Toward Semi-Supervised Classification of Underwater Benthic Habitat Imagery

Isaac Xu
Faculty of Computer Science
Dalhousie University
Halifax, Canada
isaac.xu@dal.ca

Thomas Trappenberg
Faculty of Computer Science
Dalhousie University
Halifax, Canada
tt@cs.dal.ca

Scott C. Lowe
Faculty of Computer Science
Dalhousie University
Halifax, Canada
scott.lowe@dal.ca

***Abstract*—As part of the Benthic Ecosystem Mapping and Engagement (BEcoME) project, we are working toward automating underwater image classification using modern semi-supervised learning approaches. To begin this work, we tested the Bootstrap Your Own Latent (BYOL) model on the well known MNIST dataset. Our interest is in datasets where only a minority of the samples are labelled, a problem typical for many real world datasets, which we simulated by redacting the labels from part of the MNIST dataset. We find that the semi-supervised methodology is more resilient against a decrease in the number of labelled training samples than a fully-supervised model trained only on the labelled images. When very few of the samples were labelled ($<$0.8%), there is an appreciable performance advantage to the semi-supervised models when compared with a purely supervised model.**

***Index Terms*—machine learning, artificial intelligence, neural networks, semi-supervised learning, image classification, underwater imagery**

I. Introduction

Benthic habitats are seafloor environments which are representative of the diverse and sensitive nature of underwater ecologies [1]. Benthic habitat mapping is then the process of determining the locations and spatial extent of such seafloor landscapes. It is a vital process in understanding the impacts of human activity and climate change on such habitats and assessing how best to manage and preserve fragile ocean ecosystems. During the benthic habitat mapping process, underwater images of the ocean floor are collected and manually annotated by expert ocean scientists. This annotation step is the most time-consuming (and least intellectually stimulating) stage of the habitat mapping pipeline. As part of the Benthic Ecosystem Mapping and Engagement (BEcoME) project[1], funded by Ocean Frontier Institute, we are working to use machine learning techniques to help automate this process.

Underwater images of the seafloor can be collected for many reasons, and only a small fraction of these images are annotated with a habitat classification by marine scientists. The problem at hand is that the limited amount of labelled data is often insufficient to train a reasonably performing supervised model. Since there exist large datasets of unlabelled images that are readily available, a logical approach is thus to harness this unlabelled data when training our model. One approach in addressing this problem is to utilize self-supervised learning.

Recent advancements in self-supervised learning have led to the creation of models which can match the performance of fully supervised models on large-scale image classification tasks [2] [3] [4]. Additionally, some methods such as Bootstrap Your Own Latent (BYOL), which is the method chosen in this paper, use an approach that does not require negative samples. This approach simplifies the learning process, enabling training with smaller batch sizes that can be deployed within our computational constraints [4].

Our goal is to collate a large body of seafloor imagery that is publicly available on the web and train a large model on this dataset using self-supervised learning. We will then fine-tune this model by training on the significantly smaller labelled dataset to yield a classification model. We expect this will produce a model which performs well and can generalize much better than a model trained solely on the available labelled data.

Currently, we are collecting the data to fuel this model by harvesting publicly available datasets and soliciting contributions from collaborators. Meanwhile, we have deployed a BYOL [4] model on the MNIST dataset [5] to evaluate its effectiveness. The MNIST dataset was chosen for its simplicity and familiarity, allowing for an example dataset to which we can quickly modify and test models for. The goal here is to understand how BYOL performance begins to deteriorate with gradually decreasing numbers of labelled data samples, thus establishing a clearer idea of both the model's limitations as well as its comparative performance to that of supervised learning.

II. Related Works

The original BYOL paper [4] contains fine-tuning comparisons of BYOL against SimCLR [2] and supervised learning on the ImageNet dataset. The group compared representations over 1%, 2%, 5%, 10%, 20%, 50% and 100% of data. According to their testing, BYOL outperformed the alternative models at every stage. In our experiment, we are largely seeking to reproduce and confirm BYOL's performance capability. This understanding of the model we obtain would be necessary for our eventual implementation of it for the BEcoME dataset.

[1]https://www.ofibecome.org/

Another point of interest is in the trend and deterioration of BYOL's linear projection performance when compared to that of supervised learning. While it is not the primary focus of this paper, when graphed against number of labelled data samples, we believe that the location of the intersect between a semi-supervised linear projection performance and that of supervised learning provides an indicator of the classification difficulty for a particular dataset. A standardized model using such a metric would be useful in providing an intuition for how well a model should be expected to perform at classifying this specific dataset.

III. Methods

The BYOL model [4] uses two asymmetric architectures known as online and target models. The online model consists of a convolutional neural network (CNN) encoder as well as a multi-layer perceptron (MLP) projector and a MLP predictor. The target model has only the encoder and the projector, which take their weights as the exponential moving average (EMA) of those from the online model. The core concept behind BYOL is to train an online model capable of predicting the embedding vector generated by the target model when they are each presented with differently augmented samples of the same image. We initially attempted to examine how BYOL would perform as we limited the amount of labelled data available for fine-tuning. Our goal was to estimate BYOL's performance when subjected to constraints in real world datasets such as our underwater imagery data.

Our BYOL implementation was implemented in PyTorch [6]. As described above, there are three main parts to the BYOL model: an encoder, a MLP structure (used for the projector and predictors) and finally a classifier, which is a single linear layer.

The encoder is made up of of six convolutional blocks each consisting of a 2-D convolution with a 3×3 kernel with "same" padding, batch normalization, and activation with a rectified linear unit (ReLU). We downsample with a stride of two on every alternative convolutional layer. Finally, we take the average of each of the feature maps across 256 output channels in the form of a global average pooling layer [7].

The projector and predictors both consist of two linear layers, with batch normalization and ReLU activation in-between. Finally, the classifier consists of a single linear layer and a softmax output. The full network architecture is shown in Figure 1.

For the first part of our experimentation with MNIST, we trained the network on the full 60,000 MNIST dataset for 5 epochs using self-supervised learning. Although we used augmentations such a random crop and Gaussian blurring such as those used in SimCLR [2], we could not use SimCLR's colour-based augmentations due to the greyscale nature of MNIST images. The online and target networks are fed two such independently augmented copies of the same image sample. We use cosine similarity with the embedding vector of the target model as the utility function for evaluating the online model's output. For symmetry, the two augmented images are

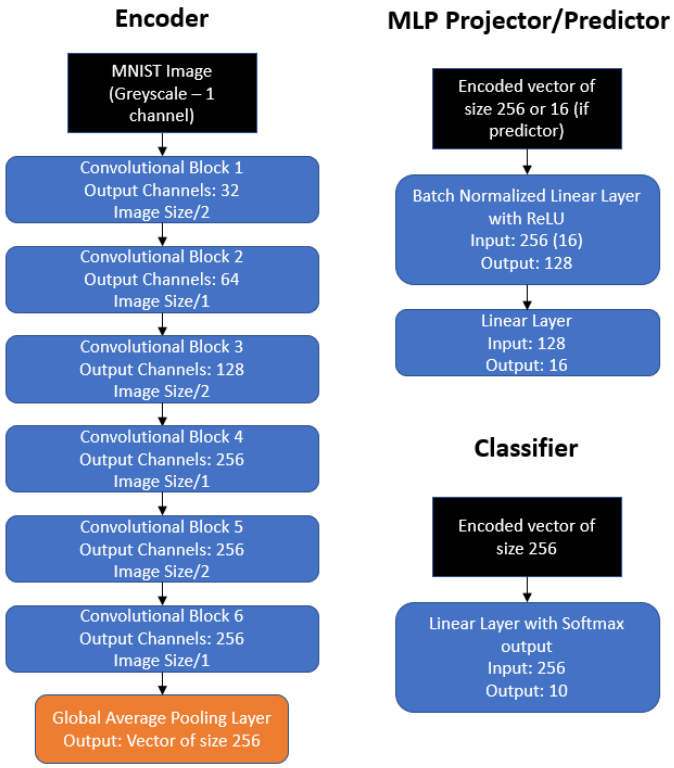

Fig. 1. **Network architecture.** The encoder maps an input image to a size 256 embedding space. It does so by gradually expanding the number of channels and downsampling the image by 1/2 on odd-numbered convolutional blocks. The projector and classifier are then compatible with the size 256 vector outputted by the encoder and map it to a size 16 vector or a size 10 softmax vector respectively. Finally, the predictor is intended to stack on top of the projector and therefore maps a size 16 vector to another size 16 vector after processing through its hidden layer.

presented both ways around. The cosine similarity is measured for each presentation, summed, and and negated to provide our loss function, which is backpropagated through the online network. The online network is updated using the Adam [8] optimizer, with a momentum of 0.9, weight decay 1.5×10^{-6} and learning rate 0.01. The target model is not updated by the backpropagation step; this is instead updated after each batch to be the EMA of the encoder and projector weights from the online model.

After the self-supervised training process, we discard the projector and predictor and keep only the trained encoder module. This network is used to initialize our classifier model.

We compare four classifier training methods: fully-supervised (Sup.), linear projection (Linear Probe), fine-tuning (Fine Tune), and transfer learning (Trans.). When training these models on the labelled partition of the dataset, we used the Adam optimizer with the same parameters described above for the self-supervised training step, except for the learning rate, which was reduced to 0.001 Our fully-supervised learning consisted of training an identical encoder-classifier architecture from scratch on the labelled dataset only. This model serves as a baseline to compare our other experiments against. The linear projection model was built by adding a single trainable

linear layer on top of the pre-trained encoder from the self-supervision step. The fine-tuning approach was to take the output from the linear projection model, unfreeze the encoder and train the whole network with a reduced learning rate of 0.0001. Finally, our transfer learning model was initialized by taking the trained encoder, adding an untrained linear layer, and training the entire model.

We performed experiments with a varying number of labelled data available for the models (60,000, 6,000, 600, 500, 400, 300, 200, 150, or 100 images). Because the number of iterations per epoch decreases as the dataset size shrinks, we increased the number of epochs as the number of labelled images was reduced. These models were tested for 40, 100, 500, 600, 750, 900, 1050, 1250, and 1500 epochs, respectively. The number of epochs for each set of labelled images was determined through some experimentation and set to be approximately where the accuracy of the model had largely peaked.

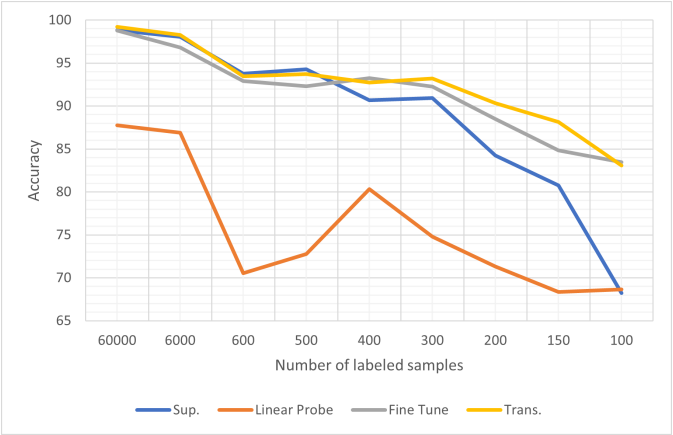

Fig. 2. **Model performance.** Classification accuracy against the number of labelled images used for training. Performance for models initialized with a self-supervised model (Linear Prob, Fine Tune and Trans.) decrease slower than the fully supervised model (Sup.).

IV. Results

As shown in Figure 2, the performance of all models decrease as the number of labelled samples decreases. However, the transfer learning (Trans.) and fine-tuning (Fine Tune) models decrease in performance slower than the fully-supervised (Sup.) model. The linear projection (Linear Probe) model also decreases in performance at a slower rate, but has generally lower accuracy than the other models. With large datasets, it is difficult to discern any difference in performance between the supervised learning, transfer learning, and fine-tuning models, but a gap between the models becomes clear with 400 (0.7%) or fewer labelled images. Although the general trend and proof-of-concept for the model can be established, more trials would be required to obtain a statistically significant understanding of the nuances between these models.

TABLE I
Accuracy of models. We report the test accuracy while reducing the number of labelled training samples.

Num. labelled	Epochs	Test accuracy (%) Sup.	Linear Probe	Fine Tune	Trans.
60,000	40	98.82	87.77	98.80	99.23
6,000	100	98.06	86.90	96.79	98.26
600	500	93.77	70.54	92.89	93.46
500	600	94.28	72.76	92.29	93.71
400	750	90.67	80.32	93.26	92.72
300	900	90.92	74.77	92.27	93.21
200	1,050	84.23	71.31	88.48	90.31
150	1,250	80.77	68.35	84.83	88.15
100	1,500	68.24	68.67	83.47	83.08

V. Discussion

Our expectations are that given a large labelled dataset, semi-supervised learning would be comparable to that of fully supervised-learning. More interesting is the point at which semi-supervised training significantly surpasses supervised learning. For our experiments on MNIST, this threshold appears to be around 400–500 labelled images or 0.6% to 0.8% of the 60,000 self-supervised samples. Previous work has found that a semi-supervised BYOL model was able to surpass a fully-supervised model on ImageNet when the labelled dataset was 10% of the samples (120k images) [4]. This begs the question of why our experiments with MNIST appear to have a lower threshold for surpassing supervised learning when compared to the ImageNet experiments in the original paper. One possibility for such a large difference in the effect of labelled data size may be due the relative difficulty of the classification task. Because MNIST is an easier dataset to classify (a model does not need many examples to learn to classify digits), the representational learning conducted by the semi-supervised approach did not provide a noticeable advantage until the training samples were significantly reduced. In this sense, comparing supervised learning with semi-supervised techniques may act as an empirical and de facto method for assessing the difficulty of a particular classification task.

Another interesting point is the fact that while transfer learning and fine-tuning highlight the effectiveness of learned self-supervised representations, the linear probe highlights its limitations. From this observation, it seems that although the learned embedding provides a strong starting point for the model, it is relatively insufficient by itself. One possible reason for this may be that the augmentations we applied were not sufficiently challenging for the model, and the model may have solved the problem with a short-cut that did not require it to learn a strong embedding space. Indeed, SimCLR discovered that colour augmentations were the most important augmentation for RGB images [2], with a large decrease in performance if they were not present. Since MNIST data is greyscale, we were unable to use augmentations across the colour channels. Alternative augmentations we could try include contrasting and brightening, which can be performed by offsetting and scaling the normalized intensity values, or inverting the image. Furthermore, as stated in [2], additional

augmentations such as Sobel filtering or motion blurring could also benefit the learned encodings, as these were found to improve results with ResNet-50 models on ImageNet.

In general, there appears to be noticeable benefits to a semi-supervised approach for datasets with limited labelled samples. There are many directions to proceed from the results demonstrated here. We would be interested in looking at how class imbalance could impact semi-supervised methods. Another project of interest could be to establish a metric of how difficult a classification task may be, through tested performances with semi-supervised and supervised approaches. Ultimately, we hope to apply the lessons learned here towards achieving better results in the classification of underwater imagery as part of the BEcoME project.

REFERENCES

[1] P. T. Harris and E. K. Baker, "1 - why map benthic habitats?" in *Seafloor Geomorphology as Benthic Habitat*, P. T. Harris and E. K. Baker, Eds. London: Elsevier, 2012, pp. 3–22. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780123851406000013

[2] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," 2020.

[3] X. Chen and K. He, "Exploring simple siamese representation learning," 2020.

[4] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. D. Guo, M. G. Azar, B. Piot, K. Kavukcuoglu, R. Munos, and M. Valko, "Bootstrap your own latent: A new approach to self-supervised learning," 2020.

[5] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[6] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," 2019.

[7] M. Lin, Q. Chen, and S. Yan, "Network in network," 2014.

[8] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.