Title

Improving plankton image classification using context metadata

Journal

Limnology and Oceanography Methods, 17(8)

ISSN

1541-5856

Authors

Ellen, Jeffrey S

Graff, Casey A

Ohman, Mark D

Publication Date

2019-08-01

DOI

10.1002/lom3.10324

Peer reviewed

LIMNOLOGY
and
OCEANOGRAPHY: METHODS

ASLO

Limnol. Oceanogr.: Methods 17, 2019, 439–461

doi: 10.1002/lom3.10324

Improving plankton image classification using context metadata

Jeffrey S. Ellen [1]*, **Casey A. Graff,** [1a] **Mark D. Ohman** [2]

[1]Department of Computer Science and Engineering, University of California San Diego, La Jolla, California
[2]Scripps Institution of Oceanography, University of California San Diego, La Jolla, California

Abstract

Advances in both hardware and software are enabling rapid proliferation of in situ plankton imaging methods, requiring more effective machine learning approaches to image classification. Deep Learning methods, such as convolutional neural networks (CNNs), show marked improvement over traditional feature-based supervised machine learning algorithms, but require careful optimization of hyperparameters and adequate training sets. Here, we document some best practices in applying CNNs to zooplankton and marine snow images and note where our results differ from contemporary Deep Learning findings in other domains. We boost the performance of CNN classifiers by incorporating metadata of different types and illustrate how to assimilate metadata beyond simple concatenation. We utilize both geotemporal (e.g., sample depth, location, time of day) and hydrographic (e.g., temperature, salinity, chlorophyll *a*) metadata and show that either type by itself, or both combined, can substantially reduce error rates. Incorporation of context metadata also boosts performance of the feature-based classifiers we evaluated: Random Forest, Extremely Randomized Trees, Gradient Boosted Classifier, Support Vector Machines, and Multilayer Perceptron. For our assessments, we use an original data set of 350,000 in situ images (roughly 50% marine snow and 50% non-snow sorted into 26 categories) from a novel in situ *Zooglider*. We document asymptotically increasing performance with more computationally intensive techniques, such as substantially deeper networks and artificially augmented data sets. Our best model achieves 92.3% accuracy with our 27-class data set. We provide guidance for further refinements that are likely to provide additional gains in classifier accuracy.

The burgeoning number of digital imaging methods available to aquatic ecologists, both in situ (Davis et al. 1992; Samson et al. 2001; Benfield et al. 2003; Watson 2004; Olson and Sosik 2007; Cowen and Guigland 2008; Picheral et al. 2010; Schulz et al. 2010; Thompson et al. 2012; Briseño-Avena et al. 2015; Ohman et al. 2018) and in the laboratory (Sieracki et al. 1998; Gorsky et al. 2010), is generating rapidly expanding libraries of digital images useful in a variety of scientific applications. However, the accumulation of large numbers of images increases the need for much more efficient machine learning methods in order to automate the processes of image classification, data extraction, and analysis.

Until recently, most automated image classification has employed methods we refer to as "feature-based," in that they operate on a set of descriptive geometric features calculated from the digital images, such as area, shape, aspect ratio, fractal dimension, textures, and grayscale histograms (e.g. Peura and Iivarinen 1997). The feature-based algorithms then derive a mapping from the calculated values to labels corresponding to the type of organism. Ideally, this mapping will extrapolate to future images. Some of the feature-based algorithms that have been applied to classification of plankton images with varying degrees of success include Random Forest (Grosjean et al. 2004; Gorsky et al. 2010), support vector machines (SVMs) (Hu and Davis 2005; Sosik and Olson 2007; Ellen et al. 2015), and multilayer perceptron (MLP) (Wilkins et al. 1996), among others.

Since 2012, "Deep Learning" algorithms (Krizhevsky et al. 2012; LeCun and Ranzato 2013; LeCun et al. 2015) have outperformed feature-based classifiers in a variety of fields, including natural language processing (Socher et al. 2013), time series analysis (Graves et al. 2013), variational autoencoders (algorithms that learn to generate or alter existing data, such as image correction; Kingma and Welling 2013), plankton image analysis (Orenstein et al. 2015; Dai et al. 2016; Dieleman et al. 2016*b*; Graff and Ellen 2016; Wang et al. 2016; Zheng et al. 2017; Orenstein and Beijbom 2017; Luo et al. 2018), et al. Multiple algorithms have been characterized as examples of Deep Learning, the commonality being the use of repetitive layers of algorithmic structure that operate on the prior layers rather than the

*Correspondence: jellen@ucsd.edu

[a]Present address: Department of Computer Science, University of California, Irvine, Irvine, California

original input. Deep Learning algorithms tend to require orders of magnitude more computation, although often such computations are highly parallelizable and can be done rapidly given appropriate hardware. Among the most commonly adopted Deep Learning methods are convolutional neural networks (CNNs). CNNs have been applied to a spectrum of image recognition problems (e.g., LeCun and Bengio 1995; LeCun et al. 1998; Matsugu et al. 2003; Ng et al. 2015; Esteva et al. 2017). Applications of CNN and Random Forest to phytoplankton image classification include Orenstein et al. (2015), while further applications of CNNs to coral, plankton, and fish classification are surveyed by Moniruzzaman et al. (2017). A detailed end-to-end workflow utilizing CNNs to classify large numbers of plankton images is described in Luo et al. (2018), and the results they provide for their selected CNN architecture clearly illustrate that CNNs can be used to classify plankton images.

CNNs obviate the need for explicit geometric image measurements to be defined and generated, and instead operate directly on the two-dimensional image contents. When a human examines an image captured with discrete pixels such as Fig. 1a, the Gestalt theory of perceptual grouping states that we do not primarily perceive individual dots of colored ink or light, but instead comprehend unified shapes in relation to complete objects, such as in Fig. 1b (Wertheimer 1923). This recognition may consist of simplistic objects such as "tunic," "stomach," and "salp," or more specific objects based on the viewer's expertise, such as "circumferential muscle bands" or "endostyle" (Wagemans et al. 2012).

A computer's perception is entirely different, lacking these higher level taxonomic or morphometric concepts. Computer "vision" is limited to a grid of integer values (Fig. 1c) and concepts such as "42 dark gray pixel values" or "123 contiguous nonzero pixels." Feature-based methods use summary statistics such as perimeter or mean intensity to describe the image or object. In contrast, CNNs generate independent statistics for a lattice of sections of the original image, and repeat this process at multiple scales to build a statistical summary of the entire image contents.

CNNs apply a system of hierarchical filters to the grid of pixels in a manner inspired by Hubel and Wiesel's investigation of receptive fields within the visual cortex (Hubel 1959; Hubel and Wiesel 1963). The lowest layer of the CNN consists of a set of filters as in Fig. 2a. These filters are initialized by either generating random values or adopting a set of filters from a previously trained CNN. The filters are then convolved against the input image, i.e., performing element-wise multiplication between the filter and the region of the image that it covers for every possible region in the image. Every filter's convolution is input for a neuron, which sums these inputs, and applies a nonlinear activation function that produces higher valued output when the match between the filter and input region's high values are closely correlated (Fig. 2b). The neuron's output is used as input for the next layer of filters. Each subsequent layer of filters is similarly applied to its predecessor. During the training phase, as labeled images are assessed, the algorithm gradually adjusts these filters so that they are the most useful for determining differences between classes. Early layers of filters usually evolve to identify low-level

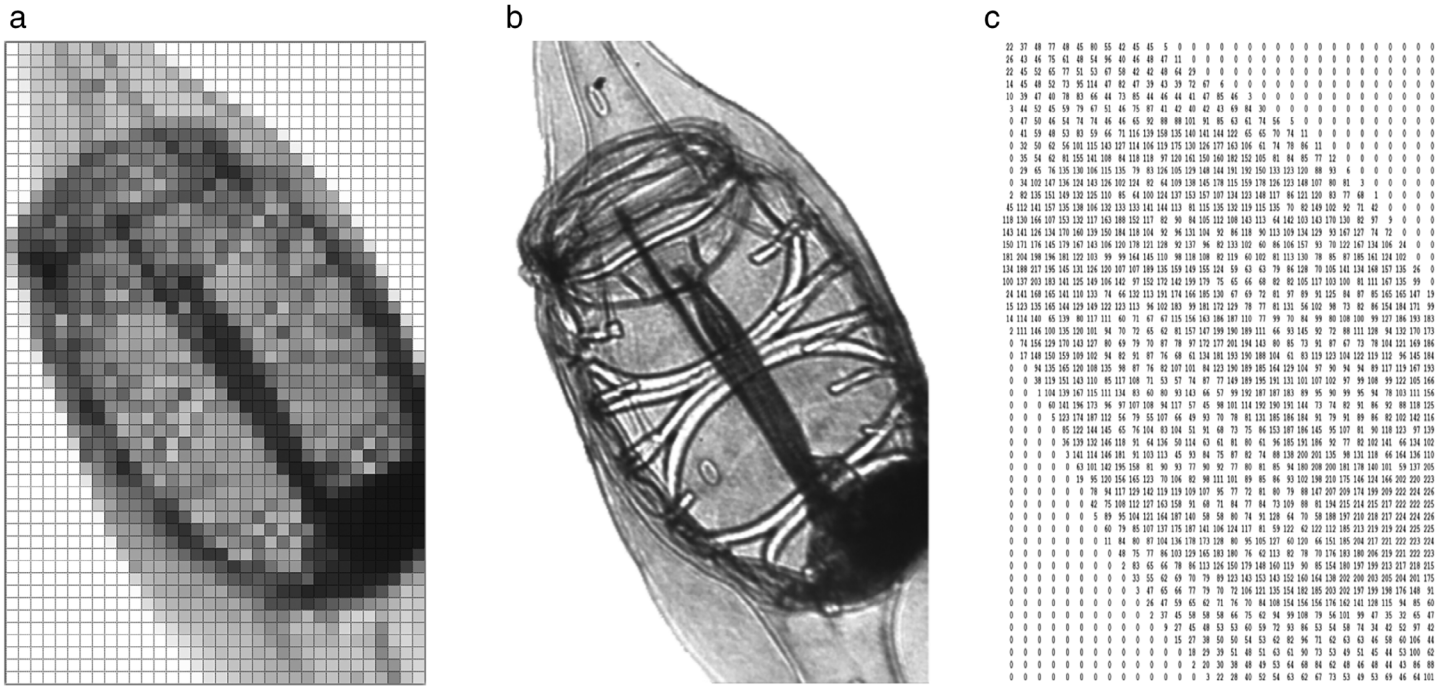

Fig. 1. Multiple renderings of a salp zooid (**a**) at low resolution (**b**) at full resolution typical of *Zooglider*, which a human generally perceives as contiguous, unified shapes, and (**c**) a numerical representation of the intensity values in (**a**).

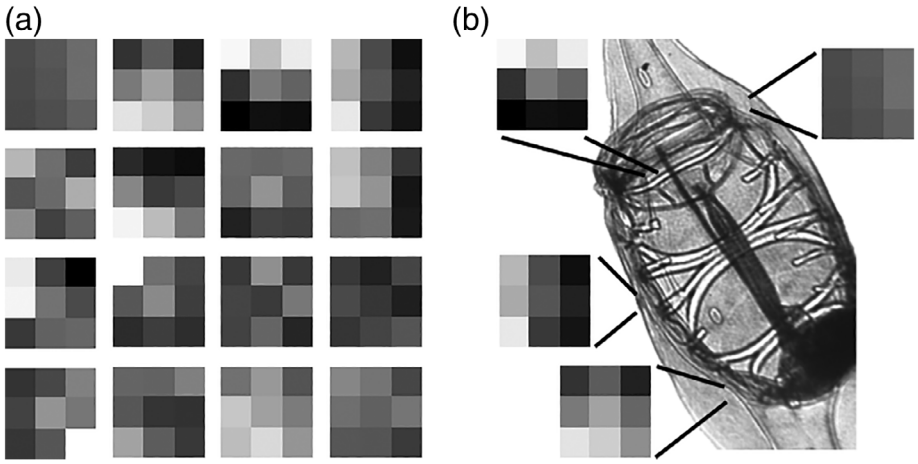

Fig. 2. Conceptual application of filters to an input image as in the first layer of a CNN. (**a**) A bank of 3×3 filters. (**b**) Conceptual representation of regions where a particular filter from (**a**) would have a strong response to the salp input image, e.g., a sharp horizontal edge at the top of a muscle band, or a dark-to-light gradient mid-tunic.

visual concepts such as colors (or in our case, shades of gray), corners, and edges at a particular orientation as in the example in Fig. 2. Secondary filters typically correspond to mid-level concepts such as curves and textures, potentially equating to muscle bands or outer tunic. Additional layers of filters evolve against their predecessors' output, ideally resulting in high level objects such as peripharyngeal band or testes that are useful for determining the final classification label.

Although CNNs and feature-based methods operate on different representations of the image data, a limitation of both approaches is that they utilize only the information contained in the image. In contrast, human taxonomists consider the context in which the sample was acquired when making identifications. For planktonic organisms, collection information such as geographic location, season, depth, time of day, and hydrographic conditions provide context metadata that may help constrain the realm of plausible answers and facilitate the identification process. The concept of utilizing metadata to improve image classification has been explored in other domains. One early work on classifying tourism photography used GPS information in conjunction with the images to improve identifying landmarks (Li et al. 2009). Other work incorporated GPS information to generate metadata such as elevation, average vegetation, and congressional district and explored two different ways of incorporating the metadata to achieve a 5-point gain in accuracy on a 100-way classification task of common objects and scenes (Tang et al. 2015). While incorporating context metadata into feature-based classifications is straightforward, it is more challenging to include such metadata into CNNs.

In this article, we assess whether incorporation of different types of context metadata improves classification accuracy for both CNNs and feature-based methods. Our numerical experiments are based on an original library of validated images from *Zooglider* (Ohman et al. 2018), a novel in situ zooplankton imaging device. We will illustrate how to optimize the use of metadata. In addition, although machine-learning methods involve many parameter values that can markedly affect the efficacy of a classifier, many practitioners simply adopt default values in commonly available software packages. We illustrate the benefits of tuning hyperparameters for both CNNs and five of the most common feature-based methods, and provide guidance for selecting hyperparameter values (where a hyperparameter is an overarching parameter whose value is chosen before the learning algorithm optimizes the model's parameters). We assess the performance of feature-based algorithms against CNNs of varying size and complexity, and quantify the benefit of including metadata.

Materials and procedures

Machine learning algorithms and image processing software

In addition to CNNs, we used five feature-based algorithms: Random Forest Classifier (RFC), Extremely Randomized Trees (XRT), Gradient Boosted Classifier (GBC), MLP, and SVM.

The Random Forest algorithm constructs an optimal decision tree by fitting it to a bootstrap sample drawn from the training set. Once that tree is optimized, more trees are constructed up to a threshold (Ho 1995). We also used two more recent modifications of RFC. The XRT algorithm uses stochastic partitions of the data instead of all data, and stochastic tree construction conditions instead of fully optimizing each tree (Geurts et al. 2006). These modifications usually cause faster algorithm convergence while producing similar or better results (Criminisi et al. 2012). The other RFC variation we use, GBC, draws on the concept of boosting, where a collection of weak models can be combined into a stronger one (Freund and Schapire 1997), in this case more abbreviated decision "stumps" instead of full trees (Friedman 2001). We also assess SVMs, which construct a decision boundary

that optimally divides the space between all the samples based on their overall proximity to each other in the metric space (Cortes and Vapnik 1995), rather than directly operating on sampled values of individual features, as in RFC. Finally, we assess MLP (Rumelhart et al. 1986), where each neuron produces a flat subset within the decision space, and by learning these flat subsets collectively forms a complex decision surface (Haykin 2009) that is extremely flexible (Lippmann 1987).

We used the Python programming language (van Rossum 1995) for high-level data handling and general computation. We used OpenCV (Bradski 2000) for image processing and manipulation. For RFC, XRT, GBC, and SVM, we used Scikit-Learn (Pedregosa et al. 2011). For MLP and CNN, we used the Lasagne library (Dieleman et al. 2016*a*) to specify our models, which were then executed by the Theano Framework (Al-Rfou et al. 2016). Alternative CNN implementations are available in TensorFlow, Caffe, and Torch, among others.

Computational equipment

We performed smaller numerical experiments on a simple server with 40 CPU cores, 128 GB of RAM, and an NVIDIA K40 GPU. For larger scale experiments, we utilized NSF's Extreme Science and Engineering Discovery Environment (XSEDE.org; Towns et al. 2014) which provided access to dozens of Graphics Processing Units (GPUs) simultaneously via their nationwide supercomputing resources. While each individual model we evaluated can be assessed on a single GPU, using the computational resources of XSEDE allowed us to more thoroughly and efficiently conduct experiments, which consisted of many thousands of trials and replicates.

Table 1. Distribution of the 350,000 ROI in our largest data set. Example illustrations of each of the 27 classes are provided in Fig. 3.

Class name	Quantity	Class name	Quantity
1. Acantharians (Sun-like)	4418	15. Fish larvae	789
2. Acantharians	1659	16. Foraminifera	320
3. Appendicularians	14,250	17. Narcomedusae	673
4. Chaetognaths	2170	18. Overturns (nonbiological)	8734
5. Cnidarians	2358	19. Phaeodarians	1159
6. Comets	1151	20. Quasi-spheres	5387
7. Copepods	44,662	21. Spheres (egg-like)	1345
8. Nauplii (copepods)	371	22. Threads	8043
9. Dense_background (biological)	2272	23. Marine snow	178,547
10. Diatoms high concentrations	4899	24. Tentacles with white streaks	1935
11. Diatoms (no spines)	20,088	25. Spheres (white)	1627
12. Diatoms (with spines)	10,115	26. Tentacles	28,984
13. Disks	2150	27. Translucent spheres	981
14. Euphausiids	913		

Image acquisition

Our images were acquired by *Zooglider*, an autonomous vehicle with a Zoocam bearing a telecentric lens system that enables in situ imaging of planktonic organisms and particles in a volume of ~ 250 mL per frame (Ohman et al. 2018). *Zooglider* operates from 400 to 0 m depth and acquires black and white silhouette images at a frequency of 2 Hz during ascent, creating a spatially resolved sequence of images with ~ 5 cm vertical resolution, each imaging an independent volume of water. *Zooglider* also measures conductivity, temperature, depth, and chlorophyll *a* (Chl *a*) fluorescence, and has a dual frequency Zonar (200/1000 kHz) intended to measure acoustic backscatter from objects approximately the same size as those imaged by the Zoocam (0.5–50 mm). We performed image correction of Zoocam image frames, including de-noising and gamma correction, to improve contrast, as described in more detail in Ohman et al. (2018). These operations help improve segmentation accuracy. Segmentation is the process of identifying which particular pixels serve as edges and lie on the boundary between two contiguous regions in an image. We used a custom, two-pass version of Canny edge detection (Canny 1986; Ohman et al. 2018) to segment regions of interest (ROI) within the field of view.

Image compilation

It is important that the images selected for annotation are selected without bias. This topic and other best practices for validating feature-based classification of plankton images are discussed by González et al. (2017). Our images were drawn from seasonal *Zooglider* coverage consisting of 225 dive profiles containing 1.45 million full frame image captures. We selected 150,000 full frame images and manually classified ~ 2 million ROI from them in an unbiased manner to obtain the 178,547 non-snow ROI used in our numerical experiments. Images were assigned to 27 categories (Table 1 and Fig. 3). We have previously found that approximately 1000 images per class is a rough guideline for the number of examples a class needs to be well defined (Graff and Ellen 2016). So for the purposes of early trials and debugging, we created a limited data set of no more than 1000 examples per class, which yielded a total of 25,047 ROI. We constructed a second data set by capping each class at 5000 examples, yielding 76,190 ROI. To quickly assess viable candidate algorithms, most of our preliminary explorations were executed on this data set, a technique we strongly recommend. As we narrowed in on a solution, we fine-tuned parameters and then report results using our largest data set, when possible. Most feature-based approaches could not be completely trained without running out of memory or continuing indefinitely. To construct the largest set, we combined all non-snow ROI with 171,447 marine snow ROI to total 350 k images (Table 1).

Our main assessments use this largest data set. We arrived at 350 k by evaluating larger data sets, but found no appreciable difference in accuracy, yet incurred longer computer run times. All ROI including snow were randomly sampled from their respective classes to avoid introducing biased metadata or other anomalies. The overall data set sizes are 1.5, 4.7, and 21.5 GB, with the

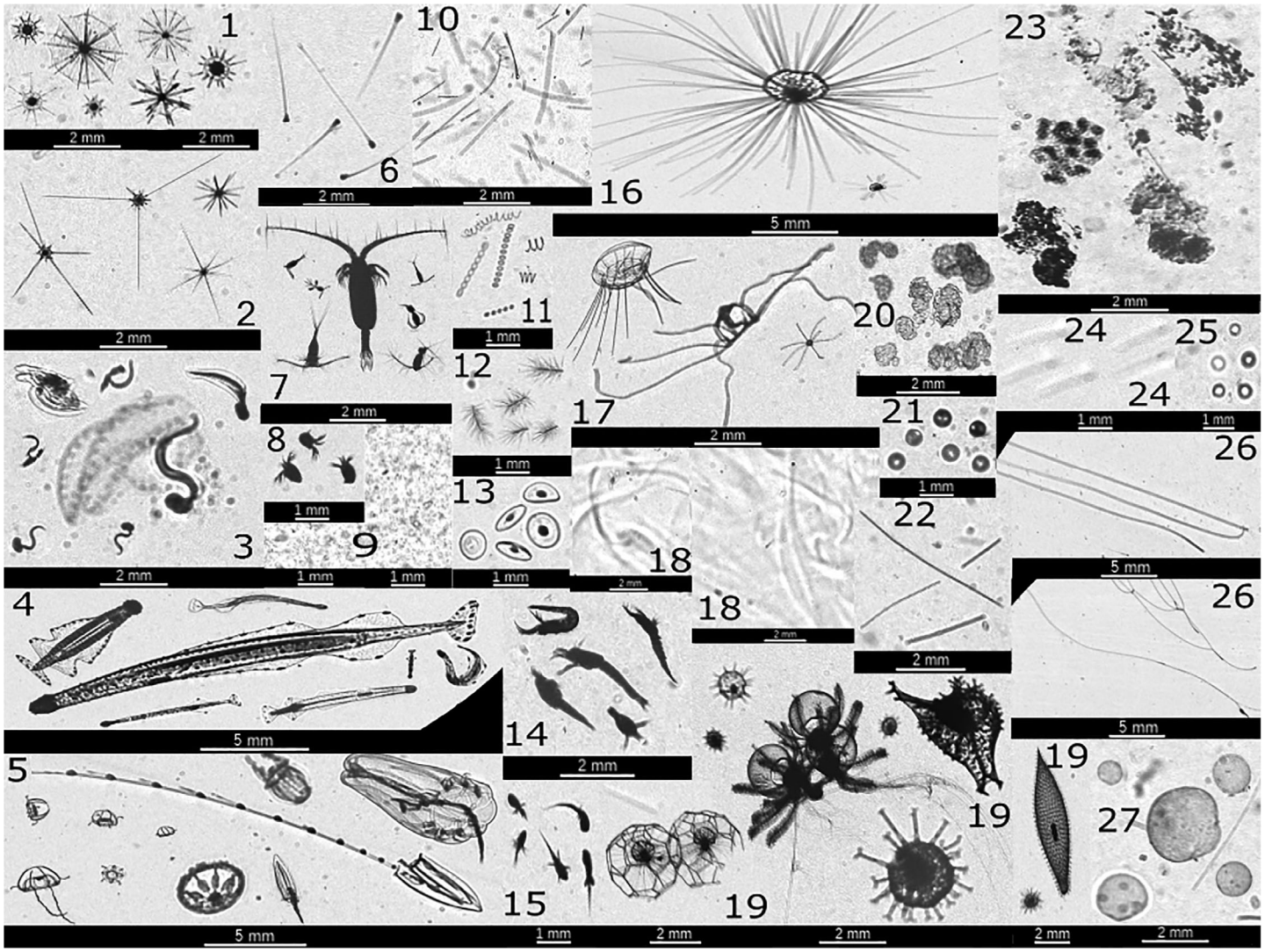

Fig. 3. Representative ROI for each of the 27 classes imaged by *Zooglider*.

metadata (described below) approximately 0.1 GB for each data set. CNNs require uniformly sized images. Based on the size of the majority of our ROI, we selected 128 × 128 pixels, which required rescaling some larger ROI, thus losing some detail. Smaller ROI were increased in size by padding with neutral pixels in order to conform to this size. Each neutral pixel contains a random intensity value sampled from a normal distribution around the mean intensity value of all images, intended to minimize introduction of artificial edges or areas of uniform color that would be falsely detected by first layer filters. Therefore, the padded neutral pixels, when rendered, yield a speckled appearance rather than a solid thick frame of a single color. When resizing, we used resampling with the Lanczos filter to resize the images (Blinn 1998).

Hydrographic, geotemporal, and geometric metadata

We incorporated three types of context metadata: hydrographic, geotemporal, and geometric (Table 2). Hydrographic metadata are intended to reflect the in situ environment of the specific water parcel in which the image was acquired. These measurements include *Zooglider* measurements of Chl *a* fluorescence, salinity, density (calculated in accordance with Fofonoff and Miller 1983 using Fernandes 2014), and temperature, an upwelling index (Schwing et al. 1986; Pacific Fisheries Environmental Laboratory 2018; for 33°N, 119°, averaged for the 10 d preceding each *Zooglider* image), and two different ways to approximate object concentration: acoustic backscatter and distance between ROI. Chl *a* and CTD measurements are made by *Zooglider* every 8 s, while Zoocam images are acquired at 2 Hz; hence, measurements are linearly interpolated to each Zoocam frame. We also use as metadata *Zooglider*-measured acoustic backscatter at the two acoustic frequencies, and the difference between them, which helps distinguish small and large sound scatterers. The Zonar on board *Zooglider* does not ensonify the same volume as that

Table 2. Three types of context metadata (geometric, geotemporal, and hydrographic) incorporated in machine learning experiments. Numerals indicate the number of variables.

Geometric (58)	Geotemporal (22)	Hydrographic (13)
Area (**7**) Area (filled area), area excluding holes, convex hull area, equivalent circular diameter, skeletal area, area excluding holes/area filled ratio, extent (area/bounding box ratio) *Perimeter* (**8**) Perimeter (filled), perimeter (with holes), convex hull perimeter, Feret diameter, height, width, fractal dimension, orientation *Circularity* (**12**) Major axis length, minor axis length, circularity (filled), circularity (with holes), elongation, eccentricity, Feret diameter/area filled ratio, Feret/area excluding holes, perimeter (filled)/Feret ratio, perimeter (filled)/area filled, perimeter (filled)/area excluding holes ratio, perimeter (filled)/major axis length ratio *Symmetry* (**8**) Centroid [X, Y], weighted centroid [X, Y] centroid distance, centroid distance/area excluded ratio, horizontal symmetry, vertical symmetry *Gray level* (**14**) Gray level normalized cumulative histogram statistics: [Slope, 1st quartile, 2nd quartile, 3rd quartile], intensity [Min, Mean, Max, Range, Std_Dev, Skew, kurtosis], intensity mean position (max-mean/range), intensity signal/noise ratio, coefficient of variation of pixel intensity *Image moments* (**9**) Central moments	*Geographic information* (**8**) Latitude, longitude, bottom depth, distance from [shore, pt. conception, Santa Barbara Basin], distance from closest shallow point (600 m), sampling depth proxy (pressure in dBar) *Temporal information* (**14**) Season (1 of 4) Time of day (day/night/twilight status—1 of 8) PDO state* El Nino/La Nina state† (as San Diego de-trended sea level anomaly)	*Measured* (**6**) Chl *a* fluorescence, salinity, temperature, 200 kHz scattering volume (S_V)‡, 1000 kHz scattering volume (S_V)‡, dB difference ($S_{V\ 1000kHz} - S_{V\ 200\ kHz}$) Derived (**7**) ρ (density), upwelling index at 33 N 119 W§, distance to nearest neighbor ROI‖ (in frame, up to 5 nearest, in mm)

*PDO from http://research.jisao.washington.edu/pdo/PDO.latest.txt (Mantua et al. 1997).
†Anomaly from http://oceaninformatics.ucsd.edu/datazoo/data/ccelter/datasets?action=summary&id=153.
‡Returns between 3 and 8.1 m from the transducers were averaged into 1 m depth bins (Ohman et al. 2018).
§Upwelling from https://www.pfeg.noaa.gov/products/PFEL/modeled/indices/upwelling/NA/data_download.html (Pacific Fisheries Environmental Laboratory 2018).
‖Calculated after full-frame images are segmented.

being imaged, so the acoustic return is not a property of any recorded ROI, but provides context information about the aggregate density of nearby sound scatterers. Acoustic backscatter is averaged in 1 m depth bins.

The full frame image from which ROI are segmented also provides information about nearby particle density. We calculate the individual distances from each region to each of its nearest neighbors in the frame, up to 5. For frames with less than six ROI, a default maximum pixel distance of 9999 is used. This metric will not always be accurate, since a region just outside of the field of view may be closer than one within the field of view, and we cannot account for the depth of the field of view. However, it is consistent as a two-dimensional metric for comparisons to other images within our data set and should provide an indication of the localized particle density.

Our geotemporal metadata identify the place and time that the image was acquired. Values measured directly aboard *Zooglider* are hydrostatic pressure, time of image capture, and latitude and longitude interpolated between each glider surfacing. Based on these position values, bottom depth is obtained from ~ 100 m grid cells calculated by downsampling bathymetry (NOAA 2016). We also calculate distance to Point Conception (a major upwelling center) and distance to the Santa Barbara Basin (a productive area, Ohman et al. 2013). Distance to the coast and distance to the nearest continental slope (600 m) are calculated using the downsampled bathymetry. We generate four types of temporal metadata: time of day (divided into eight time intervals); season (four seasons, each 3 months long); El Niño-Southern Oscillation index off California (monthly, from Lilly and Ohman 2018); and Pacific Decadal Oscillation (monthly, from Mantua et al. 1997).

Geometric features extracted from the images were used as a third type of metadata for the CNN architectures (geometric features are required for feature-based approaches). The geometric values are calculated with respect to the pixels that are designated by the segmentation algorithm as being within the region (e.g., mean intensity, kurtosis, area, diameter, weighted centroid). While these values are derived from information within the image itself, the geometric features are metadata in that they describe the original image contents and region size before the image is rescaled and pixel values are adjusted for processing by the CNN. These values include measurements of the segmentation boundary, such as perimeter length and eccentricity, and information about the originally measured intensity values, such as minimum, maximum, and average, which are not otherwise provided to the CNN. Combined, they provide context about the illumination and scale of the original image capture. Additional detail regarding these 58 geometric measurements is provided in Ellen et al. (2015).

Procedures

For each of our assessments, we split the data into 80% for training, 10% for validation, and 10% as the test set. We generated 10 different randomly selected sets with these split ratios as replicate trials.

Most of the algorithms are designed to accept feature values across a defined range, usually [0–1] or [−1 to 1]. In prior work, we examined four different whitening and normalization techniques, and found that with our images, per region normalization worked best (Graff and Ellen 2016). Commonly referred to as Global Contrast Normalization, the mean value of the image is subtracted from each pixel, and the result is divided by the standard deviation of the original pixel values. Since each type of metadata measurement has a scale different from the others (e.g., temperature or sampling depth), we also subtracted the mean of the measurement from its observations and divided by the standard deviation. All normalizations are calculated using the 80% split of training data for each replicate.

We calibrated each model to each replicate of the data, a process commonly referred to as hyperparameter tuning. While some of our feature-based algorithms require minimal tuning, CNNs require more careful evaluation to achieve a strong model. Training a single CNN consists of evaluating the network's performance on an image, then adjusting the network weights to reinforce good performance and alter bad performance. This is usually done by selecting one of the images at random without replacement, processing it, and then selecting another. The term "epoch" is used to describe the condition where the network has seen each training image one time.

This workflow creates a number of different options and hyperparameters, not all of which were evaluated. We used a batch size of 25 to evaluate multiple images simultaneously, thus increasing throughput. We imposed a limit on the number of epochs at 40, but this limit was rarely needed (see Bengio 2012 and Smith 2018 for guidance on stopping criteria and other hyperparameter choices). We also assess data set augmentation (Dai et al. 2016; Dieleman et al. 2016*b*), which involves generating synthetic examples to improve overall accuracy. Because our images are captured with known pixel pitch and images are centered by our segmentation process, we only assessed horizontal reflection, vertical reflection, and rotation.

CNN architecture

We trained our CNNs de novo, rather than adopting networks from different application domains because our de novo results were markedly better in both this and our previous work (Graff and Ellen 2016). Initial networks have nearly random weights and no discriminative power. With each successive example, weights are adjusted. The learning rate controls the amount the weights are adjusted in response to the most recent example and is an important hyperparameter. We used the Adam optimization algorithm (Kingma and Ba 2015), which updates all network values, in addition to modifying the initial learning rate. Two initialization algorithms are made available through the Lasagne/Theano software (Glorot and Bengio 2010; He et al. 2015). We evaluated both, found no significant difference, so we used the former.

Network shape has a large impact on results, and is an active area of research (Lee et al. 2015; He et al. 2016; Sabour et al. 2017; Szegedy et al. 2017). We implemented a network shape based on

the VGG-16 model (Simonyan and Zisserman 2014), but on a smaller scale, since there was a 1000-way classification problem with images of 224 × 224. We also used small filters of size 3 × 3 for every layer and the rectified linear unit activation function (ReLU), but otherwise the convolutional portion of our network was approximately one-quarter the size of their network. We had a total of five convolutional layers, with 16, 32, 32, 64, and 64 filters, respectively, with a pooling layer between each one (Fig. 4). The pooling layers serve to reduce the input size from one layer to the next by half, using maximum value pooling: that is, for each 2 × 2 area of activations, the maximum value is selected for use as a single value going forward (not the mean).

One other key architectural detail in the VGG-16 and related models is the use of fully connected layers of neurons prior to the final softmax layer that determines the classification. We reduced the size of these fully connected layers as a hyperparameter investigation, arriving at a configuration with fewer and smaller layers (one eighth or more) than those used in VGG-16, which increased accuracy while also decreasing training time by 50% or more.

Since convolutional layers are designed to operate on image pixels, there is no means to fuse metadata directly into the convolutional layers. One approach to incorporating additional context metadata is to concatenate metadata values to the penultimate network layer. This is the approach used by Tang et al. (2015) to achieve their 5-point gain in accuracy. Instead, we find better accuracy when we incorporate the features earlier into fully connected layers, as illustrated schematically in Fig. 5. Our best model, which we call Metadata Interaction, allows some interaction between the features with the output of the final pooling layer.

Figure 5 illustrates variations in the final fully connected layers, to the right of the dashed line labeled "classification" in

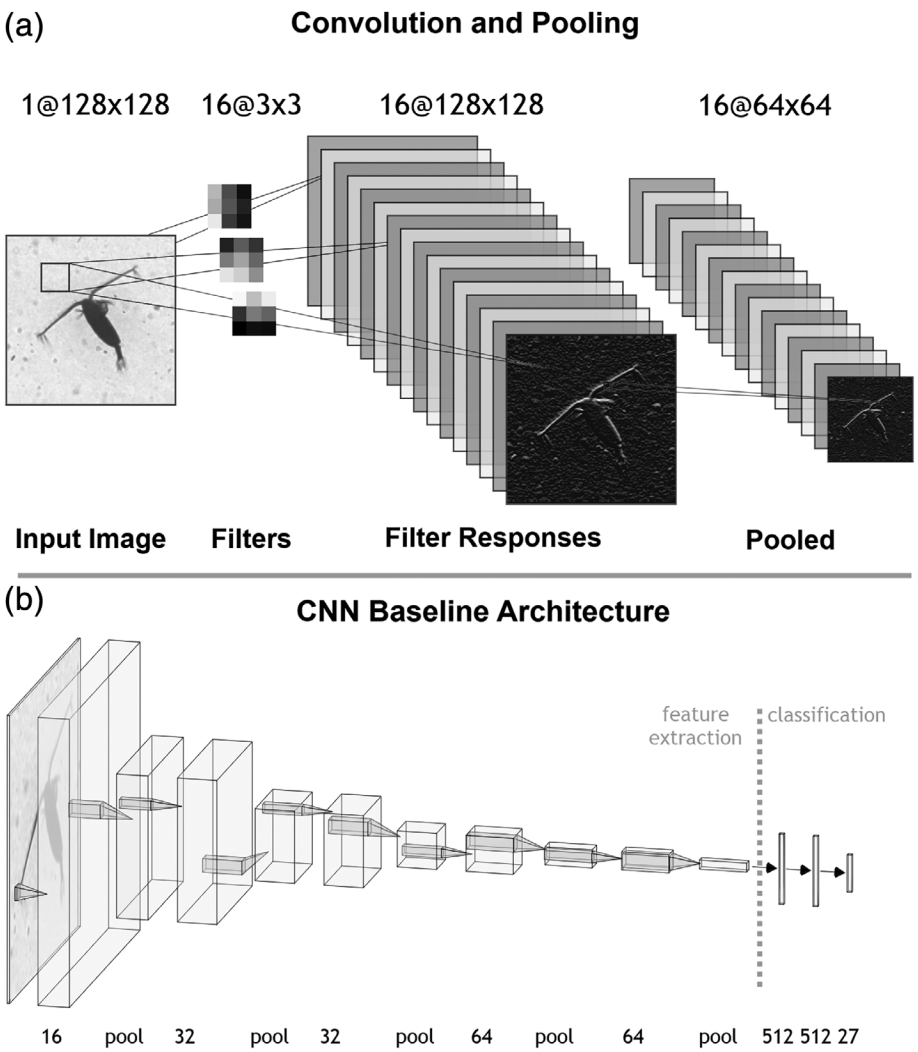

Fig. 4. Our CNN architecture. (**a**) Illustration of the first convolution and pooling layers. Our input images are 128 × 128. Each of the 16 3 × 3 filters is convolved against the input, resulting in an activation volume of 16 × 128 × 128. A 2 × 2 max pooling layer scales the image by 50%. (**b**) Our baseline architecture has five convolutional layers with 16, 32, 32, 64, and 64 filters, all are 3 × 3. A 2 × 2 pooling layer follows each filter layer. After these 10 layers are two fully connected layers, each with 512 neurons before the final softmax length 27 vector corresponding to predicted classification.

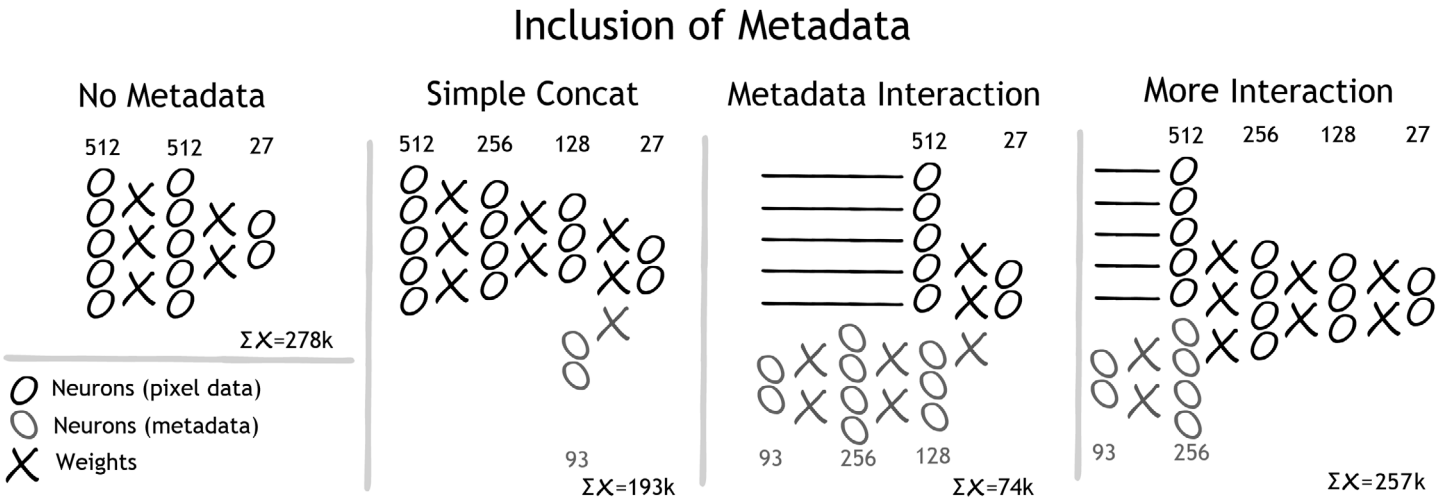

Fig. 5. Schematic illustration of our baseline (left) and three architectures for metadata incorporation (Simple Concatenation, Metadata Interaction, and More Interaction). All convolutional layers precede illustrated alternatives, as illustrated in Fig. 4.

Fig. 4. All four of these architectures have identical configurations of five convolutional and five pooling layers (Fig. 4b). In a fully connected layer, each neuron's output is routed to every neuron's input in the subsequent layer, with a weight on each route. Therefore, the number of weights applied to a fully connected layer's output is the product of the size of the layer and its successor. Regardless of whether the input is from the image or the metadata, since they are implemented as neurons, all of these layer variations are trained using the same algorithm as the convolutional layers. Our selected "No Metadata" architecture routes the convolutional layer's output to two consecutive layers of 512 neurons followed by a layer of 27 neurons, resulting in a total of ~ 278 k weights. (Fig. 5—convolutional layers not pictured that contain ~ 700 k additional weights in an identical configuration for all pictured models). If we concatenated the vector of all 93 features to the penultimate layer, that CNN would have slightly more weights than the No Metadata option. Therefore, our simple concatenation model has smaller fully connected layers of 512, 256, 128, and 27. After adding metadata, there are only ~ 193 k weights, to ensure that any gain in accuracy must be from context metadata. Our Metadata Interaction model is even more restricted. We use the same layer structure as in simple concatenation (256, 128) but route the metadata through the multiple fully connected layers instead of the CNN extracted features, so the number of weights is significantly less than either (~ 74 k weights). Alternatively, we route the metadata through a single layer, combine them with the extracted features, and use two more fully connected layers for a total of 257 k weights. These are the largest numbers, 193 k, 74 k, and 257 k, corresponding to the usage of all 93 context metadata features.

Dropout (Hinton et al. 2012) acts as "a stochastic regularization technique" (Srivastava et al. 2014). Dropout is the concept of randomly ignoring the output of some neurons in the network in order to strengthen the rest of the network, and in most cases is beneficial. We assess the impact of dropout on both pixel data and on context metadata.

Performance metrics

We report binary accuracy for each of our models, where full credit is given for each correctly classified image and none for incorrect classifications, regardless of class of origin. A confusion matrix is used to interpret class-specific distribution of true/false positives and negatives. Timing information, when provided, is for single-threaded computations for training and testing a single replicate of the data. It does not include the time to load the data set into memory. Our boxplots display whiskers equal to 1.5 times the inner quartile range, with the results of individual trials overlain as circles to indicate the distribution of the trained models. The number of trials was often as few as 10 (1 for each replicate) and rarely more than 20.

Assessment

Feature-based algorithm assessment

We assessed a range of different hyperparameters in order to select values that provide the best overall performance of each of the five feature-based algorithms that we evaluated, starting with only the 58 geometric features (Fig. 6). Heatmaps show averages at each combination of the two most influential hyperparameters we assessed across 10-folds, with the less computationally demanding algorithms having multiple repetitions (Fig. 6, left column). By examining the variance across all trials for the most important hyperparameter value (Fig. 6, right column), we are able to determine that no further search is warranted. The leftmost bar in each Fig. 6 right column panel contains a suboptimal combination, the middle bar shows the results from adjusting the key hyperparameter by a single increment, and the rightmost bar shows the results from one additional increment. The

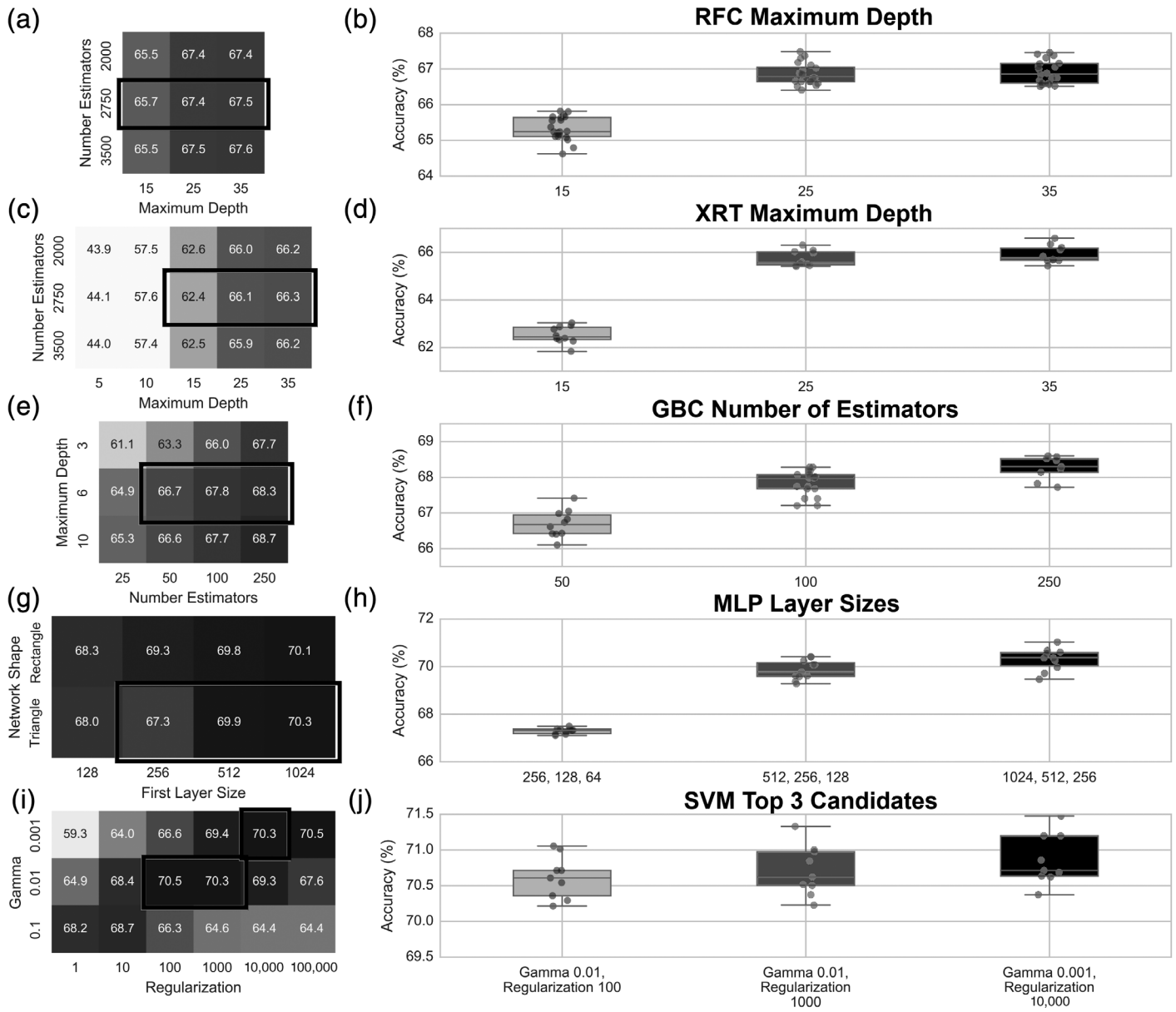

Fig. 6. Hyperparameter grid search results for five different feature-based machine learning classification methods (**a**, **b**: RFC; **c**, **d**: XRT, **e**, **f**: GBC; **g**, **h**: MLP; **i**, **j**: SVM). Cells in left column contain average results across all trials for a given hyperparameter combination. Results for each hyperparameter combination across one key region of the grid search are highlighted with a box, and the boxplots in right column show individual trials for these regions illustrating the variance within that configuration.

rightmost values show little accuracy gains, but all have significantly higher computational cost. For example, time to convergence for SVM on a single replicate of our medium data set with regularization strengths of 100, 1000, and 10,000 resulted in convergence times of 2 h, 6 h, and 34 h, respectively. Therefore, the middle bar represents the selected hyperparameter combination for all further assessments. The hyperparameters with the most impact on our assessment for the three Random Forest based algorithms are the same (RFC, XRT, and GBC, Fig. 6a–f). They each have a constraint on the maximum size of the forest constructed (number of estimators) and a limit on the number of features considered in each tree/stump (maximum depth). For SVMs, best practices are to perform a grid search over the kernel coefficient for the decision boundary (gamma) and the penalty parameter that determines the strength of the error term (Fig. 6i,j). Both are recommended to be evaluated in geometric/exponential increments (Hsu et al. 2003). Our MLP uses the Adam optimization algorithm and the rectified linear unit activation function, which is the default parameter, and also the same as our CNN architecture. The MLP's shape is determined by

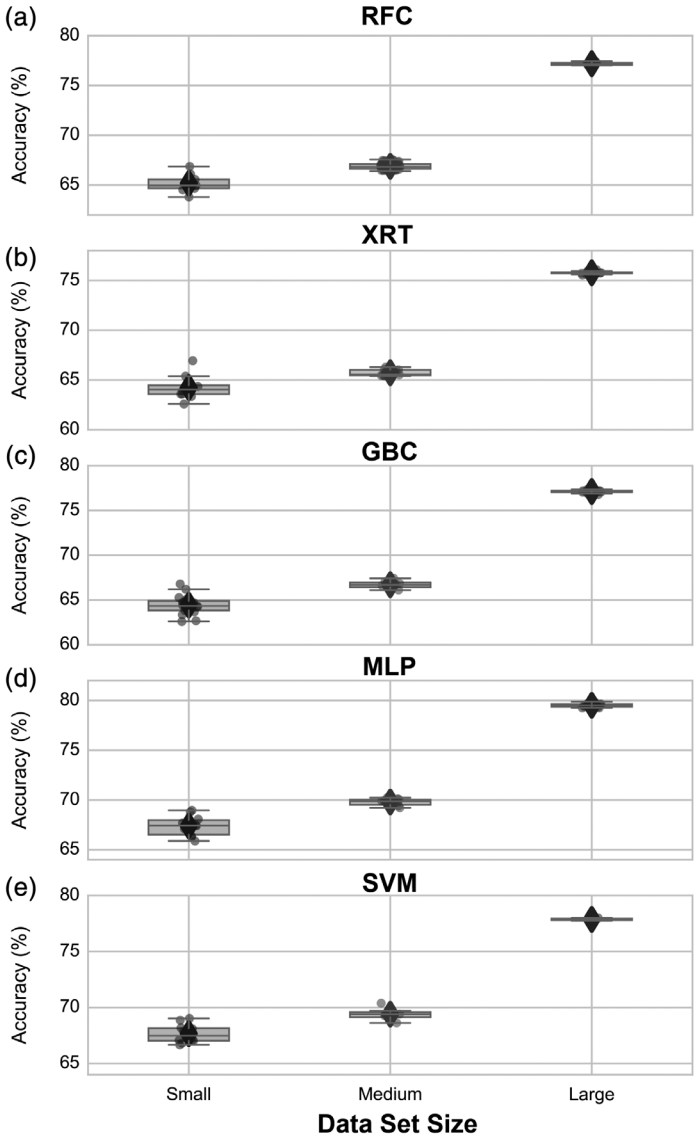

Fig. 7. Accuracy vs. data set size for five different feature-based machine learning classification methods (**a**: RFC, **b**: XRT, **c**: GBC, **d**: MLP, **e**: SVM). The small data set contains ~ 25 k images, the medium data set contains ~ 76 k images, and the large data set contains 350 k images. All sets have 27 classes.

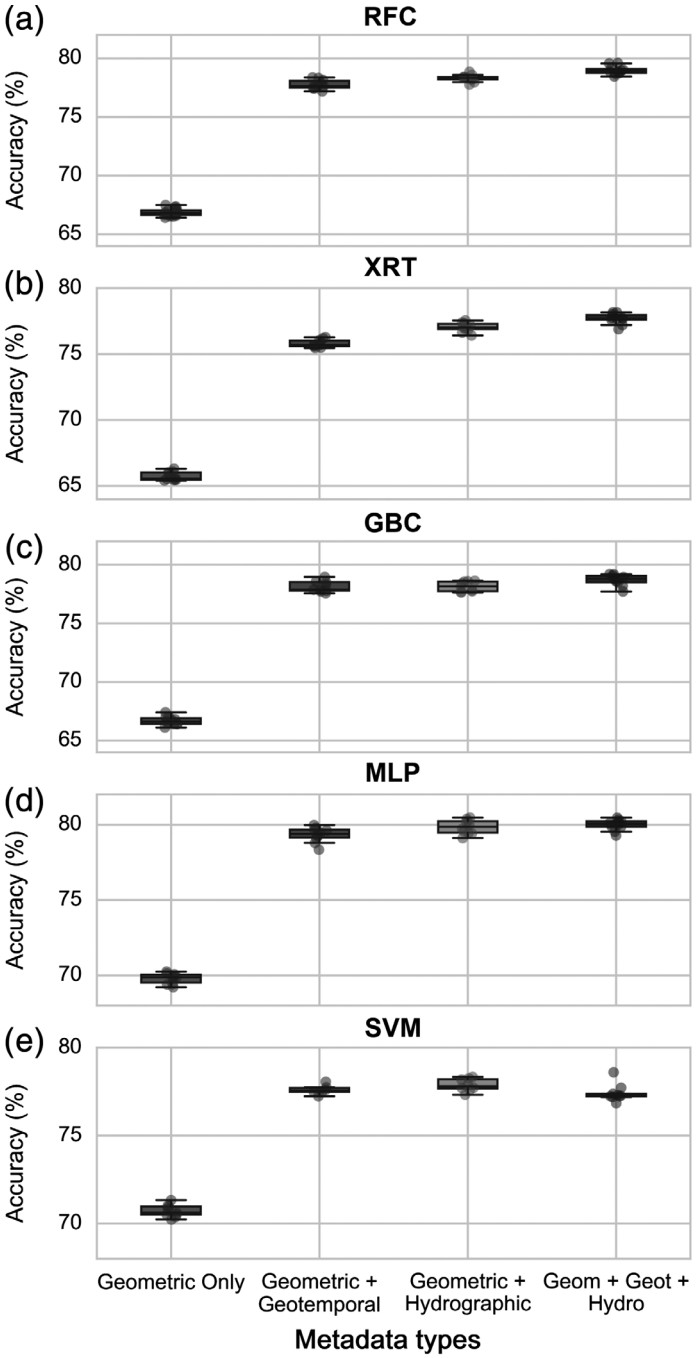

Fig. 8. The effect of metadata on classification accuracy for five different feature-based machine learning classification methods (**a**: RFC, **b**: XRT, **c**: GBC, **d**: MLP, **e**: SVM) on our medium data set. The leftmost bar in each graph corresponds to a model using only the 58 geometric features, the next bar adds 22 geotemporal features, and the next bar uses the 58 geometric features plus 13 additional hydrographic features. The rightmost bar utilizes all 93 features.

the hyperparameters of the number of "hidden" layers of neurons, and the number of neurons in each hidden layer (a hidden layer is between the input and output layers). We assessed two different network shapes; one with two equally sized layers (rectangle) and one with three layers, each half the size of the preceding layer (triangle). Our second hyperparameter is the width of the base layer (Fig. 6g,h).

Our evaluation of the effect of data set size on classification accuracy of the feature-based algorithms showed that the largest data set consistently provided the best results (Fig. 7). Our medium data set contains ~ 3× more training images than the small but the large contains ~ 14× more training images than the small; therefore, the increase in accuracy from our small to medium to large data set is less than linear with respect to the number of training images, suggesting we are approaching asymptotic performance.

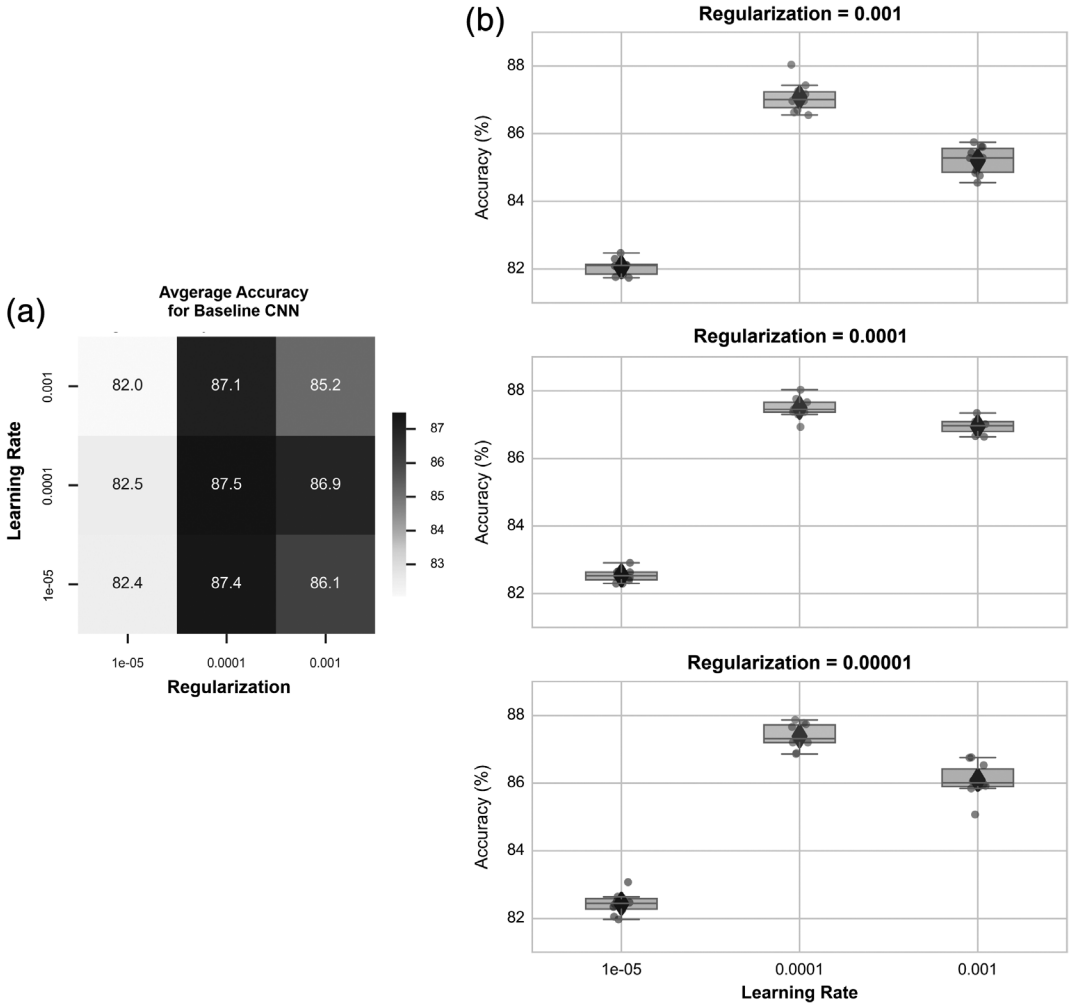

Fig. 9. Hyperparameter optimization for CNN. (**a**) Heatmap cells contain average accuracy across all trials for a given combination of hyperparameters. (**b**) Boxplots show the distribution of results for each hyperparameter combination in the heatmap. All trials use medium data set size.

Having optimized hyperparameters and data set size, we now turn to metadata. Inclusion of context metadata significantly boosts performance for all five feature-based algorithms (Fig. 8). For algorithms assessed on the medium data set, we find gains of 6.9–12.2 percentage points. This gain is similar to the benefits of using the large set on geometric data only (Fig. 7). Geotemporal and hydrographic metadata have approximately the same influence, and inclusion of both results in the best overall classification accuracy. Having nearly doubled the size (from 58 to 93), the feature-based approaches could not be completely trained without running out of memory or continuing indefinitely, so those results are not presented.

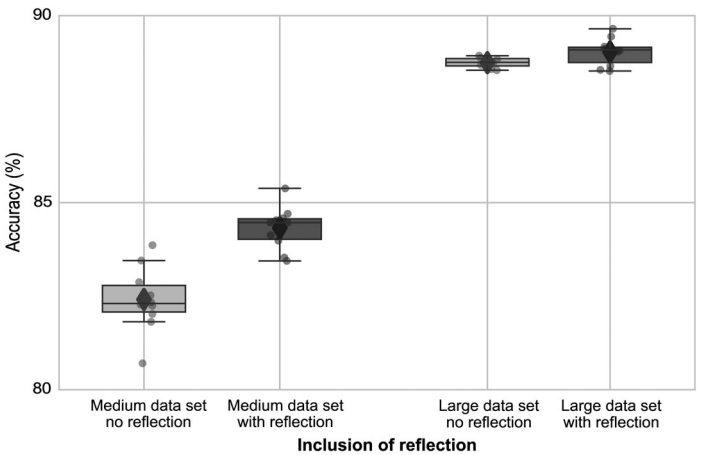

Fig. 10. The effect on classification accuracy of using reflection as a runtime augmentation with our baseline CNN architecture, with (left) medium and (right) large data sets.

CNN assessment

CNNs have more hyperparameters that dramatically affect performance, so more preliminary investigation is required. Two that have the largest impact on performance are learning rate and regularization strength. We found the effect

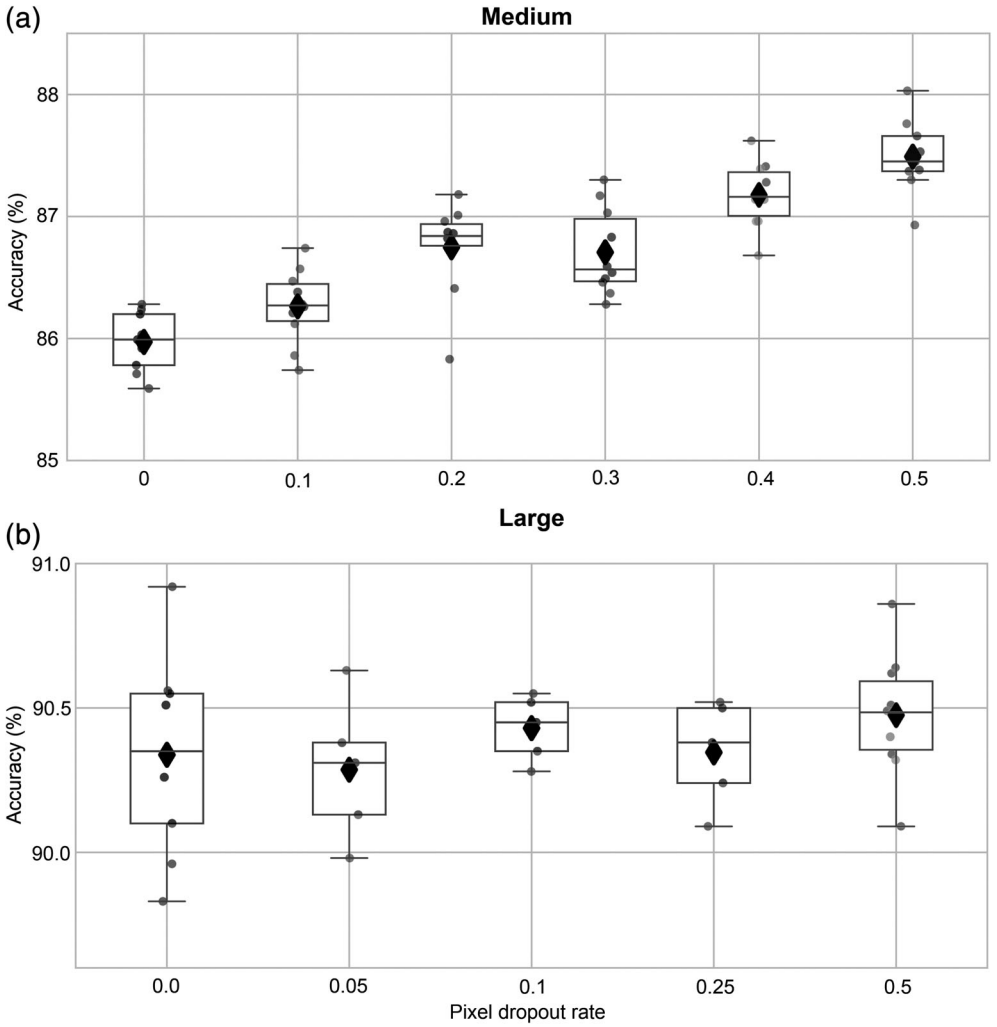

Fig. 11. The effect on classification accuracy of using dropout with our baseline CNN architecture for (**a**) the medium data set and (**b**) the large data set. *x* axis indicates the dropout probability on only the fully connected layers of neurons.

of learning rate to be much stronger than that of regularization, but both have a local maximum at regularization = 0.0001 (Fig. 9). As our CNN architecture matured, we revisited this assessment, but setting both values to 0.0001 remained optimal for our data. All subsequent figures use this value.

Augmentation strategies of horizontal and vertical image reflection have a stronger impact on performance with our medium than with our larger data set (Fig. 10). Our implementation used a 50% chance at runtime for each reflection operation on each image in each epoch, thus there was no additional computational demand, so we used this augmentation on all subsequent figures.

We evaluated the impact of dropout by incrementing the probability that any particular neuron will have its output ignored. We found a nearly monotonic association between dropout probability and accuracy on our medium data set (Fig. 11a) but a negligible effect with our large data set (Fig. 11b). Figure 11 reports results when we applied the dropout probabilities on only the fully connected layers of neurons. Our finding of limited influence of dropout with larger data sets is notable because of the widespread use of dropout (Srivastava et al. 2014). Since using dropout does not decrease performance, and it is widely used in machine learning applications, we choose to use it for the remainder of our assessments to be in line with contemporary conventions. Since the dropout value of 50% provided the most benefit for the medium data set, we used 50% dropout for the rest of our assessments. We assessed numerous network configurations before arriving at our selected baseline method. This baseline method performed as well or better than the other alternatives we evaluated (Fig. 12). Our baseline model has five convolutional layers (16 filter convolutional layer, pooling layer, 32, pool, 32, pool, 64, pool, 64, pool) as shown in Fig. 4. This model, labeled "Baseline" in Fig. 12, had an improvement in accuracy of 1.5 points over a similar model with three convolutional layers (16, pool, 32, pool, 64, pool), labeled "Fewer Layers" in Fig. 12. Models with dropout applied to the convolutional layers (Fig. 12—"More

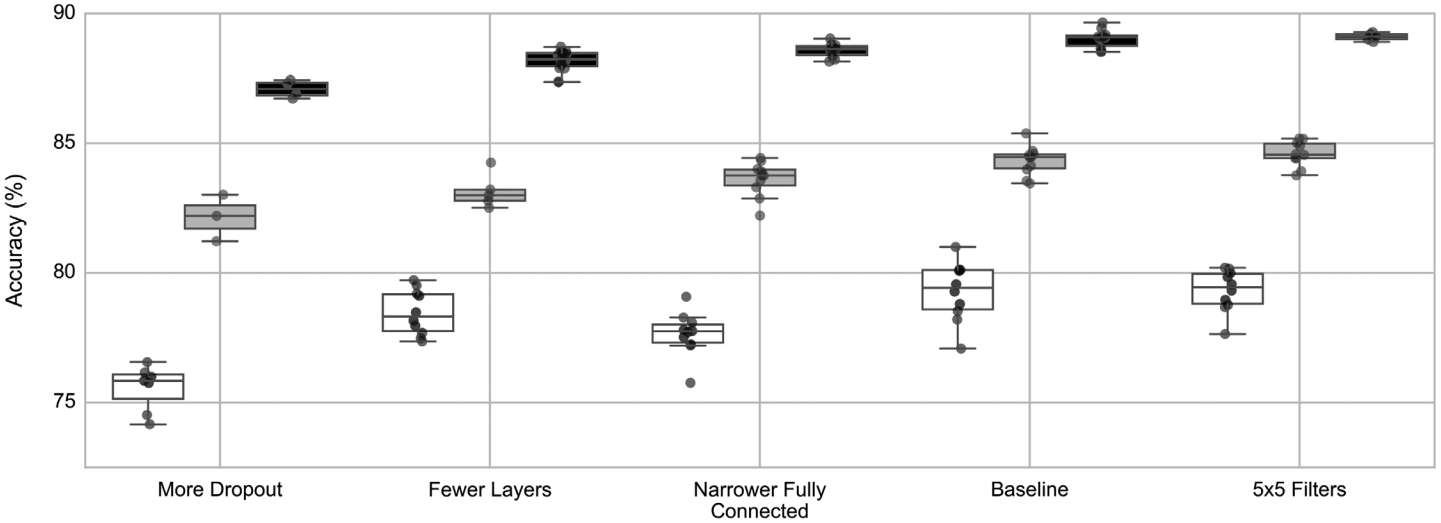

Fig. 12. The effects of CNN architecture (i.e., changes in dropout, number of layers, connectivity, and filter size) relative to our baseline architecture (4th column) for different training set sizes: small (white), medium (gray), and large (black). All results use pixel dropout and reflection.

Dropout"), and with narrower fully connected layers (512, 256, 128, 27—Fig. 12—"Narrower Fully Connected") had lower accuracy than our selected baseline configuration with three fully connected layers (Fig. 5–512, 512, 27). Substituting 5×5 filters in place of our 3×3 filters, triples the amount of memory required but yields nearly identical results (Fig. 12—"5×5 Filters" vs. "Baseline"). Accuracy from our CNN is markedly better than all of the feature-based approaches: the accuracy of our baseline CNN on even our smallest data set (25 k ROI; ~ 1 k per class) exceeds accuracy of each of the feature-based classifier accuracies on the largest data set (350 k ROI; maximum 5 k per class) as shown in Fig. 3. Our CNNs exhibited a nearly linear relationship between convergence time and number of training examples, i.e., 1–2 h per trial on our small data set and 8–12 h per trial on our largest data set.

Having selected our baseline CNN, we then analyzed the effect of augmenting the pixel information with context metadata (Fig. 13). Both geotemporal and hydrographic context

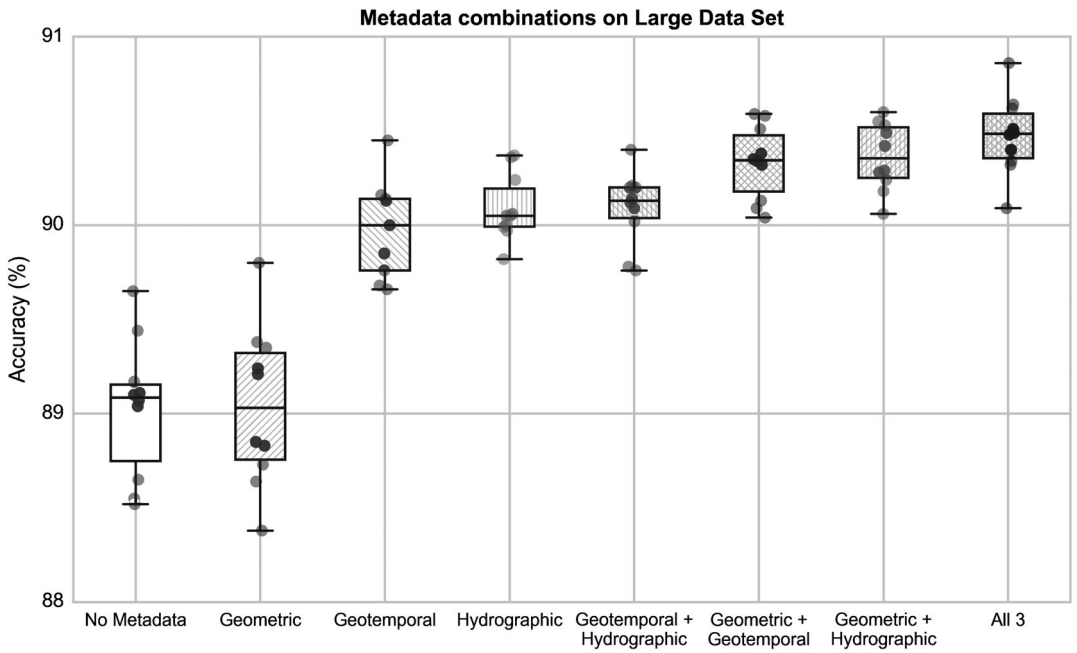

Fig. 13. The effects on classification accuracy of adding context metadata on our large data set. Experiments include no metadata and hatching indicates the contribution of every combination of geometric (/), geotemporal (\), and hydrographic (|) metadata.

metadata individually make a significant improvement on classification accuracy ($p < 0.001$; Fig. 13). However, the combination of both geotemporal and hydrographic metadata yields a classification accuracy similar to each of them individually, potentially indicating overlap or redundancy between the features. Combining each individually with the geometric metadata provides a boost in performance. One possible explanation for this result is that the geometric metadata includes information about the original region of interest size, which on its own did not prove valuable, but size given a depth or temperature may have discriminative value. Using all three metadata types provides still more accuracy gain, to 90.5% accuracy. Hence, our remaining analysis will be conducted utilizing all 93 features (Table 2).

We find that the manner in which metadata are incorporated affects accuracy. We obtained better accuracy when we incorporate the features earlier into fully connected layers (Fig. 14). Our Simple Concatenation metadata model not only has smaller weights overall than our model without metadata (Fig. 5—193 k vs. 278 k), but specifically has smaller fully connected layers of 512, 256, 128, 27. Above we have shown (Fig. 12) that this configuration is less effective than a configuration with layers of 512, 512, 27, so all accuracy gained must be from the metadata inclusion. Because the metadata interaction requires more weights for the metadata, we remove the fully connected layers from the pixel-based data entirely, providing evidence that all improvement from the Metadata Interaction model over the Simple Concatenation model is from the metadata and interaction, not network size or shape.

We found that applying dropout to the fully connected layers containing the features derived from metadata is detrimental to accuracy (Fig. 15). Metadata dropout is detrimental even if pixel dropout is removed (Fig. 15a), especially at high dropout fractions. Metadata dropout is detrimental for the large set (Fig. 15b).

We investigated more advanced CNN architectures to pursue additional accuracy (Fig. 16). Cyclic Pooling and Rolling (Dieleman et al. 2016*b*) have been shown to improve accuracy at the cost of much longer runtimes (5–8× longer). Our Metadata Interaction model provides almost as much benefit as Cyclic Pooling and Rolling (median 90.70% vs. 90.40%). Doubling the number of filters in each layer results in a small performance gain (to 90.92%) at the cost of 50% longer runtimes. Doubling the number of layers instead results in a smaller performance gain at the cost of 100% longer runtimes. Cycling Pooling and Rolling are still beneficial with metadata, and across different network sizes. Cyclic Pooling can be applied by itself, but Metadata Interaction plus Cyclic Pooling and Rolling is better, and this combination is significantly better than using Pooling and Rolling without metadata ($p < 0.0001$). Doubling the number of filters is now not nearly as beneficial; this result makes sense if the additional filters were being devoted to learning rotations of other meaningful filters. Accordingly, doubling the number of layers and also augmenting with Cyclic Pooling and Rolling provides more of a gain than just doubling the number of filters. Our best model achieves 92.28% accuracy with our 27-class data set.

The confusion matrix in Fig. 17a evaluates class-wise performance of our best performing model, which includes context metadata added via Metadata Interaction, Cyclic Pooling, and Rolling (Fig. 17a). Our confusion matrix is shaded to prioritize true positive rate. For example, 21 of the 75 fish larvae in the test set were mislabeled as copepods, so that cell has a strong red shading, but the 111 snow mislabeled as copepods (corresponding to 0.06% out of the 17,889 snow ROI in the test set) is essentially uncolored. Figure 17b illustrates the benefit of inclusion of metadata for particular classes, showing that most classes benefit. The

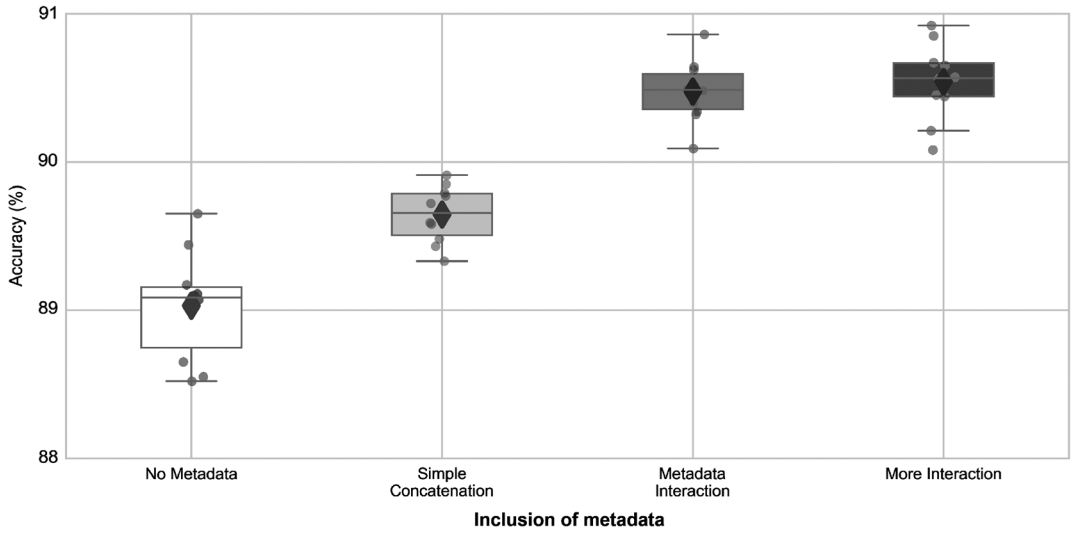

Fig. 14. The effects on classification accuracy of different approaches to incorporating metadata (Simple Concatenation, Metadata Interaction, and More Interaction) for the large data set.

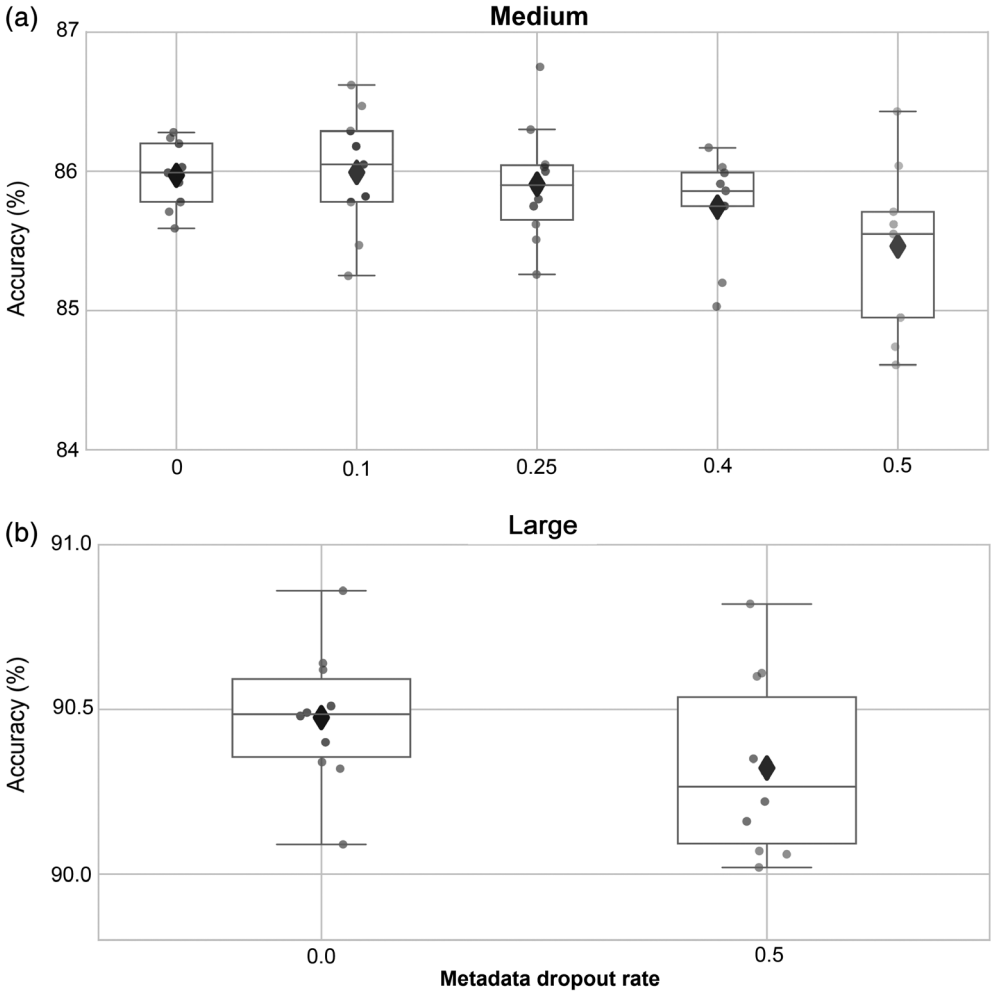

Fig. 15. The effects on classification accuracy of including dropout with our CNN architecture for (**a**) the medium data set and (**b**) the large data set. *x* axis indicates the probability that an individual unit's value would be dropped.

four largest gains are found for nauplii, narcomedusae, euphausiids, and fish larvae, corresponding to some of the least observed classes. Prior to inclusion of metadata, some chaetognaths had previously been labeled as three other relatively thin and straight classes: appendicularians, tentacles, and thread-like diatoms, while all three error types are minimized with the addition of the metadata.

Discussion

Impact of context metadata

We found that inclusion of context metadata provides gains in classification accuracy for both CNNs and feature-based classifiers. In the case of CNNs, the accuracy gain averaged 1.3 points, increasing the overall classification accuracy to 90.5% prior to enhancing CNN architecture. While the numerical increase is modest, the results were consistent across all replicates and represent a systematic improvement in overall accuracy, with appreciable gain in specific classes of images. Inclusion of metadata also improved CNN execution time, reducing convergence time by 17% (from 30.9 epochs to 25.6 epochs). In the case of feature-based classifiers, inclusion of metadata markedly increased classifier accuracy on the medium data set between 6.9 and 12.2 points, depending on the method considered.

Our estimate of the impact of adding metadata is likely conservative, because our feature-based models are possibly undersized for the metadata. Since models with and without metadata cannot be the same size while also being the same complexity, we favored the models without metadata. We initially tuned our models with 58 geometric measurements, held hyperparameters constant (e.g., number and depth of decision trees), but then added the geotemporal and hydrographic context metadata without retuning, nearly doubling the input to 93 features. Metadata increased execution time as much as 1.6×, proportional to the increase in the number of features (from 58 to 93). However, hyperparameter choices had 10–100× more impact on runtime

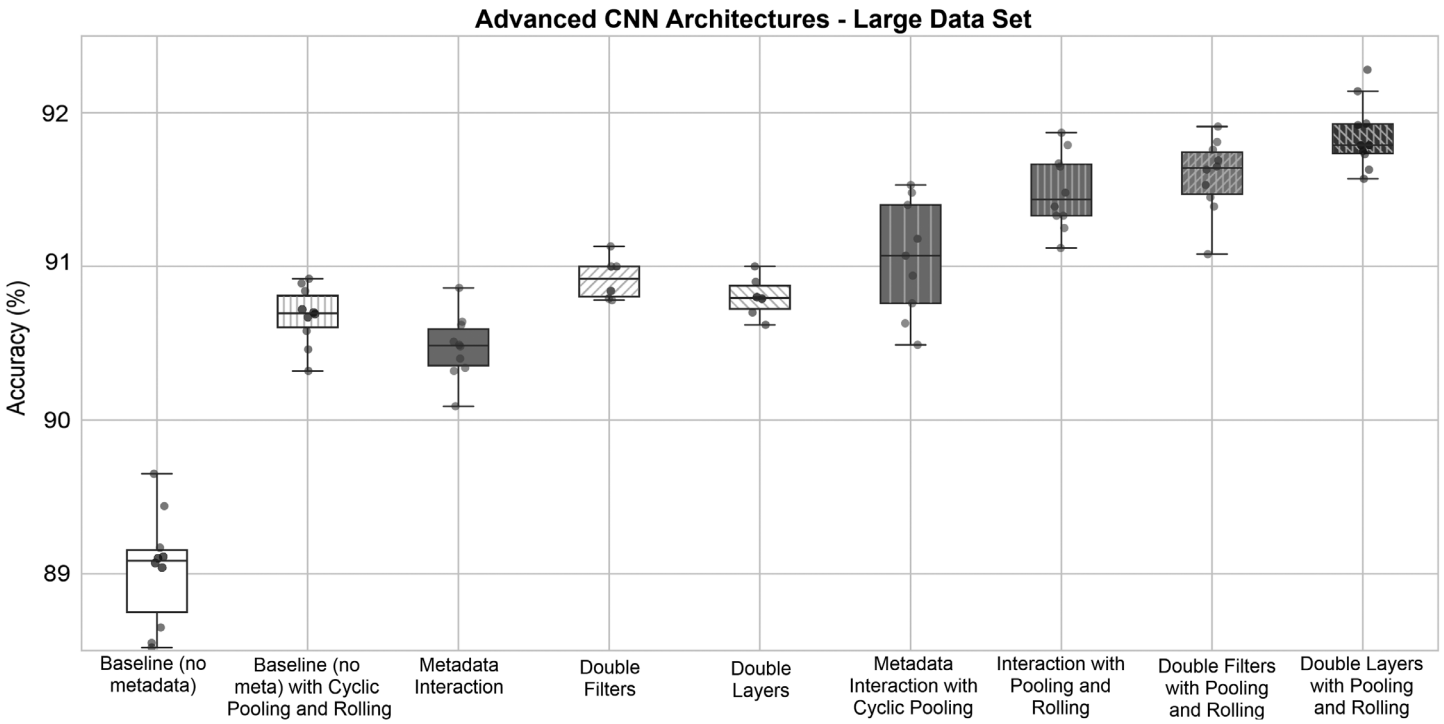

Fig. 16. The effects on classification accuracy of advanced CNN architectures on our large data set. Models with metadata are shaded and models with other augmentations have hatching. See text for explanation.

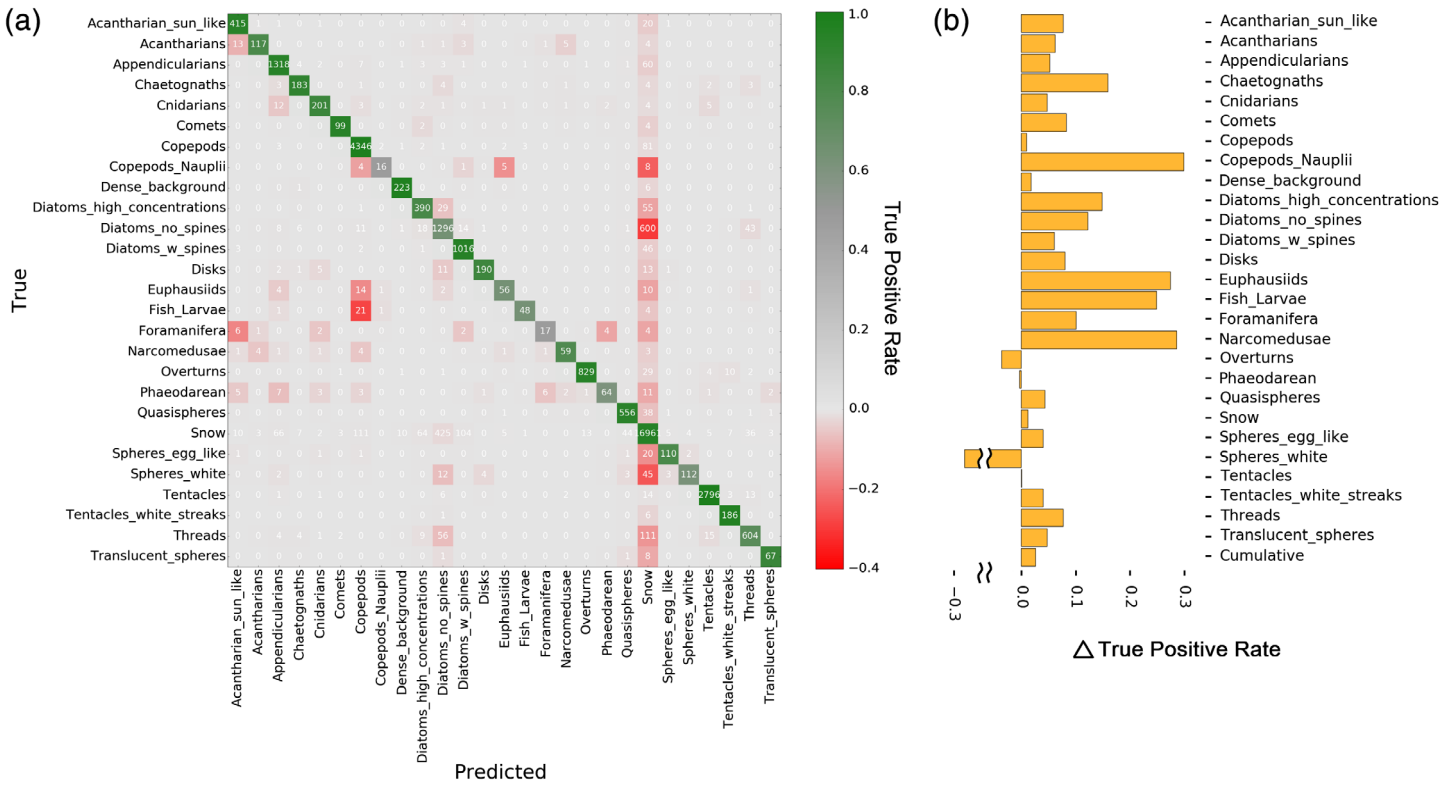

Fig. 17. A confusion matrix indicating the specific errors made by our best performing model, which includes metadata interaction as well as cyclic pooling and rolling. (**a**) Rows indicate the true label and columns indicate the CNN algorithm's predicted label. Color intensity is proportional to the true positive rate. (**b**) Gains in classification accuracy from inclusion of metadata for each category of organism.

than this increase. Overall, these are substantial gains that illustrate the clear advantage to incorporating context metadata across a variety of machine learning methods.

Although we divided metadata into three categories for illustrative purposes, they are all treated equally within our architecture, which includes them in the later layers of our CNN. Inclusion of multiple types of metadata will usually outperform a single type for two reasons. The first is due to the CNN architecture, where strong positive correlations outweigh neutral or negative correlations, so images benefitting from one type of metadata will usually not be harmed by the inclusion of additional metadata that are neutral or even slightly contradictory. The second reason is that our Metadata Interaction architecture allows for combination of features to impact the classification (e.g., a specific temperature value takes different meanings in winter vs. summer).

We assessed 12 different architectures for incorporating context metadata into our CNNs. The most naïve incorporation of metadata provided an accuracy gain of 0.6 points, less than half the benefit provided by our best architecture. Seven of our interaction architectures yielded nearly identical results at 0.8 points beyond that Simple Concatenation approach. We used efficiency of execution as a tiebreaker for designating our preferred Metadata Interaction model. However, an efficient network with a data set size of 350 k could be undersized for larger data sets. Perhaps some of other architectures with additional fully connected neurons processing context metadata would outperform the simpler architecture we presented.

CNNs vs. feature-based algorithms

Prior to the development of CNNs, plankton images were classified with varying degrees of success primarily using geometric features (reviewed in González et al. 2017). Recently, CNNs have been applied to plankton classification problems hinting at the potential of the approach (Wang et al. 2016; Zheng et al. 2017). A public competition (Robinson et al. 2017) stimulated new solutions (Dieleman et al. 2016*b*) but there has not yet been a quantitative assessment of specific design choices when considering a CNN for plankton image analysis. Luo et al. (2018) validated the findings from the contest and showed that CNNs do successfully generalize to future images and therefore can be used as part of an end-to-end workflow that they outline in detail. However, their "described method is highly tuned to images collected by a particular instrument" (Luo et al. 2018). In addition to incorporating context metadata, here we optimized and quantitatively evaluated the performance of CNNs that incorporate a variety of machine learning augmentations in comparison with classical feature-based classifiers. We found that CNNs do consistently improve upon applications of feature-based approaches.

On the smallest data sets, the computational requirements for CNNs exceed feature-based approaches, but as the size of the data set increases, feature-based approaches require more resources because CNNs are influenced less by data set size. Since CNNs consider images individually, there is a linear relationship with data set size and number of images. Since feature-based algorithms generally consider the whole data set in the aggregate, they scale more steeply than linear with respect to data set size, with SVMs being more than quadratic (Cortes and Vapnik 1995; Pedregosa et al. 2011). Our results clearly show the benefit of larger data sets, although that benefit can only be realized if the algorithm can be successfully trained. In practice, CNNs are also more tractable on larger data sets because GPUs have hundreds or thousands of cores well suited to the types of calculations that CNNs depend upon.

One disadvantage of CNNs is that they currently lack direct interpretability (Zeiler and Fergus 2014). In contrast, statistics can be calculated on a trained RFC model about the relative importance of individual features, and particular values of those features. In a CNN, the first layer of filter weights can be rendered, but the interaction architecture of a CNN causes subsequent layers to lack a straightforward visualization, although this is an area of open research (Castelvecchi 2016).

Optimizing machine learning architectures for plankton classification

Both CNNs and feature-based algorithms require hyperparameter tuning for optimal performance. For our feature-based algorithms, we followed best practices for hyperparameter optimization and found, as previously described in the literature, that attention to the number of estimators and depth for RFC-based approaches (Boulesteix et al. 2012), network size and activation function for MLPs (Haykin 2009), and gamma and regularization for SVMs (Hsu et al. 2003) improves performance. We found increasing the amount of training data improved accuracy. These two conclusions are consistent with an earlier investigation (Ellen et al. 2015).

We trained our CNNs de novo. In some applications of CNNs, starting with a pretrained model could result in faster training times and increased accuracy, as demonstrated on phytoplankton images (Orenstein et al. 2015). In training our CNNs, we followed current best practices for CNNs (Bengio 2012; Smith 2018), although this guidance is evolving rapidly. Notably, we found that dropout (Hinton et al. 2012) produced little-to-no effect on our pixel-based data, and was even detrimental when applied to our context metadata layers. Both types of data set augmentation we evaluated were beneficial. Reflection increased accuracy by 0.34 points with no increase in runtime, while Cyclic Pooling and Rolling (Dieleman et al. 2016*b*) increased accuracy by 1.6 points at a cost of a ~ 4× increase in execution time. Dieleman et al. (2015) first tried the concept of Cyclic Pooling and Rolling on a different class of rotationally invariant images (galaxy morphology). An alternative method of obtaining rotational augmentations by Li et al. (2018) may be more efficient than the one we used from Lasagne (Dieleman et al. 2016*a*). Additionally, iterative augmentation is possible by using early models to iteratively process available unlabeled data in order to harvest additional training images, as described in Luo et al. (2018).

Since CNNs scale better with data set size than feature-based approaches, it is easier to consider more complicated and deeper architectures with them (i.e., Deep Learning). Published CNN benchmarks for image classification have increased from 19-layer networks, to an ensemble of seven separate 22-layer networks, to 152-layer networks, then thousands of layers (Simonyan and Zisserman 2014; Szegedy et al. 2015; He et al. 2016). A particular model called ResNet (He et al. 2015) was applied to plankton by Li and Cui (2016) with modest results, which the authors suggest could be the result of insufficient training images. In limited trials, we modified a version of ResNet (Szegedy et al. 2017) to fit our image dimensions with 24 layers, and it provided an increase of 0.8 points over our five layer Metadata Interaction model on our medium data set, at a cost of ~ 12× longer run time. We tried a 50-layer version of ResNet, and it performed worse than the 24-layer model (0.3 points lower, at a cost of ~ 1.25× longer run time). These preliminary results suggest that the 50-layer network was overfitting, and the 24-layer network is closer to the optimal configuration for our images.

Metadata limitations

Supervised Machine Learning algorithms depend on training data being representative of future samples. For plankton image classification, this guidance is applicable not only for the distribution of the sampled organisms (González et al. 2017), but also for any context metadata used. The term "concept drift" (Widmer and Kubat 1996; González et al. 2017) describes the condition when this future distribution is not stationary. Some of the metadata distributions will drift faster than the images of the individuals themselves, as the population level responses can lag the changes in context measurements. One additional concern is that metadata will be less useful for conditions that are not well represented in the training set; time of day is not informative if all samples are collected at night.

Comments and recommendations

Recommendations

Training sets should, in most circumstances, reflect the proportional distribution of classes. The percentage of marine snow in our ROI imaged in situ exceeds 90%, but our largest data set is only 50% marine snow. We conducted limited evaluations on more unbalanced data and found a small increase in overall performance, but most of that increase was due to higher accuracy on snow only. Accuracy on non-snow classes decreased slightly, while false positives in the snow category increased. Since we will be correcting algorithm errors manually, we found this outcome less desirable than the situation shown in the confusion matrix above, where very few non-snow ROI end up with the label of snow. Many options exist for penalty functions where different types of errors are assigned different costs to create different types of confusion matrices (Elkan 2001), which then further facilitates treatment of larger data sets.

We only present results where each trained model is used to label images independently, but in practice, multiple models can be used simultaneously or sequentially. Combining multiple individual models in an attempt to achieve greater accuracy than any one on its own is called ensembling. Ensembling of feature-based models without metadata on plankton images can be beneficial (Ellen et al. 2015). The concept of ensembling is well accepted, as nearly every major machine learning competition is won by an ensemble of multiple models (Robinson et al. 2017). The dynamics of an ensemble make academic analysis difficult, because the efficacy of each model needs to be examined as well as the effects of interactions between them, but evidence supports their implementation.

Most of our workflow would remain the same regardless of data set size with one exception. Small data sets with low performing models may learn so slowly or erratically as to never finish training. We set a hard limit on the number of epochs as a precaution against incurring computing costs on poorly performing models. Our 40 epoch limit was reached on ~ 20% of small data set trials, ~ 8% of medium data set trials, and ~ 2.5% of large data set trials. If we were doing more exhaustive investigation on smaller data sets, we would resume training the model from the 40^{th} epoch for those trials.

Comments

Our CNNs are significantly smaller with a larger number of training examples, particularly a larger number of training examples per class, than other contemporary evaluations of CNNs with plankton images (Dieleman et al. 2016*b*; Wang et al. 2016; Moniruzzaman et al. 2017; Zheng et al. 2017). Luo et al. (2018) note that "Deep-learning methods require large amounts of training data, and our 42,000 item training set for 108 categories was likely on the low end." The overall quality, resolution, and between-class distinctiveness of our images are similar to previous studies. Based on previous publications, we did not expect our models to perform as well as they did with so few layers and filters. Some preliminary results suggest that our networks with 10 convolutional layers are approaching asymptotic accuracy with respect to CNN complexity.

We outlined our calibration process for both feature-based approaches and CNNs, and found that in many situations, accepted practices do hold (e.g., the importance of hyperparameters for feature-based approach hyperparameters and augmentation for CNNs) but we did find the benefit of dropout to be less significant than previously observed.

We found geometric, geotemporal, and hydrographic metadata to be useful for classification of in situ images for both feature-based and CNN approaches. We found the context metadata to be useful not only as a straightforward augmentation at the end of the CNN architecture, but found other

incorporation strategies to be twice as beneficial for accuracy, in addition to being more computationally efficient.

CNNs are evolving rapidly, with repeated layer substructures (e.g., ResNet), optimization functions, and ensembling techniques as three prominent research areas that will likely boost performance beyond our current results. The four factors that we found to provide the most benefit (data set size, appropriate network depth, data set augmentation, and inclusion of context metadata) were generally additive. We anticipate further advances by optimizing these four factors while also incorporating future structural refinements of Deep Learning methods.

References

Al-Rfou, R., and others. 2016. Theano: A Python framework for fast computation of mathematical expressions. arXiv preprint arXiv:1605.02688. [accessed 2018 June]. Available from https://github.com/Theano/Theano

Benfield, M., C. Schwehm, R. Fredericks, G. Squyres, S. Keenan, and M. Trevorrow. 2003. Measurement of zooplankton distributions with a high-resolution digital, p. 17–30. *In* P. Strutton and L. Seuront [eds.], Handbook of Scaling Methods in aquatic ecology: Measurement, analysis and simulation. CRC Press.

Bengio, Y. 2012. Practical recommendations for gradient-based training of deep architectures, p. 437–478. *In* K.-R. Müller, G. Montavon, and G. B. Orr [eds.], Neural networks: Tricks of the trade. Lecture notes in computer science, v. 7700. Springer.

Blinn, J. 1998. Jim Blinn's corner: Dirty pixels, 1st Edition. Morgan Kaufmann.

Boulesteix, A. L., S. Janitza, J. Kruppa, and I. R. König. 2012. Overview of random forest methodology and practical guidance with emphasis on computational biology and bioinformatics. WIREs Data Mining Knowl Discov **2**: 493–507. doi:10.1002/widm.1072

Bradski, G. 2000. The OpenCV library. Dr. Dobb's J. Soft. Tools **25**: 120–125.

Briseño-Avena, C., P. L. D. Roberts, P. J. S. Franks, and J. S. Jaffe. 2015. ZOOPS-O^2: A broadband echosounder with coordinated stereo optical imaging for observing plankton in situ. Meth. Oceanogr. **12**: 36–54. doi:10.1016/j.mio.2015.07.001

Canny, J. 1986. A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. **PAMI-8**: 679–698. doi:10.1109/TPAMI.1986.4767851

Castelvecchi, D. 2016. Can we open the black box of AI? Nature news. Nature **538**: 20–23. doi:10.1007/BF00994018

Cortes, C., and V. Vapnik. 1995. Support-vector networks. Mach. Learn. **20**: 273–297. doi:10.1007/BF00994018

Cowen, R. K., and C. M. Guigland. 2008. In situ ichthyoplankton imaging system (ISIIS): System design and preliminary results. Limnol. Oceanogr.: Methods **6**: 126–132. doi:10.4319/lom.2008.6.126

Criminisi, A., J. Shotton, and E. Konukoglu. 2012. Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning, p. 81–227. *In* Foundations and trends in computer graphics and vision, v. 7. now Publishers. doi:10.1561/0600000035

Dai, J., R. Wang, H. Zheng, G. Ji, and X. Qiao. 2016. ZooplanktoNet: Deep convolutional network for zooplankton classification. *In* IEEE OCEANS 2016-Shanghai. IEEE. 1804–1809.

Davis, C. S., S. M. Gallager, M. S. Berman, L. R. Haury, and J. R. Strickler. 1992. The video plankton recorder (VPR): Design and initial results. Arch. Hydrobiol. Beih. Ergeb. Limnol. **36**: 67–81.

Dieleman, S., K. W. Willett, and J. Dambre. 2015. Rotation-invariant convolutional neural networks for galaxy morphology prediction. Mon. Not. R. Astron. Soc. **450**: 1441–1459. doi:10.1093/mnras/stv632

Dieleman, S., and others. 2016a. Lasagne: First release. [accessed 2018 September]. Available from http://doi.org/10.5281/zenodo.27878

Dieleman, S., J. De Fauw, and K. Kavukcuoglu. 2016b. Exploiting cyclic symmetry in convolutional neural networks. arXiv preprint. arXiv:1602.02660.

Elkan, C. 2001. The foundations of cost-sensitive learning, p. 973–978. *In* 17th International Joint Conference on Artificial Intelligence, Seattle, 4–10 August 2001.

Ellen, J., H. Li, and M. D. Ohman. 2015. Quantifying California current plankton samples with efficient machine learning techniques, p. 1–9. *In* OCEANS'15 MTS/IEEE, Washington, DC, 19–22 October 2015. IEEE.

Esteva, A., B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun. 2017. Dermatologist-level classification of skin cancer with deep neural networks. Nature **542**: 115–118. doi:10.1038/nature21056

Fernandes, F. 2014. Python-seawater v3.3.2. Zenodo. [accessed 2018 September]. Available from http://doi.org/10.5281/zenodo.11395

Fofonoff, P., and R. C. Millard, Jr. 1983. Algorithms for computation of fundamental properties of seawater, p. 1–53. UNESCO Tech. Pap. Mar. Sci. 44. [accessed 2018 September]. Available from http://unesdoc.unesco.org/images/0005/000598/059832eb.pdf

Freund, Y., and R. E. Schapire. 1997. A decision-theoretic generalization of on-line learning and an application to boosting. J. Comput. Syst. Sci. **55**: 119–139. doi:10.1006/jcss.1997.1504

Friedman, J. H. 2001. Greedy function approximation: A gradient boosting machine. Ann. Stat. **29**: 1189–1232. doi:10.1214/aos/1013203451

Geurts, P., D. Ernst, and L. Wehenkel. 2006. Extremely randomized trees. Mach. Learn. **63**: 3–42. doi:10.1007/s10994-006-6226-1

Glorot, X., and Y. Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks, p. 249–256. *In* Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics.

González, P., E. Álvarez, J. Díez, Á. López-Urrutia, and J. J. del Coz. 2017. Validation methods for plankton image classification systems. Limnol. Oceanogr.: Methods **15**: 221–237. doi:10.1002/lom3.10151

Gorsky, G., and others. 2010. Digital zooplankton image analysis using the ZooScan integrated system. J. Plankton Res. **32**: 285–303. doi:10.1093/plankt/fbp124

Graff, C. A. and J. Ellen. 2016. Correlating filter diversity with convolutional neural network accuracy, p. 75–80. *In* 15th IEEE International Conference on Machine Learning and Applications (ICMLA).

Graves, A., A. R. Mohamed and G. Hinton. 2013. Speech recognition with deep recurrent neural networks, p. 6645–6649. *In* IEEE International Conference on Acoustics, Speech and Signal Processing.

Grosjean, P., M. Picheral, C. Warembourg, and G. Gorsky. 2004. Enumeration, measurement, and identification of net zooplankton samples using the ZooScan digital imaging system. ICES J. Mar. Sci. **61**: 518–525. doi:10.1016/j.icesjms.2004.03.012

Haykin, S. S. 2009. Neural networks and learning machines, v. 3. Pearson.

He, K., X. Zhang, S. Ren, and J. Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification, p. 1026–1034. *In* IEEE International Conference on Computer Vision.

He, K., X. Zhang, S. Ren, and J. Sun. 2016. Deep residual learning for image recognition, p. 770–778. *In* IEEE International Conference on Computer Vision and Pattern Recognition.

Hinton, G. E., N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint. arXiv:1207.0580.

Ho, T. K. 1995. Random decision forests, p. 278–282. *In* Proceedings of the Third International Conference on Document analysis and recognition, v. 1.

Hsu, C. W., C.-C. Chang, and C.-J. Lin. 2003. A practical guide to support vector classification Technical report, Department of Computer Science, National Taiwan University. 1–16.

Hu, Q., and C. Davis. 2005. Automatic plankton image recognition with co-occurrence matrices and support vector machine. Mar. Ecol. Prog. Ser. **295**: 21–31. doi:10.3354/meps295021

Hubel, D. H. 1959. Single unit activity in striate cortex of unrestrained cats. J. Physiol. **147**: 226–238. doi:10.1113/jphysiol.1959.sp006238

Hubel, D. H., and T. N. Wiesel. 1963. Shape and arrangement of columns in cat's striate cortex. J. Physiol. **165**: 559–568. doi:10.1113/jphysiol.1963.sp007079

Kingma, D. P., and M. Welling. 2013. Auto-encoding variational bayes. arXiv preprint. arXiv:1312.6114.

Kingma, D. P., and L. Ba. 2015. ADAM: A method for stochastic optimization. *In* International Conference on Learning Representations.

Krizhevsky, A., I. Sutskever, and G. E. Hinton. 2012. ImageNet classification with deep convolutional neural networks, p. 1097–1105. *In* Advances in Neural Information Processing Systems.

LeCun, Y., and Y. Bengio. 1995. Convolutional networks for images, speech, and time series. Handb. Brain Theory Neural Netw. **3361**: 10.

LeCun, Y., L. Bottou, Y. Bengio, and P. Haffner. 1998. Gradient-based learning applied to document recognition. Proc. IEEE **86**: 2278–2324. doi:10.1109/5.726791

LeCun, Y., and M. Ranzato. 2013. Deep learning tutorial. *In* Tutorials in International Conference on Machine Learning. Atlanta, 16 June 2013.

LeCun, Y., Y. Bengio, and G. E. Hinton. 2015. Deep learning. Nature **521**: 436–444. doi:10.1038/nature14539

Lee, C. Y., S. Xie, P. Gallagher, Z. Zhang, and Z. Tu. 2015. Deeply-supervised nets, p. 973–978. *In* Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics. Proceedings of Machine Learning Research (PMLR). V. 38.

Li, J., Z. Yang, H. Liu, and D. Cai. 2018. Deep rotation equivariant network. Neurocomputing **290**: 26–33. doi:10.1016/j.neucom.2018.02.029

Li, X., and Z. Cui. 2016. Deep residual networks for plankton classification, p. 1–4. *In* OCEANS 2016 MTS/IEEE Monterey.

Li, Y., D. J. Crandall, and D. P. Huttenlocher. 2009. Landmark classification in large-scale image collections, p. 1957–1964. *In* International Conference on Computer Vision.

Lilly, L. E., and M. D. Ohman. 2018. CCE IV: El Niño-related zooplankton variability in the southern California Current System. Deep-Sea Res. Part I Oceanogr. Res. Pap. **140**: 36–51. doi:10.1016/j.dsr.2018.07.015

Lippmann, R. 1987. An introduction to computing with neural nets. IEEE ASSP Mag. **4**: 4–22. doi:10.1109/MASSP.1987.1165576

Luo, J. Y., J.-O. Irisson, B. Graham, C. Guigand, A. Sarafraz, C. Mader, and R. K. Cowen. 2018. Automated plankton image analysis using convolutional neural networks. Limnogr. Oceanogr.: Methods **16**: 814–827. doi:10.1002/lom3.10285

Mantua, N. J., S. R. Hare, Y. Zhang, J. M. Wallace, and R. C. Francis. 1997. A Pacific interdecadal climate oscillation with impacts on salmon production. Bull. Am. Meteorol. Soc. **78**: 1069–1080.

Matsugu, M., K. Mori, Y. Mitari, and Y. Kaneda. 2003. Subject independent facial expression recognition with robust face detection using a convolutional neural network. Neural Netw. **16**: 555–559. doi:10.1016/S0893-6080(03)00115-1

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

Ng, J. Y.-H., M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici. 2015. Beyond short snippets: Deep networks for video classification, p. 4694–4702. *In* Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

NOAA National Center for Environmental Information. 2016. San Diego, California coastal digital elevation model – issued 2012-03-07, updated 2016-07-28. [accessed 2018 June]. Available from https://www.ngdc.noaa.gov/thredds/dodsC/regional/san_diego_13_mhw_2012.nc.html

Ohman, M. D., and others. 2013. Autonomous ocean measurements in the California Current Ecosystem. Oceanography **26**: 18–25. doi:10.5670/oceanog.2013.41

Ohman, M. D., R. E. Davis, J. T. Sherman, K. R. Grindley, B. M. Whitmore, C. F. Nickels, and J. S. Ellen. 2018. *Zooglider*: An autonomous vehicle for optical and acoustic sensing of zooplankton. Limnol. Oceanogr.: Methods **17**: 69–86. doi:10.1002/lom3.10301

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

Orenstein, E. C., O. Beijbom, E. E. Peacock, and H. M. Sosik. 2015. WHOI-plankton-a large scale fine grained visual recognition benchmark dataset for plankton classification. *In* The Third Workshop on Fine-Grained Visual Categorization at CVPR 2015. arXiv preprint arXiv:1510.00745.

Orenstein, E. C., and O. Beijbom. 2017. Transfer learning and deep feature extraction for planktonic image data sets, p. 1082–1088. *In* 2017 IEEE Winter Conference on Applications of Computer Vision (WACV).

Pacific Fisheries Environmental Laboratory. 2018. PFEL upwelling index: Pacific Fisheries Environmental Laboratory. [accessed 2018 June] Available from https://www.pfeg.noaa.gov/products/PFEL/modeled/indices/upwelling/NA/data_download.html

Pedregosa, F., and others. 2011. Scikit-learn: Machine learning in Python. J. Mach. Learn. Res. **12**: 2825–2830.

Peura, M., and J. Iivarinen. 1997. Efficiency of simple shape descriptors, p. 443–451. *In* Aspects of visual form. World Scientific.

Picheral, M., L. Guidi, L. Stemmann, D. M. Karl, G. Iddaoud, and G. Gorsky. 2010. The Underwater Vision Profiler 5: An advanced instrument for high spatial resolution studies of particle size spectra and zooplankton. Limnol. Oceanogr.: Methods **8**: 462–473. doi:10.4319/lom.2010.8.462

Robinson, K. L., J. Y. Luo, S. Sponaugle, C. Guigand, and R. K. Cowen. 2017. A tale of two crowds: Public engagement in plankton classification. Front. Mar. Sci. **4**: 82.

Rumelhart, D. E., G. E. Hinton, and R. J. Williams. 1986. Learning representations by back-propagating errors. Nature **323**: 533–536. doi:10.1038/323533a0

Sabour, S., N. Frosst, and G. E. Hinton. 2017. Dynamic routing between capsules, p. 3856–3866. *In* Advances in Neural Information Processing Systems (NIPS 2017), v. 30, Long Beach, CA, 4–9 December 2017. Neural Information Processing Systems Foundation.

Samson, S., T. Hopkins, A. Remsen, L. Langebrake, T. Sutton, and J. Patten. 2001. A system for high-resolution zooplankton imaging. IEEE J. Ocean. Eng. **26**: 671–676. doi:10.1109/48.972110

Schulz, J., K. Barz, P. Ayon, A. Luedtke, O. Zielinski, D. Mengedoht, and H.-J. Hirche. 2010. Imaging of plankton specimens with the lightframe on-sight keyspecies investigation (LOKI) system. J. Eur. Opt. Soc. Rapid Publ. **5** (10017S): 1–9. doi:10.2971/jeos.2010.10017s

Schwing, F. B., M. O'Farrell, J. M. Steger, and K. Baltz. 1986. Coastal upwelling indices west coast of North America, 1946–95, p. 144. NOAA Tech. Memo. NOAA-TM-NMFS–SWFSC-231. Natl. Oceanic and Atmos. Admin., Natl. Mar. Fish. Serv.

Sieracki, C. K., M. E. Sieracki, and C. S. Yentsch. 1998. An imaging-in-flow system for automated analysis of marine microplankton. Mar. Ecol. Prog. Ser. **168**: 285–296. doi:10.3354/meps168285

Simonyan, K. and A. Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. CoRR, arXiv preprint arXiv:abs/1409.1556.

Smith, L. N. 2018. A disciplined approach to neural network hyper-parameters: Part 1--learning rate, batch size, momentum, and weight decay. arXiv preprint arXiv:1803.09820.

Socher, R., A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank, p. 1631–1642. *In* Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing.

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

Srivastava, N., G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**: 1929–1958.

Szegedy, C., W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. 2015. Going deeper with convolutions, p. 1–9. *In* Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

Szegedy, C., S. Ioffe, V. Vanhoucke, and A. A. Alemi. 2017. Inception-v4, inception-resnet and the impact of residual connections on learning, p. 4278–4284. *In* Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, v. 4.

Tang, K., M. Paluri, F. F. Li, R. Fergus, and L. Bourdev. 2015. Improving image classification with location context, p. 1008–1016. *In* Proceedings of the IEEE International Conference on Computer Vision.

Thompson, C. M., M. P. Hare, and S. M. Gallager. 2012. Semi-automated image analysis for the identification of bivalve larvae from a Cape Cod estuary. Limnol. Oceanogr.: Methods **10**: 538–554. doi:10.4319/lom.2012.10.538

Towns, J., and others. 2014. XSEDE: Accelerating scientific discovery. Comput. Sci. Eng. **16**: 62–74. doi:10.1109/MCSE.2014.80

van Rossum, G. 1995. Python tutorial. Technical report CS-R9526. Centrum voor Wiskunde en Informatica (CWI).

Wagemans, J., J. H. Elder, M. Kubovy, S. E. Palmer, M. A. Peterson, M. Singh, and R. von der Heydt. 2012. A century of Gestalt psychology in visual perception: I. Perceptual grouping and figure–ground organization. Psychol. Bull. **138**: 1172–1217. doi:10.1037/a0029333

Wang, R., J. Dai, H. Zheng, G. Ji, and X. Qiao. 2016. Multi features combination for automated zooplankton classification, p. 1–5. *In* IEEE OCEANS 2016-Shanghai.

Watson, J. 2004. HoloMar: A holographic camera for subsea imaging of plankton. Sea Technol. **45**: 53–55.

Wertheimer, M. 1923. Untersuchungen zur Lehre von der Gestalt II. Psycologische Forschung **4**: 301–350.

Widmer, G., and M. Kubat. 1996. Learning in the presence of concept drift and hidden contexts. Mach. Learn. **23**: 69–101. doi:10.1007/BF00116900

Wilkins, M. F., L. Boddy, C. W. Morris, and R. Jonker. 1996. A comparison of some neural and non-neural methods for identification of phytoplankton from flow cytomery data. Bioinformatics **12**: 9–18. doi:10.1093/bioinformatics/12.1.9

Zeiler, M. D. and R. Fergus. 2014. Visualizing and understanding convolutional networks, p. 818–833. *In* European Conference on Computer Vision.

Zheng, H., R. Wang, Z. Yu, N. Wang, Z. Gu, and B. Zheng. 2017. Automatic plankton image classification combining multiple view features via multiple kernel learning. BMC Bioinf. **18**: 570. doi:10.1186/s12859-017-1954-8

Acknowledgments

JSE expresses particular thanks to Charles Elkan for scientific guidance and constructive commentary. Tristan Biard, Laura Lilly, Catherine Nickels, Linsey Sala, Stephanie Sommer, Emma Tovar, and Ben Whitmore conducted numerous hours of image identifications and validations, without which these experiments would not have been possible. This research was possible because of support by the Office of Naval Research and SPAWAR Systems Center San Diego via the SMART scholarship program. The Gordon and Betty Moore Foundation funded the development of the *Zooglider* that collected these images. The Scripps Institution of Oceanography's Instrument Development Group developed *Zooglider.* This research was conducted using the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number ACI-1548562 for machine learning experimentation. Most results were obtained using Comet from the San Diego Supercomputing Center and Bridges from the Pittsburg Supercomputing Center through XSEDE allocations TG-OCE150020 and TG-OCE160022. Machine Learning small scale experimentation used a Tesla K40 GPU donated by the NVIDIA Corporation. The National Science Foundation-supported *California Current Ecosystem* Long Term Ecological Research (CCE-LTER) site also provided indirect support.

Conflict of Interest

None declared.

Submitted 24 January 2019
Revised 11 May 2019
Accepted 13 June 2019

Associate editor: Malinda Sutor