L., Karl, D.M., Iddaoud, G., Gorsky, G.: The Underwater Vision Profiler 5: an advanced instrument for high spatial resolution studies of particle size spectra and zooplankton. Limnol. Oceanogr.: Methods **8**(1), 462–473 (2010)
26. Py, O., Hong, H., Zhongzhi, S.: Plankton classification with deep convolutional neural networks. In: 2016 IEEE Information Technology, Networking, Electronic and Automation Control Conference, pp. 132–136 (2016)
27. Qi, H., Brown, M., Lowe, D.G.: Low-shot learning with imprinted weights. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5822–5830 (2018)
28. Sosik, H.M., Olson, R.J.: Automated taxonomic classification of phytoplankton sampled with imaging-in-flow cytometry. Limnol. Oceanogr.: Methods **5**(6), 204–216 (2007)
29. Wang, F., Xiang, X., Cheng, J., Yuille, A.L.: NormFace: $L_2$ hypersphere embedding for face verification. In: Proceedings of the 2017 ACM on Multimedia Conference, MM 2017, pp. 1041–1049. ACM Press, New York (2017)