Label-free Monitoring of Self-Supervised Learning Progress

Isaac Xu
Faculty of Computer Science
Dalhousie University
Halifax, Canada
isaac.xu@dal.ca
0000-0003-4443-0582

Scott Lowe
Faculty of Computer Science
Dalhousie University
Halifax, Canada
scott.lowe@dal.ca
0000-0002-5237-3867

Thomas Trappenberg
Faculty of Computer Science
Dalhousie University
Halifax, Canada
tt@cs.dal.ca
0000-0002-6144-8963

2022 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE) | 978-1-6654-8432-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/CCECE49351.2022.9918377

***Abstract*—Self-supervised learning (SSL) is an effective method for exploiting unlabelled data to learn a high-level embedding space that can be used for various downstream tasks. However, existing methods to monitor the quality of the encoder — either during training for one model or to compare several trained models — still rely on access to annotated data. When SSL methodologies are applied to new data domains, a sufficiently large labelled dataset may not always be available. In this study, we propose several evaluation metrics which can be applied on the embeddings of unlabelled data and investigate their viability by comparing them to linear probe accuracy (a common metric which utilizes an annotated dataset). In particular, we apply k-means clustering and measure the clustering quality with the silhouette score and clustering agreement. We also measure the entropy of the embedding distribution. We find that while the clusters did correspond better to the ground truth annotations as training of the network progressed, label-free clustering metrics correlated with the linear probe accuracy only when training with SSL methods SimCLR and MoCo-v2, but not with SimSiam. Additionally, although entropy did not always have strong correlations with LP accuracy, this appears to be due to instability arising from early training, with the metric stabilizing and becoming more reliable at later stages of learning. Furthermore, while entropy generally decreases as learning progresses, this trend reverses for SimSiam. More research is required to establish the cause for this unexpected behaviour. Lastly, we find that while clustering based approaches are likely only viable for same-architecture comparisons, entropy may be architecture-independent.**

***Index Terms*—computer vision, machine learning, self-supervised learning, clustering representations**

I. Introduction

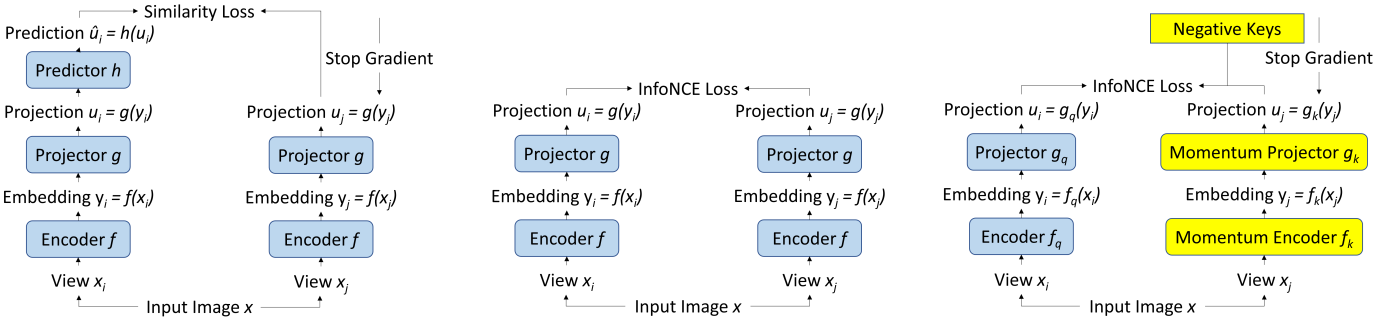
For many specialized fields seeking to deploy deep learning models, generating labels to produce viable datasets for supervised machine learning can be a costly process in terms of time and expertise. Taking advantage of unlabelled data, recent self-supervised learning (SSL) methods have achieved state of the art performance as a means for extracting high-level features from complex inputs such as imagery [1]–[4]. These SSL methods have mainly been evaluated on labelled data. In this work, we propose and evaluate metrics to monitor learning and the performance of the SSL models without annotations.

In computer vision, an encoder model maps an input image from pixel-space to a lower dimensional representational space as an embedding vector which captures high-level contextual and semantic information in the image. With supervised learning, the encoder is trained as part of the classification model. Then, through transfer learning, additional neural network "head" layers can be appended onto the encoder to interpret vectors in this embedding space for the purpose of other downstream tasks.

In SSL, the encoder can be trained with labels generated from the data itself via a "pretext task", meaning expensive human-annotation is not required. An encoder trained with SSL can learn a more robust and generalizable embedding space than those generated from a supervised learning process [5], [6]. Recently, instance learning has been demonstrated to be a successful SSL pretext task. In this method, the representational distance between independently augmented views of the same sample is minimized [3], [4]. A subset of instance learning known as contrastive learning, also maximizes the distance between views of different samples [1], [7].

A common evaluation method for models prepared with SSL is to train a linear read-out layer on top of the encoder on a supervised classification task [1]–[4], [7]. This process is referred to as a linear probe (LP). Another increasingly popular method is to pass a dataset through the encoder and use a k-nearest neighbours (kNN) classifier [2], [8]. Since the distance between sample embeddings in the representational space carries semantic meaning, a sample's class can be predicted using the classes of its top k nearest neighbours. Although these methods only measure the performance on one downstream task (whole-frame classification), their performance is indicative of the utility of the embedding space on other tasks. However, these methods still require labelled test datasets, which can be challenging to acquire.

Our proposal for monitoring learning progress without labels is based on the conjecture that due to the semantic meaning in learned embedding space, similar samples should increasingly group together over learning. Hence, a core part of our evaluation approach relies upon employing k-means clustering on sample embeddings and characterization of the clusters using traditional clustering metrics. We also look to the agreement between two independent k-means clustering attempts to provide an intuition for clustering consistency. We expect that the more well-formed ground truth clusters are,



(a) SimSiam. (b) SimCLR. (c) MoCo-v2.

Fig. 1: Overview of SSL methodologies. For a given image, x, two views of the image (x_i and x_j) are created with randomly sampled augmentations. In SimSiam (a) and SimCLR (b), the two views are passed through the same encoder, f, and projector, g, layers; in MoCo-v2 (c), view x_j passes through a moving average model instead. For SimSiam, the loss is the similarity between the output of a predictor head on the x_i branch and the projection of x_j; whereas for SimCLR and MoCo-v2, InfoNCE is used with negative samples drawn from the batch (SimCLR) or the memory queue (MoCo). In SimSiam and MoCo-v2, a stop-gradient is applied to prevent the loss returning down the x_j branch of the network.

the more consistently k-means would capture these clusters. Lastly, we measure the entropy of the data embeddings, as we hypothesise that entropy will decrease during training when we progress from an initial high-dispersion and low-modal distribution to a more densely packed higher-modal distribution.

We compare these results to LP accuracy, the current standard measure of model quality. A stronger correlation with LP accuracy indicates greater viability for the label-free metric. We apply our proposed metrics on three SSL techniques: SimSiam [3], SimCLR [1], and MoCo-v2 [7]. Our experiments were performed on CIFAR-10 and CIFAR-100 [9] datasets.

Additionally, we applied these metrics to a variety of architectures pre-trained with supervised learning on ImageNet [10]. These were ResNet [11], EfficientNet [12], and DenseNet [13] models provided in torchvision [14]. Here, we examine if these embedding measures are capable of distinguishing between the quality of models using different architectures.

II. Background and Related Work

A. Instance learning

The SSL methods we use employ strong augmentations to create a pretext task. Each sample from a particular dataset is augmented twice independently, producing two "views" of the sample. The encoder is motivated to generate embeddings which are robust against the augmentation operation, such that the independent views for the same sample have similar embeddings.

1) SimSiam: With the SimSiam training configuration [3], both views (i and j) pass through the encoder, plus a multi-layer perceptron (MLP) section dubbed the projector. One view passes through another MLP dubbed the predictor. The network is trained to minimise the distance between the projected representation of one view, $\mathbf{u}_i$, and the prediction generated from the second view, $\hat{\mathbf{u}}_j$, as given by

$$L_{i,j} = -\sqrt{\mathrm{d}_{\cos}(\mathbf{u}_i, \hat{\mathbf{u}}_j) + \mathrm{d}_{\cos}(\mathbf{u}_j, \hat{\mathbf{u}}_i)}, \tag{1}$$

where $\mathrm{d}_{\cos}(\cdot,\cdot)$ is the cosine similarity distance measure. In order to prevent collapse to a trivial solution, updates to minimize this loss propagate back only through the prediction branch. We can interpret this process as tasking the model to predict the average embedding (in the projector space) over all views of a sample, when presented with any one view of that sample. The SimSiam method is displayed in Figure 1a.

2) SimCLR: Meanwhile, the SimCLR approach uses contrastive learning, as seen in Figure 1b. Views of other same batch samples are used as negative views and the network must learn embeddings such that two views of the same sample (positive views) are close together, whilst negative views are far apart. In the SimCLR formulation, the predictor head is not used and we only need consider the projected view, $\mathbf{u}$. For a given pair of positive views, i and j, in a minibatch of N samples, the loss is

$$L_{i,j} = -\log \frac{\exp\left(\mathrm{d}_{\cos}(\mathbf{u}_i, \mathbf{u}_j)/\tau\right)}{\sum_{k=1}^{2N}(1-\delta_{ik})\exp\left(\mathrm{d}_{\cos}(\mathbf{u}_i, \mathbf{u}_k)/\tau\right)}, \tag{2}$$

where δ is the Kronecker delta function and τ is a temperature scaling factor [1]. This loss is also referred to as the normalized temperature-scaled cross entropy loss [1] or as InfoNCE [15].

3) MoCo-v2: The final method MoCo-v2, seen in Figure 1c, shares the InfoNCE loss function with SimCLR. The main difference between the two methods lies in the use of a momentum or "key" encoder and a memory queue for the projections of negative views, referred to as negative keys. Here, the main encoder is referred to as a query encoder. The key encoder parameters are updated as the exponential moving average (EMA) of the query encoder's parameters at every batch. The views processed by the key encoder are then

added to the queue of negative keys. The objective is then to compare the "query" view projections against the positive keys (same sample view projections) and the negative keys.

B. Extrinsic metrics

Extrinsic metrics are measures of clustering quality based on an external reference. In this work, we use the mutual information [16] between the cluster labels generated by k-means and the ground truth class labels, acting as a benchmark to compare label-free metrics against. We can imagine the class and cluster labels to be two discrete random variables. For discrete random variables X and Y, their mutual information is defined as

$$\mathrm{I}(X;Y) = \sum_{y\in Y}\sum_{x\in X} P(x,y)\log\left(\frac{P(x,y)}{P(x)P(y)}\right), \quad (3)$$

where $P(x,y)$ denotes the joint probability distribution for X and Y, while $P(x)$ and $P(y)$ denotes their respective marginal distributions. Intuitively, we can conceptualize mutual information as a measure of how much information we can obtain about Y upon observing X and vice versa.

For implementation, we use the adjusted mutual information (AMI) score from the scikit-learn library [17]. The AMI corrects for the chance level of mutual information which would be measured given a certain finite number of samples. The AMI between two discrete random variables X and Y is defined as

$$\mathrm{AMI}(X;Y) = \frac{\mathrm{I}(X;Y) - \mathbb{E}[\mathrm{I}(X^*;Y^*)]}{(\mathrm{H}(X)+\mathrm{H}(Y))/2 - \mathbb{E}[\mathrm{I}(X^*;Y^*)]}, \quad (4)$$

where H is entropy and the expected information is taken over a hypergeometric model of X and Y.

C. Intrinsic measures

Intrinsic measures of clustering quality consider properties of the clusters, such as inter-cluster and intra-cluster distances. The silhouette score is one such measure [18]. For each sample, i, the silhouette score is defined as

$$S_i = \frac{b_i - a_i}{\max(a_i, b_i)}, \quad (5)$$

where b_i is the average inter-cluster distance between sample i and the nearest neighbouring cluster's samples, while a_i is the average intra-cluster distance from sample i to other same cluster samples. The individual silhouette scores are averaged over all samples to provide an overall silhouette score.

Our work differs from deep clustering [19] in that rather than incorporating clustering into the training process, we are looking to evaluate a trained model (of varying degrees) using clustering. Furthermore, a major goal is to dissociate model evaluation from label schemes and therefore we cannot rely upon the use of extrinsic measures which take into account ground truths. We use such extrinsic measures only as a means of comparison against investigated label-free methods.

III. Methods

A. Datasets

Our networks were trained on either the CIFAR-10 or CIFAR-100 dataset [9]. These datasets consist of 32×32 pixel natural RGB images, with either 10 or 100 classes, respectively. All training was performed on the training partition and representations were evaluated using the test partition of these datasets.

B. Self-supervised learning

We used the modified ResNet-18 [11] backbone from SimCLR [1] for CIFAR-10 images, with a 512 dimensional representation space. During SSL, we augmented the images to create pairs of views using the CIFAR-10 augmentation stack from SimCLR [1]. When training SimSiam, we added a three layer projector and a two layer predictor, with a width of 2048 throughout except for the bottleneck layer of the predictor, which had a width of 512. For SimCLR, we instead added a two-layer projector, with hidden dimension 2048 and output dimension 128. These are also the same dimensions we use for the MoCo architectures.

The networks were trained with SSL using stochastic gradient descent (SGD) using a one-cycle learning rate schedule [20] with cyclic momentum from 0.85 to 0.95. For SimSiam, the peak learning rate $\eta = 0.06$, weight decay $\lambda = 5\times10^{-4}$, and we trained the network for 800 epochs. For SimCLR, $\eta = 0.5$, $\lambda = 1\times10^{-4}$, temperature $\tau = 0.5$, and we trained the network for 1000 epochs. Lastly, for MoCo-v2, $\eta = 0.06$, $\lambda = 5\times10^{-4}$, $\tau = 0.1$, queue length was 4096, EMA multiplier $m = 0.99$, and the network was trained for 800 epochs.

C. Representation evaluation

Every 20 epochs of the SSL process, we created a checkpoint of the model, referred to as a "milestone". For each milestone, we passed the test partition samples through the encoder backbone to generate a set of 512-d embedding vectors, Z_{512}. Typical clustering methods do not work well on large representation spaces [21], so we reduced Z_{512} down to a 3-d space, using uniform manifold approximation and projection (UMAP) [22] with $n = 50$ neighbours. We then clustered the 3-d UMAP projections of the embedding vectors Z_3, using k-means with the cosine distance metric and $k_1 = 10$ or $k_1 = 100$ as per the number of annotated classes in the dataset. The cluster labels generated from this clustering are referred to as C_1.

We evaluated the quality of C_1 by measuring the amount of information about the ground truth labels C_{GT}, contained in C_1 using $\mathrm{AMI}(C_1; C_{\mathrm{GT}})$. This serves as a baseline to compare our other metrics against.

We defined S_1 as the silhouette score for the clusters C_1, evaluated by Euclidean distance for the original embedding vectors Z_{512}. We similarly defined S_{GT} as the silhouette score of C_{GT}, also for Z_{512}. This measurement provides an upper-bound on the *utility* of S_1 — that which could be obtained with "perfect" cluster assignments.

Finally, we used k-means again to generate a second set of clusters, C_2, with double the classes: $k_2 = 2k_1$. We then defined the *clustering agreement* as the adjusted mutual information between these two sets of clusters, $\mathrm{AMI}(C_1; C_2)$.

To measure the entropy of the embedding space, we take the Z_3 vectors and bin the values along each dimension to yield a 3-d histogram. The bin width is set separately for each dimension as $l_i = 0.4\,\sigma_i$, where σ_i is the standard deviation of the Z_3 vectors in dimension i. The 3-d histogram bin counts are divided by the total number of test samples, to yield an empirically observed probability distribution, from which we measure the entropy.

D. Pre-trained models

We loaded pre-trained models provided in torchvision [14], which had been trained on ImageNet-1k. We used models with ResNet, DenseNet, and EfficientNet architectures of varying sizes: ResNet 18, 34, 50, 101, and 152; DenseNet 121, 161, 169, and 201; EfficientNet (v1) sizes b0 through b7. The same methodology as described above was applied, using the pre-trained model as a (frozen) encoder, with the following changes. (1) CIFAR-10 and -100 images were upscaled to the same resolution as that which the model was trained on. (2) Due to the significantly larger image resolutions for the bigger EfficientNet models, we reduced all batch sizes to 48. (3) The maximum learning rate was reduced to 0.003 for LPs. (4) The entropy bin width was increased to $l_i = 0.8\,\sigma_i$, because the distribution in the representation space was found to be up to twice as large for SL pre-trained models when compared with SSL.

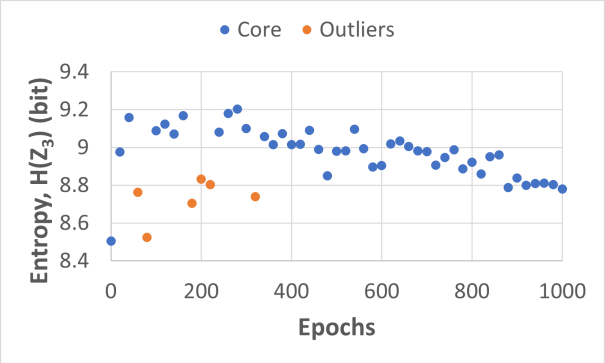


Fig. 2: Entropy progression w.r.t. training for SimCLR on CIFAR-100. We highlight milestones with outlying samples in the embedding space (orange).

E. Linear probe

For each milestone, we extracted the encoder from the network and froze its weights. We added a linear layer on top of the encoder and used the training partition to learn a linear mapping from the embedding space to the target labels. The linear layer was trained with the Adam optimizer [23], for 20 epochs, using a one-cycle learning rate schedule with peak learning rate of 0.08, cyclic momentum from 0.85 to 0.95, and weight decay of 1×10^{-4}. During LP training, we used the augmentations from the CIFAR-10 policy discovered by AutoAugment [24].

IV. Results

We measured the Pearson correlation over training milestones between our metrics and the LP accuracy, the results of which are shown in Table I. Similarly, we measure the correlation for the torchvision pre-trained models separately for each architecture type and in a general "overall" case, presented in Table II. The overall case is intended to be a measure of how feasible a metric may be for cross-architecture comparison. The notations used are as in the previous section, with subscripts GT indicating ground truth labels and 1 indicating clusters C_1. Only metrics pertaining to C_1 require k-means clustering to be performed.

We observe that $\mathrm{AMI}(C_1, C_{\mathrm{GT}})$ correlates strongly with LP accuracy throughout all combinations of methods and datasets. The metric also performs well across the pre-trained models but could not be extended to cross-architecture evaluation, as evidenced by its weak correlation in the overall case. This result demonstrates that clustering is indeed able to progressively pick out ground truth classes as clusters, reflective of learning progression with the likely caveat of being limited to same-architecture comparisons. Although the silhouette score S_{GT} remains near zero ($|S_{\mathrm{GT}}| < 0.05$) and assigns a poor score to the ground truth interpreted as clusters, we find it is well correlated with the LP measurements in SimSiam and SimCLR cases. However, it appears to be inconsistent for MoCo-v2 as well as potentially for EfficientNet and DenseNet architectures, suggesting that the embedding clustering shapes even when representing ground truth, may not progress in a manner reflective of learning progression.

Our results show that the label-free metrics we investigated only present weak correlations with the label-based scores on average. More specifically, the silhouette score S_1 did not correlate well with LP accuracy. As is the case with its ground truth counterpart, it also assigned poor scores to minimally separated clusters. We also found that there could be a large change in the correlation if the initial network state is dropped from the correlates. Due to the non-linearity of learning progression, equally spacing out milestones in terms of epochs can lead to a relatively lower number of data points reflective of early training. As such, early training progression which may differ in nature from later training progression, may not be as sufficiently captured.

Although clustering agreement $\mathrm{AMI}(C_1, C_2)$, was correlated with LP accuracy for SimCLR and to a lesser extent, MoCo-v2, it was either not correlated or inversely correlated when training with SimSiam. This observation may be tied to the unexpected result that entropy increases for SimSiam but decreases for SimCLR. We hypothesis that this may be because SimSiam is more susceptible to dimensional collapse [25], [26]. In future work, this could be confirmed by observing the singular value spectra of the projector and predictor embedding spaces.

TABLE I: Pearson correlation between performance metrics and linear probe accuracy. We display correlation scores both with (w/ init) and without (w/o) the network initialization (i.e. before training begins) milestone included in the trend. Grey: no significant correlation ($p < 0.05$). Black: positively correlated. Red: negatively correlated.

		SimSiam				SimCLR				MoCo-v2			
		CIFAR-10		CIFAR-100		CIFAR-10		CIFAR-100		CIFAR-10		CIFAR-100	
Metric	Label-free	w/ init	w/o	w/ init	w/o	w/ init	w/o	w/ init	w/o	w/ init	w/o	w/ init	w/o
$\mathrm{AMI}(C_1, C_{\mathrm{GT}})$	✗	0.97	0.96	0.98	0.98	0.93	0.97	0.93	0.97	0.92	0.95	0.88	0.91
$\mathrm{AMI}(C_1, C_2)$	✓	0.00	−0.21	−0.62	−0.71	0.71	0.91	0.76	0.90	0.64	0.80	0.47	0.61
S_{GT}	✗	0.93	0.97	0.94	0.96	0.89	0.62	0.95	0.91	0.37	−0.29	0.51	−0.11
S_1	✓	0.05	−0.59	0.16	−0.57	0.59	−0.07	0.28	−0.78	−0.09	−0.78	−0.47	−0.85
$\mathrm{H}(Z_3)$	✓	0.82	0.92	0.86	0.87	−0.57	−0.83	0.18	−0.20	−0.26	−0.62	0.60	0.47

TABLE II: Pearson correlation between performance metrics and linear probe accuracy when transferring models pre-trained on ImageNet to CIFAR-10 or CIFAR-100. Correlations were measured across pre-trained models of the same architecture (ResNet, EfficientNet, or DenseNet), but different sizes. Grey: no significant correlation ($p < 0.05$). Black: +ve correlation. Red: −ve.

		ResNet		DenseNet		EfficientNet		Overall	
Metric	Label-free	CIFAR-10	CIFAR-100	CIFAR-10	CIFAR-100	CIFAR-10	CIFAR-100	CIFAR-10	CIFAR-100
$\mathrm{AMI}(C_1, C_{\mathrm{GT}})$	✗	0.86	0.92	0.73	0.69	0.80	0.72	0.20	0.38
$\mathrm{AMI}(C_1, C_2)$	✓	0.83	0.96	−0.67	0.32	0.38	0.45	0.03	0.23
S_{GT}	✗	0.95	0.97	−0.15	0.10	0.24	0.83	0.16	0.21
S_1	✓	0.95	0.99	0.44	−0.06	−0.16	−0.12	−0.06	0.13
$\mathrm{H}(Z_3)$	✓	−0.19	−0.87	0.75	−0.89	−0.85	−0.54	−0.34	−0.52

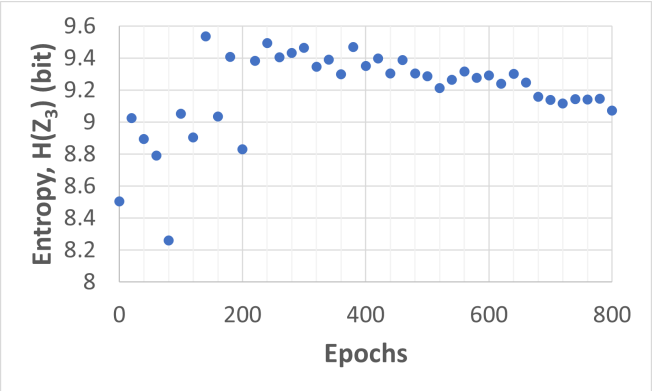


Fig. 3: Entropy progression w.r.t training for MoCo-v2 on CIFAR-100.

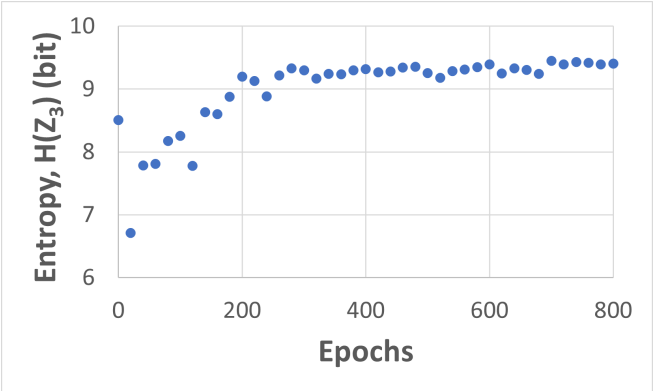


Fig. 4: Entropy progression w.r.t training for SimSiam on CIFAR-100.

We expected the entropy of the embeddings to be inversely correlated with LP accuracy, since a distribution of embedding vectors which is more tightly clustered will be associated with lower entropy. For SSL, this was only the case for SimCLR and MoCo-v2 on CIFAR-10. Investigating the distribution of CIFAR-100 Z_3 embeddings in early milestones, we discovered that SimCLR had outlier embeddings (at a large distance away from the rest of the samples) which caused a lower entropy measurement, due to its sensitivity to the size of the space spanned by the embeddings. As shown in Figure 2, these outliers occur sporadically during the early stages of training and the trend for entropy during training is otherwise consistent. In Figure 3, we observe a similar issue for MoCo-v2 training on CIFAR-100, where due to their low entropy measure, the early milestones cause the metric to be positively correlated to LP accuracy. It is important to note however, that this is not the case for SimSiam's positive entropy correlation, where the relationship is largely consistently positive throughout training, as can be seen in Figure 4.

It is not currently clear what causes these outliers to arise — whether they are outliers in the raw representational space embeddings Z_{512} as well as in the UMAP-reduced Z_3 and if so why these samples deviate from the others. One possible explanation could be that these images have unusual properties which the network has not yet internalized and hence behave like out-of-domain (OOD) data. In any case, it appears that for SimCLR and MoCo-v2, the entropy remains low and unstable during the early stages of training, before stabilizing at a higher entropy and then following a linear downward trend. This may also be the case for SimSiam, but from a

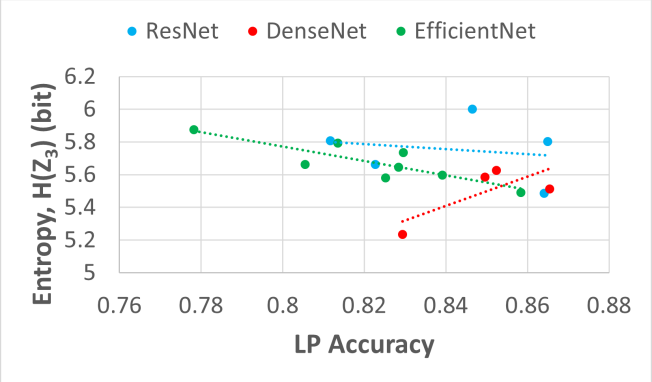


Fig. 5: Entropy vs linear probe on CIFAR-10, for networks pre-trained on ImageNet.

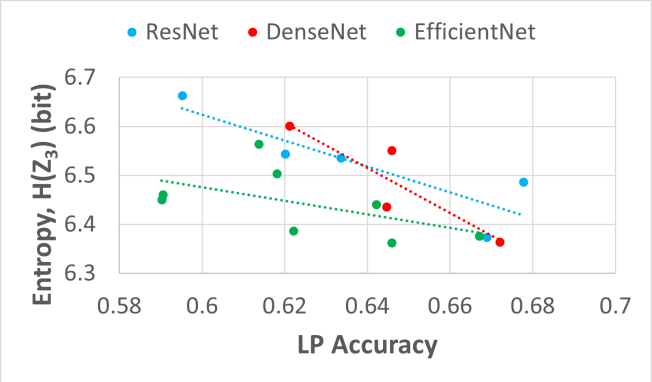


Fig. 6: Entropy vs linear probe on CIFAR-100, for networks pre-trained on ImageNet.

high initial entropy to a lower entropy prior to an increasing trend. Although it is not as visible for CIFAR-10 trials, such behaviour may still exist in the training prior to our first examined milestone. Lastly, also of note are the apparent drops in entropy around every 100 epochs seen in Figure 4. This periodicity appears to be inconsistent and we currently do not believe it to be significant. This behaviour may be caused stochastic elements introduced by the dimensionality reduction process.

Due to the limited number of model examples, all but the strongest correlating metrics did not produce significant results for our pre-trained models. Here, entropy is the only metric which produces a statistically significant result for the overall case on pre-trained torchvision models, showing a largely negative correlation with model quality across architectures. Our results for entropy are presented in Figure 5 and Figure 6. In these figures, we observe that there appears to be a degree of overlapping points between architectures along the same general negative trend, which is a promising sign for a capable cross-architecture comparison metric. We also note that the weak correlation result for CIFAR-10 appears to be caused by a DenseNet point and a ResNet point.

V. Conclusion

The label-free metrics we propose in this paper are generally indicative of the quality of the embedding space as measured with downstream classification when training the network with SimCLR or MoCo-v2. However, none of our metrics were able to consistently measure the utility of the embedding space learned with SimSiam. If the cause for why entropy reverses direction depending on methodology can be identified, entropy may be a viable means of label-free learning monitoring or potentially a means to compare different models. Further work is needed to resolve why the metrics work relatively well only with some SSL methodologies and to test them on additional methods.

As cross-architecture measures of model quality, clustering-based metrics appear to be insufficient as even with ground truth, we were unable to obtain strong results in an overall scenario encompassing ResNet, DenseNet, and EfficientNet architectures. However, there is some evidence that entropy may be architecture-independent given the significant result for CIFAR-100 in this overall case and by visual examination of entropy behaviour compared to LP accuracy. Additional architecture examples would help establish greater confidence in our results.

Acknowledgment

This work was supported by the Ocean Frontier Institute as part of the Benthic Ecosystem Mapping and Engagement project.

References

[1] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," 2020.
[2] X. Chen, S. Xie, and K. He, "An empirical study of training self-supervised vision transformers," 2021.
[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] H. Liu, J. Z. HaoChen, A. Gaidon, and T. Ma, "Self-supervised learning is more robust to dataset imbalance," *CoRR*, vol. abs/2110.05025, 2021. [Online]. Available: https://arxiv.org/abs/2110.05025
[6] D. Hendrycks, M. Mazeika, S. Kadavath, and D. Song, "Using self-supervised learning can improve model robustness and uncertainty," *CoRR*, vol. abs/1906.12340, 2019. [Online]. Available: http://arxiv.org/abs/1906.12340
[7] X. Chen, H. Fan, R. Girshick, and K. He, "Improved baselines with momentum contrastive learning," 2020.
[8] Z. Wu, Y. Xiong, S. Yu, and D. Lin, "Unsupervised feature learning via non-parametric instance-level discrimination," 2018.
[9] A. Krizhevsky, "Learning multiple layers of features from tiny images," University of Toronto, Tech. Rep., 2009.
[10] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.
[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385
[12] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," *CoRR*, vol. abs/1905.11946, 2019. [Online]. Available: http://arxiv.org/abs/1905.11946
[13] G. Huang, Z. Liu, and K. Q. Weinberger, "Densely connected convolutional networks," *CoRR*, vol. abs/1608.06993, 2016. [Online]. Available: http://arxiv.org/abs/1608.06993

[14] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NIPS 2017 Workshop on Autodiff*, 2017. [Online]. Available: https://openreview.net/forum?id=BJJsrmfCZ

[15] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *CoRR*, vol. abs/1807.03748, 2018. [Online]. Available: http://arxiv.org/abs/1807.03748

[16] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, pp. 379–423, 1948.

[17] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[18] P. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *J. Comput. Appl. Math.*, vol. 20, no. 1, pp. 53–65, 1987. [Online]. Available: http://portal.acm.org/citation.cfm?id=38772

[19] E. Min, X. Guo, Q. Liu, G. Zhang, J. Cui, and J. Long, "A survey of clustering with deep learning: From the perspective of network architecture," *IEEE Access*, vol. 6, pp. 39 501–39 514, 2018.

[20] L. N. Smith, "A disciplined approach to neural network hyper-parameters: Part 1 - learning rate, batch size, momentum, and weight decay," *CoRR*, vol. abs/1803.09820, 2018. [Online]. Available: http://arxiv.org/abs/1803.09820

[21] R. Bellman, "Dynamic programming," *Science*, vol. 153, no. 3731, pp. 34–37, 1966.

[22] L. McInnes, J. Healy, and J. Melville, "UMAP: Uniform manifold approximation and projection for dimension reduction," 2020.

[23] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014. [Online]. Available: https://arxiv.org/abs/1412.6980

[24] E. D. Cubuk, B. Zoph, D. Mané, V. Vasudevan, and Q. V. Le, "Autoaugment: Learning augmentation policies from data," *CoRR*, vol. abs/1805.09501, 2018. [Online]. Available: http://arxiv.org/abs/1805.09501

[25] T. Hua, W. Wang, Z. Xue, Y. Wang, S. Ren, and H. Zhao, "On feature decorrelation in self-supervised learning," *CoRR*, vol. abs/2105.00470, 2021. [Online]. Available: https://arxiv.org/abs/2105.00470

[26] L. Jing, P. Vincent, Y. LeCun, and Y. Tian, "Understanding dimensional collapse in contrastive self-supervised learning," *CoRR*, vol. abs/2110.09348, 2021. [Online]. Available: https://arxiv.org/abs/2110.09348