Sample Efficiency Gains in Contrastive Unsupervised Representation Learning

Blake Bordelon[*]
School of Engineering and Applied Science
Harvard University
Cambridge, MA 02138
blake_bordelon@g.harvard.edu

Cedric Flamant[†]
Department of Physics
Harvard University
Cambridge, MA, 02138
cflamant@g.harvard.edu

Salma Abdel Magid[‡]
School of Engineering and Applied Science
Harvard University
Cambridge, MA, 02138
sabdelmagid@g.harvard.edu

Abstract

Unsupervised representation learning allows the discovery of useful transformations of data from unlabeled examples. The utility of learning such representations is that simple downstream classifiers can be easily trained on the outputs of the representation function once small amounts of labeled data become available. In this work, we empirically study the sample efficiency gains of contrastive unsupervised representation learning (CURL). We find that the boost in downstream test accuracy above baseline supervised training is dependent on the number of available unlabeled and labeled data points, as well as the dimension of the latent space. Intuitively the largest gains due to CURL pretraining occur when labeled data is scarce, the latent space is low-dimensional, and the data is generated from many underlying classes. We provide some theoretical analysis of this algorithm by studying its optimal solution in the special case when the representation function is linear, as well as investigate the dimension dependence of the accuracy boost due to CURL.

1 Introduction

Learning representation functions with unlabeled data has been shown to be beneficial for downstream classification tasks [1] [2] [3]. Recent theoretical works have attempted to explain utility of unsupervised representation learning for classification by invoking the concept of latent classes and hypothesizing that similar points are sampled from the same latent class [4] [5]. Particular focus has been given to the set of contrastive representation learning algorithms, which leverage prior assumptions about which data points should have similar embedded representations (*i.e.* fragments of nearby text in a corpus or nearby frames in a video). Although these works have refined the concepts of contrastive learning and the necessity of negative sampling (penalizing similar representations of data points expected to be dissimilar), there has been little work empirically evaluating the sample efficiency of such schemes compared to baseline supervised learning algorithms. This question is

[*]blake_bordelon@g.harvard.edu
[†]cflamant@gmail.com
[‡]sabdelmagid@g.harvard.edu

Preprint. Under review.

pressing given the ubiquity of unlabeled data and the difficulty of acquiring labeled data, for example, in the case of natural images [6]. Rather than learning to predict labels that were provided by human labelers, unsupervised or self-supervised systems learn representations through assumptions about similar semantic content [5] or by making linear predictions in the latent space [3].

In this work, we focus particular attention on a version of CURL analyzed by Arora *et. al.* [5], providing empirical support for the claimed effects of CURL pretraining on downstream supervised loss, specifically the impact of increasing the budget of unlabeled examples and class collision due to spurious negative sampling. However, Arora *et. al.* also demonstrate a negative result in that, CURL does not provably choose a representation function $f \in \mathcal{F}$ that is competitive with the best function in this class on downstream classification tasks [5].

Since theoretical analysis is insufficient to guarantee competitiveness with functions from $\mathcal{F}$, we aim to assess the competitiveness of CURL with pure-supervised training where the supervised classifiers are a compositions of functions from $\mathcal{F}$ and linear transformations. Our novel contributions are obtaining empirical estimates of the comparison between the representation functions picked by CURL and the representations picked through supervised training directly. We also discover a dependence of the accuracy gain on the output dimension of the representation function, where CURL offers greater improvements when learning low dimensional representations. We find that if access to positive samples is unambiguous, the representations learned by CURL are competitive with those obtained by direct supervised learning, especially if the dimension of the latent representations are low.

In addition to this ideal setting, wherein samples from the same class can be taken unambiguously, we also explore the performance of a semi-supervised method, we call "label bootstrapping" where a classifier is first trained on labeled examples then pseudo-labels are assigned to an unlabeled dataset so that positive samples can be used for CURL. Under some circumstances, CURL can produce a modest improvement on classification accuracy, but the approach is limited since no class identity information is gained through CURL training on pseudo-labels.

2 CURL Definition and Analytic Results

In the CURL framework, we assume that the data comes from a set $\mathcal{C}$ of latent classes and that access to samples from the same class are available to the learning algorithm. The goal of the algorithm is to learn a representation function $f \in \mathcal{F}$ that produces low contrastive loss, defined as

$$\mathcal{L}_{CURL}(f) = \mathbb{E}_{c,c' \in \mathcal{C}} \mathbb{E}_{x,x^+ \sim c, x^- \sim c'} \ell(f(x)^\top f(x^-) - f(x)^\top f(x^+)) \tag{1}$$

where x, x^+ are both sampled from class c and x^- is sampled from another class c'. ℓ is a convex non-decreasing function such as the logistic or hinge-loss (with negative argument). We refer to x^+ and x^- as positive and negative samples respectively.

We provide an analytic solution to CURL for the special case where the loss is linear and the representation function is a linear transformation: $\ell(z) = z$ and $f(x) = Ax$ with a constraint on the output covariance. The solution is that A is chosen to whiten the class means in the latent space, providing equal variance of the class mean vectors along each latent dimension (see Supplementary Information).

Our analysis of the linear case exhibits the necessity of providing a scale for the CURL learning problem. Without such a constraint, the contrastive objective forces all data points to be as far away as possible. Including such a constraint on the volume in latent space, we may ask the questions: what is the best way to represent the data for linear readout, and what is the lowest representation dimension for which latent classes can be separated simply?

As a simple case, consider the following example. Suppose our latent space used tanh activations so that all datapoints are restricted to a d-dimensional hypercube $[-1, 1]^d$ in the latent representation space. We show that if $d \geq \log_2(C)$, where C is the number of classes, and a representation function f can be chosen from $\mathcal{F}$ such that each latent class cluster concentrates at its own vertex under the image of f, then a set of linear readout weights $W \in \mathbb{R}^{C \times d}$ can be chosen to give zero top-1 classification risk (Supplementary Information contains the proof). We refer to this type of representation function as vertex coding.

The simple example of vertex coding demonstrates that very low dimensional representations can suffice even as the number of latent classes grows (*e.g.* $C = 1024$ classes gives a vertex coding bound on the latent dimension of $d = 10$). If the appropriate representation function is learned, namely a mapping of all points from each class to near their respective corner of the hypercube, then downstream classification becomes trivial.

Machine learning practice seems to violate logarithmic growth in the number of latent dimensions (dimension of the penultimate layer of a NN) as a function of the number of classes. Though such low-dimensional representations that respect class identity are possible in principle, they are often more difficult to learn than high dimensional ones since the probability that a separating hyperplane exists for a random sample of N points with binary labels increases with d [7]. This result is often used to justify projecting data to high dimensional spaces with kernel machines or over-parameterized architectures. A growing body of research demonstrates that over-parameterization may actually help rather than hinder generalization as well as reduce the training error due to more powerful function approximation [8][9].

Despite the benefits of high dimensional representations and over-parameterized models, low-dimensional representations can be beneficial for compression and extracting robust and interpretable latent features [10] [11] [12]. We empirically show in this work that CURL provides larger improvements to downstream supervised accuracy when learning low dimensional representation functions, demonstrating that unsupervised pretraining may be more useful when attempting to extract low dimensional features while maintaining linear separability of latent classes.

3 Experimental Results

3.1 Synthetic Data

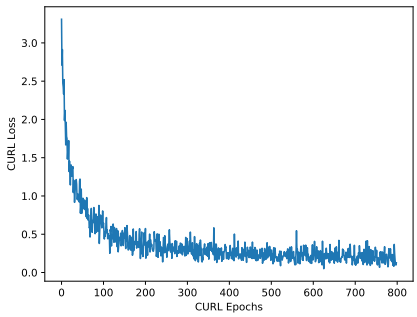
To demonstrate the coincidence of the contrastive loss decreasing during training and the downstream supervised loss decreasing, we generated small synthetic datasets from a Gaussian mixture model, allowing rapid training of downstream supervised classifiers. Arora *et. al.* show that the CURL loss acts as a surrogate for the downstream classification loss, as we verify with the following experiment [5].

To generate the data, we sampled mean vectors μ_c for each class c from a correlated Gaussian $\mu_c \sim \mathcal{N}(0, \Sigma_\mu)$ and then sampled individual data points from a Gaussian mixture

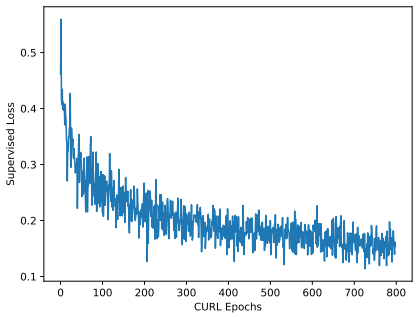
$$p(x) = \frac{1}{C} \sum_{c=1}^{C} \mathcal{N}(x; \mu_c, \sigma^2 I) \tag{2}$$

For this set of initial experiments, we assumed that for each initial sample x, positive samples x^+ (samples from the same class) and negative samples x^- (samples from one of the other classes) could be produced unambiguously. In a later section, we will discuss the limitations of unsupervised representation learning when this demand is not met.

After every epoch (250 SGD steps on the CURL objective), we trained to convergence a supervised logistic classifier on the outputs, keeping the representation function f fixed. This allows us to quantify how well the representation function performs on the downstream C-way classification task as it trains with CURL. In this experiment we used $C = 30$ latent classes with $x \in \mathbb{R}^{10}$ and a two layer neural network $f : \mathbb{R}^{10} \to \mathbb{R}^{10}$. The supervised linear-logistic classifier $g : \mathbb{R}^{10} \to \Delta^{29}$ maps the outputs of f to the standard simplex of dimension 29, producing a probability distribution over the possible latent classes. Prediction is taken as the maximum index over the output vector $\hat{y} = \text{argmax}_i g(f(x))_i$. At each epoch the supervised classifier g is trained to convergence. Then test points were sampled from the underlying Gaussian mixture model described above to estimate the out of sample test error. As expected, the test supervised loss decreases with the CURL loss, indicating that this algorithm generates representations that are useful for downstream classification. Figure 1 shows the CURL loss and the downstream supervised loss during training. Here both f and the readout logistic classifier are implemented in PyTorch.



(a) CURL loss during training.



(b) Downstream supervised loss during CURL training.

Figure 1: CURL loss as a surrogate for downstream test risk for the synthetically generated dataset. An epoch is defined as 250 steps of SGD. CURL algorithm learns representations where the latent classes are closer to being linearly separable.

3.2 Sample Efficiency Boosts

For MNIST, we let $\mathcal{F}$ be the space of 2-layer fully connected neural networks with input dimension 784 and hidden dimension 470 (60% of the input dimension). We used tanh nonlinearities to constrain the dynamic range of our latent space. The hidden layer is then mapped to an output latent dimension which varies across our different experiments. We took ℓ to be the logistic loss for both CURL and downstream supervised training.

In this section, we make the unrealistic assumption that samples from the same latent class can be obtained unambiguously but that negative samples could potentially come from the same class. We make this choice since we are primarily interested in the best case improvement in downstream classification accuracy due to CURL and how this boost depends on the amount of unlabeled M and labeled N examples at our disposal, as well as the dimension of the latent representation space. The results of this section are meant to illustrate the benefit CURL provides when similar data points can be sampled without ambiguity, but negative samples are drawn randomly. Below, we discuss how this sampling assumption is not always satisfied and the consequences that such a sampling assumption has on the utility of CURL in practice.

Figure 2 shows the difference in latent representations learned from pure supervised training with limited data $N = 100$ and learning a $d = 20$ dimensional representation from a larger budget of unlabeled examples $M = 10000$ with the CURL objective. The contrastive algorithm appears to learn representations where the latent classes have low intraclass deviation and large margin separation. This intuitive picture is reinforced by our experiments comparing downstream test risk for pure supervised models and CURL-pretrained models.

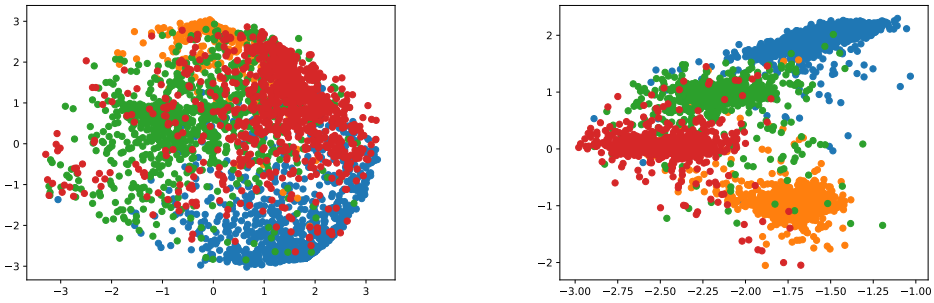
We compare training an end-to-end neural network model with A) pure supervised learning on N examples and B) CURL pretraining on M examples, learning a final linear classifier from N labeled data points. Let $\mathcal{F}$ be the set of all representation functions that map from the input space to a subset of $\mathbb{R}^d$. When performing pure-supervised training, we pick a function g_{sup} from $\mathcal{G} = \{x \to Wf(x) | f \in \mathcal{F}, W \in \mathbb{R}^{C \times d}\}$ that minimizes the empirical supervised risk (logistic loss) on N data points: $g_{sup} = \text{argmin}_{g \in \mathcal{G}} \hat{\mathcal{L}}^{(N)}_{sup}(g)$ where the superscript (N) indicates the data-set with N labeled examples. Thus, the supervised model may optimize all of the parameters of the neural network.

In contrast, the CURL procedure first picks a function $\hat{f}_{CURL} \in \mathcal{F}$ that minimizes the empirical CURL risk $\hat{f}_{CURL} = \text{argmin}_{f \in \mathcal{F}} \hat{\mathcal{L}}^{(M)}_{CURL}(f)$. Then a set of linear readout weights are chosen with fixed $\hat{f}_{CURL}$: the classifier is chosen from $\mathcal{G}_{CURL} = \{x \to W\hat{f}_{CURL}(x) | W \in \mathbb{R}^{C \times d}\}$ as the minimizer of the empirical risk on the same supervised data set: $\hat{g}_{CURL} = \text{argmin}_{g \in \mathcal{G}_{CURL}} \hat{\mathcal{L}}^{(N)}_{sup}(g)$. For this CURL procedure, the supervised training is only performed to optimize the readout weights W with representation function $\hat{f}_{CURL}$ fixed.

Figure 3 shows the M dependence of the downstream test risk for CURL-learnt representations as well as the boost in accuracy obtained through CURL-pretraining. For these experiments, $d = 100$ and all 10 classes are used. The results shown in the figure are averaged over 5 random samples of supervised data to reduce the variance. The accuracy boost B is defined as the difference in the test risk between the classifier picked through the pure-supervised procedure and that of the CURL procedure described above:

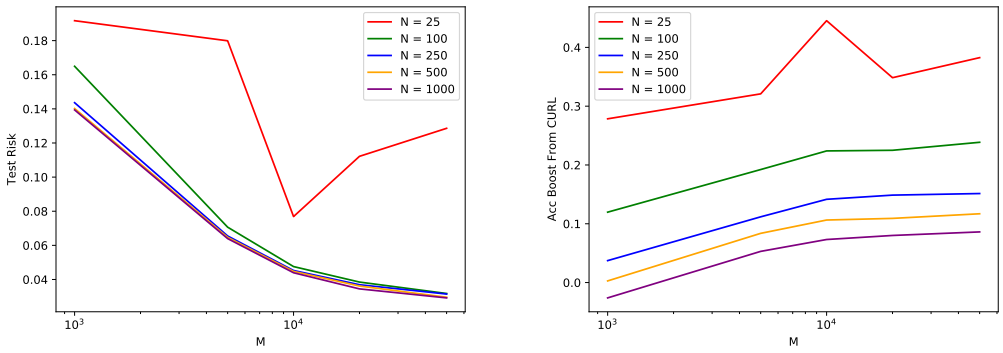
$$B = \mathcal{L}_{sup}(\hat{g}_{sup}) - \mathcal{L}_{sup}(\hat{g}_{CURL}) \tag{3}$$

The boost is our definition of the utility of CURL training. For example Figure 1 (right) shows that for $N = 100$ labeled data points and $M = 10,000$ unlabeled data points, the test risk for the CURL pretrained model is around 6%, a 20% improvement on the pure-supervised model. Intuitively, our experiments show that the boost is largest for small N and large M with diminishing returns as M increases.



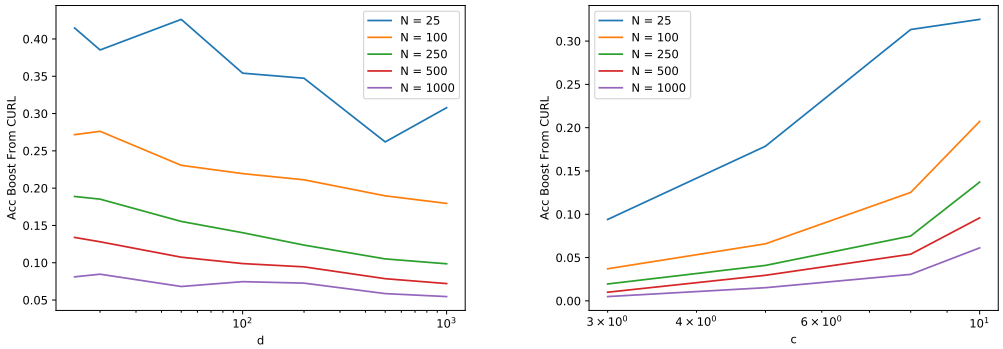
(a) Pure supervised training $N = 100$ (b) CURL pretraining $M = 10000$

Figure 2: First 4 classes of MNIST test data under the image of the representation function learned with a) pure supervised training with $N = 100$ labeled data points and b) CURL pretraining with $M = 10,000$ unlabeled data points. The representation is projected from $d = 20$ dimensions onto the first two principal components for visualization. The downstream test risk is 24.7% for a) pure supervised model and 6.2% for b), the CURL pretrained model.



(a) Downstream Test Risk for CURL (b) Accuracy Boost from CURL

Figure 3: Accuracy Boost from CURL on MNIST, defined as the difference between pretraining with CURL and M unlabeled data points and only using supervised data for readout weights and the test risk from training the entire model with N supervised data points. Access to positive samples was unambiguous but class collision due to spurious negative sampling was allowed. For small N, the boost in accuracy due to pretraining with CURL is large but becomes smaller as N increases, though it always provides some advantage. Larger M gives a slight increase in the boost, as expected.



(a) Dimension Dependence of Boost

(b) Class Number Dependence of Boost

Figure 4: Dimension and class dependence of CURL accuracy boost. M was fixed at $10,000$ as d and c were varied. CURL offers greater utility when learning low dimensional representations and when there are many underlying latent classes.

3.3 Dimension and class number dependence of Accuracy Boost

In addition to the (M, N) sample complexity dependence on the CURL accuracy boost, we also find a dependence on the dimension of the representation d and the number of latent classes C. For these experiments we fixed $M = 10000$. The largest accuracy boosts due to CURL pretraining occur when the representation function maps to a low dimensional space, demonstrating the utility of CURL in this regime. Learning a linear classifier from a high dimensional space is not as difficult due to Cover's theorem and the fact that random feature models are low-rank approximations of reproducing kernel Hilbert spaces [8] [13]. Generating low dimensional representations with low supervised test accuracy requires greater effort and it is in this regime that CURL provides the greatest improvements.

When there are very few classes, CURL suffers from the class collision problem, since negative samples are occasionally drawn from the same class as the positive sample. Suppose that we can correctly sample similar pairs x, x^+ but the negative samples x^- are drawn at uniformly random from any class $c \in \mathcal{C}$. In this case, there is a non-zero probability of $\tau = \frac{1}{C}$ that the negative sample x^- is drawn from the same class as x. Curiously, CURL will train more effectively when there are many underlying latent classes since this reduces the probability of drawing a negative sample from the same class from which x was drawn. The effect of this phenomenon is dramatic when there are only two classes since, for each negative sample, it is equally likely that a sample will be drawn from c as from the other class.

3.4 Getting Something from Nothing: Bootstrapping CURL

We have shown in previous sections that having knowledge of same class examples within the unlabeled data (x^+ for a given x) can provide substantial gains in test accuracy upon employing CURL on the latent space representation. However, this is not very realistic in most practical situations, as this knowledge could potentially be better used to perfectly cluster same-class examples, and a single correct class label within the cluster would label the entire cluster. A more realistic approach would exploit better-than-chance probabilities of correctly sampling from similar and dissimilar classes. A naive way to attain this is to simply use the full network trained on the available labeled data and use it to approximately classify the unlabeled data, a process referred to as pseudo-labeling [14]. Typically, such an approach introduces confirmation bias, where incorrect predictions are strengthened leading to effects similar to overfitting [15]. However when using CURL on the pseudo-labeled data, we find that this "bootstrapping" procedure can result in improved performance.

In Figure 5 we show a comparison of bootstrapped CURL to regular supervised training when only a fraction of the data is labeled. The experiment was conducted on the Google Street View House Numbers dataset [16]. For each run we randomly selected 10% of the total data (7325 from 73257 images), then treated a fraction of these images as labeled data and the remainder as unlabeled. A

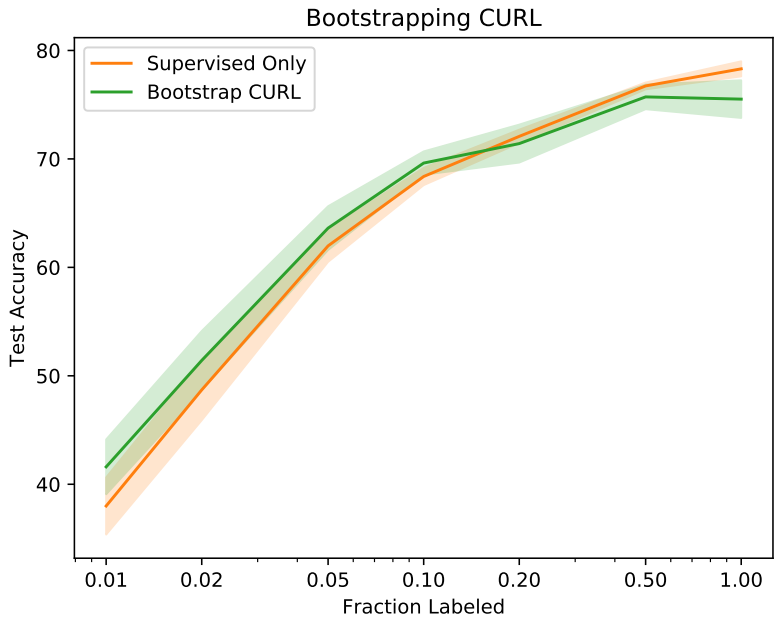


Figure 5: Using a randomly-selected 10% (7325 images) subset of the SVHN dataset for each of 10 runs, we first trained on a fractional budget (x-axis) of labeled data to convergence, keeping the best test score for the supervised network (orange line). For bootstrapped CURL, the entire subset (including the fraction that we assume to be labeled) was given approximate labels using the supervised network, and the bulk of the network was then trained with a contrastive softplus loss. The head of the network was finally trained on the original fraction of correctly-labeled data while the bulk weights are frozen, keeping the best test accuracy (green line). We show $\pm 1\sigma$ of the results from the 10 runs.

simple network consisting of two convolution layers, two maxpool layers, and a linear layer outputting to an 80-dimensional latent space through a tanh activation was used for the bulk, and two linear layers were used for the head. The bulk of the network $f : X \to [-1, 1]^{80}$ casts the input images to latent space, while the head of the network classifies from latent space to the simplex of dimension 9, $g : [-1, 1]^{80} \to \Delta^9$, giving normalized probabilities over the 10 classes. The full network was trained on the available labeled data and then used to approximately classify the unlabeled data. The bulk of the network was then trained on the approximately-labeled data combined with the labeled data using the contrastive loss

$$\mathcal{L}_{CURL}(f) = \text{softplus}\left(\frac{f(x)^\top f(x^-) - f(x)^\top f(x^+)}{80}\right), \tag{4}$$

where the normalization by the latent space dimension forces the argument to be in $[-2, 2]$, helping with numerical stability and convergence. Finally, the head of the network was trained on the labeled data with the bulk weights frozen. All training was performed with color-jitter and random affine data augmentation of the images.

We find that when unlabeled data exceeds 10 times the amount of labeled data, we can obtain improved performance, despite the fact that the unlabeled data was itself classified with a statistically lower accuracy. This could be due to the CURL loss inducing a simpler latent space representation than the naturally-occurring representations in supervised training, as seen in Figure 2. Such clusters might be better-characterized by a few labeled points when the network forms decision boundaries in the head classifier.

As a proof-of-concept, we tested bootstrapped CURL on the STL-10 dataset for semi-supervised learning [17]. The 10-class dataset contains 5,000 labeled images in total, and 100,000 unlabeled images from a similar but broader distribution of images. We used a modified ResNet-18 [18] where we added two additional dense layers to the end to form the head of the network, and had the bulk output to a latent space of 128 dimensions through a hyperbolic tangent activation. During

the supervised training of the full network on all 5,000 labeled images, employing color-jitter and random affine data augmentation, the test accuracy plateaued to 62.01%. When applying bootstrapped CURL, the test accuracy increased to 63.25%. This modest gain alone might not warrant the use of bootstrapped CURL for semi-supervised learning, but the latent space representation it induces could be useful in conjunction with label-propagation methods [19].

4 Conclusion

CURL provides the greatest accuracy boosts in low dimensions, tasks involving many latent classes, and in the presence of large amounts of unlabeled data and scarce labeled data. Though learning low dimensional representations whose geometry respects class membership requires greater computational effort, these low dimensional representations can be useful for compression, denoising, and interpretability while still giving faithful linear readout.

If positive samples are unavailable, CURL label bootstrapping can provide only modest boosts in test accuracy. Future work could examine the downstream accuracy boosts due to other schemes that do not require positive samples, such as soft-labeling and predictive coding [3], or combining the nice latent space representations induced by CURL with other semi-supervised methods like iterative pseudo-labeling [14] or label propagation [19].

Acknowledgments

We thank Jean-Baptiste Tristan for the helpful comments.

4.1 Code Availability

The code for these experiments can be found at https://github.com/blakebordelon/CURL-Efficiency

References

[1] Lajanugen Logeswaran and Honglak Lee. An efficient framework for learning sentence representations, 2018.

[2] Xiaolong Wang and Abhinav Gupta. Unsupervised learning of visual representations using videos, 2015.

[3] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding, 2018.

[4] Sanjeev Arora and Andrej Risteski. Provable benefits of representation learning, 2017.

[5] Sanjeev Arora, Hrishikesh Khandeparkar, Mikhail Khodak, Orestis Plevrakis, and Nikunj Saunshi. A theoretical analysis of contrastive unsupervised representation learning, 2019.

[6] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, June 2009.

[7] Adam Kowalczyk. Counting function theorem for multi-layer networks. In *Proceedings of the 6th International Conference on Neural Information Processing Systems*, NIPS'93, pages 375–382, San Francisco, CA, USA, 1993. Morgan Kaufmann Publishers Inc.

[8] Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. Reconciling modern machine learning practice and the bias-variance trade-off, 2018.

[9] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization, 2016.

[10] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 1096–1103, New York, NY, USA, 2008. ACM.

[11] Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 833–840, USA, 2011. Omnipress.

[12] W. James Murdoch, Chandan Singh, Karl Kumbier, Reza Abbasi-Asl, and Bin Yu. Definitions, methods, and applications in interpretable machine learning. *Proceedings of the National Academy of Sciences*, 116(44):22071–22080, Oct 2019.

[13] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1177–1184. Curran Associates, Inc., 2008.

[14] Dong-Hyun Lee. Pseudo-label : The simple and efficient semi-supervised learning method for deep neural networks. *ICML 2013 Workshop : Challenges in Representation Learning (WREPL)*, 07 2013.

[15] Eric Arazo, Diego Ortego, Paul Albert, Noel E. O'Connor, and Kevin McGuinness. Pseudo-labeling and confirmation bias in deep semi-supervised learning, 2019.

[16] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011.

[17] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In Geoffrey Gordon, David Dunson, and Miroslav Dudík, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 215–223, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2015.

[19] Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondrej Chum. Label propagation for deep semi-supervised learning, 2019.

[20] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

5 Supplementary Information

5.1 CURL as a Whitening Filter: A Linear Model

Proposition 1: *Let $f(x) = Ax$ and $\ell(z) = z$. The $A \in \mathbb{R}^{d \times N}$ that minimizes $\mathcal{L}_{CURL}$ with a constraint on output covariance is a whitening filter for the latent class means: if the latent class means in the input space have covariance $\Sigma_\mu = \mathbb{E}_c(\mu_c - \overline{\mu})(\mu_c - \overline{\mu})^\top = U\Lambda U^\top$ for some orthogonal matrix $U \in \mathbb{R}^{N \times N}$ and diagonal matrix $\Lambda \in \mathbb{R}^{N \times N}$ with ordered diagonal elements $\lambda_1 > \lambda_2 > ... > \lambda_N$, then the CURL optimal A satisfies $A = \sqrt{\nu} \sum_{i=1}^{d} \lambda_i^{-1/2} v_i u_i^\top$ for arbitrary orthogonal vectors $v_i \in \mathbb{R}^d$ and some scalar ν.*

Proof: Assume that data are drawn from a set $\mathcal{C}$ of latent classes. The contrastive unsupervised representation learning (CURL) objective is the following

$$\mathcal{L} = \mathbb{E}_{c,c' \sim \mathcal{C}} \mathbb{E}_{x,x^+ \sim c} \mathbb{E}_{x^- \sim c'} \ell(f(x)^\top f(x^-) - f(x)^\top f(x^+)) \tag{5}$$

where c' is assumed distinct from c [5] [1]. Let's attempt to derive an analytic solution for a simple case. One in which $\ell(z) = z$ and $f(x) = Ax$.

The linear model considers minimizing the right hand side and the CURL objective is represented by the left hand side.

Let the input variables $x \in \mathbb{R}^N$ and the matrix $A \in \mathbb{R}^{d \times N}$. With this choice, the objective above can be reduced to the following

$$\mathcal{L} = \mathbb{E}_{c,c' \sim \mathcal{C}} \mathbb{E}_{x,x^+ \sim c} \mathbb{E}_{x^- \sim c'} x^\top A^\top A(x^- - x^+) = \mathbb{E}_{c,c' \sim \mathcal{C}} \mu_c^\top A^\top A(\mu_{c'} - \mu_c) = \text{Tr}(A^\top A \mathbb{E}_{c,c' \neq c}(\mu_{c'} - \mu_c)\mu_c^\top) \tag{6}$$

The last expectation can be broken up

$$\mathbb{E}_{c' \neq c} \mu_c' = \frac{1}{|\mathcal{C}| - 1} \sum_{c' \neq c} \mu_{c'} = \frac{1}{|\mathcal{C}| - 1}(\overline{\mu}|\mathcal{C}| - \mu_c) = \frac{|\mathcal{C}|}{|\mathcal{C}| - 1}\overline{\mu} - \frac{1}{|\mathcal{C}| - 1}\mu_c \tag{7}$$

Now note

$$\mathbb{E}_{c,c'}(\mu_{c'} - \mu_c)\mu_c^\top = \frac{|\mathcal{C}|}{|\mathcal{C}| - 1}\overline{\mu}\,\overline{\mu}^\top - \frac{|\mathcal{C}|}{|\mathcal{C}| - 1}\mathbb{E}_c \mu_c \mu_c^\top = -\frac{|\mathcal{C}|}{|\mathcal{C}| - 1}\left[\mathbb{E}_c \mu_c \mu_c^\top - \overline{\mu}\,\overline{\mu}^\top\right] = -\frac{|\mathcal{C}|}{|\mathcal{C}| - 1}\Sigma_\mu \tag{8}$$

where Σ_μ is the covariance matrix for the latent class means.

We are therefore interested in the optimization problem

$$\min_A \; -\text{Tr}(A^\top A \Sigma_\mu). \tag{9}$$

Without any constraints, the solution to this problem is to make $A \to \infty$. Therefore to arrive at a sensible solution, we must constrain the scale of A. One choice of a constraint would be to limit size of the output latent center covariance matrix *i.e.* the Frobenius norm of $A\Sigma_\mu A^\top$. Using a Lagrange multiplier $1/\nu$, we can introduce the following Lagrangian for this constrained optimization problem

$$\mathcal{L} = \text{Tr}(A\Sigma A^\top A \Sigma A^\top) - 2\nu \text{Tr}(A\Sigma_\mu A^\top) + \nu^2 \text{Tr}(I^\top I) = ||A\Sigma_\mu A^\top - \nu I||_F^2 \tag{10}$$

which is recognized as a multi-dimensional scaling or matrix completion problem. Let the decomposition of the covariance matrix be $\Sigma_\mu = U \Lambda U^\top$. Make the ansatz that the singular value decomposition of A is given by

$$A = \sqrt{\nu} V \tilde{\Sigma} U^\top \tag{11}$$

for some arbitrary orthogonal matrix $V \in \mathbb{R}^{d \times d}$ and a diagonal matrix $\tilde{\Sigma} \in \mathbb{R}^{d \times N}$ with $\tilde{\Sigma}_{ii} = \sigma_i$. Plugging this ansatz into the objective and dropping constant terms gives

$$\mathcal{L} = \nu^2 \text{Tr}(\tilde{\Sigma}^\top \tilde{\Sigma} \Lambda \tilde{\Sigma}^\top \tilde{\Sigma} \Lambda) - 2\nu^2 \text{Tr}(\tilde{\Sigma}^\top \tilde{\Sigma} \Lambda) = \nu^2 \sum_{i=1}^N \left[\sigma_i^4 \lambda_i^2 - 2\sigma_i^2 \lambda_i\right] \tag{12}$$

Note that if $d \geq N$ then the optimal solution is to choose $\sigma_i = \lambda_i^{-1/2}$. However, if $d < N$, then only d values of σ_i are non-zero and the loss is optimized by choosing only the d principal components with the largest eigenvalues λ_i. Thus, if the eigenvalues of Σ_μ are ordered $\lambda_1 > \lambda_2 > ... > \lambda_N$ then $\sigma_i = \lambda_i^{-1/2}$ for $1 \leq i \leq d$. We may therefore write

$$A = \sqrt{\nu} \sum_{i=1}^d \lambda_i^{-1/2} v_i u_i^\top \tag{13}$$

for arbitrary orthogonal vectors $v_i \in \mathbb{R}^d$.

This optimal weight matrix can therefore be interpreted as a whitening filter for the mean vectors of the latent classes.

Though we are generally interested in how nonlinear models perform when trained on the CURL objective, this exercise provides the intuition that learned representations will spread out the latent clusters into an isotropic ball, making the downstream task of finding separating hyperplanes easier.

5.2 Vertex Coding Construction

Suppose each class concentrates about one of the 2^d vertices so that all of the points lie in a hypercube of length $\epsilon < \frac{2}{d+1}$ around the vertex. For example, all of the points from the class assigned to the vertex $v = (+1, +1, ... + 1)$ lie in $[1-\epsilon, 1]^d$. In general, if e_i is the i-th standard basis vector and the k-th vertex is written as $v_i = \sum_{i=1}^{d} a_i^{(k)} e_i$ for binary variables $a_i^{(k)} \in \{\pm 1\}$, then the points for class k are confined to $x^{(k)} \in S_k = [a_1^{(k)}(1-\epsilon), a_1^{(k)}] \times [a_2^{(k)}(1-\epsilon), a_2^{(k)}] \times ... \times [a_d^{(k)}(1-\epsilon), a_d^{(k)}]$.

Construct a weight matrix W where the rows of the matrix are given by a scaled version of the vertex vectors $W_{ki} = \frac{1}{d} a_i^{(k)}$.

Proposition 2: *The weights W defined above achieve zero top-1 classification risk.*

Proof: Let $x^{(k)}$ be a point from class k. The prediction score for the correct class is

$$s_k = W_k^\top x^{(k)} \geq \frac{1}{d} \sum_{i=1}^{d} (1-\epsilon) = 1 - \epsilon > 1 - \frac{2}{d+1} \tag{14}$$

For a distinct class $k' \neq k$, the weight vector must differ from the true vertex vector in at least one dimension. Therefore, we can obtain an upper bound on the prediction score.

$$s_{k'} = W_{k'}^\top x^{(k)} \leq \frac{1}{d}\Big[d - 1 - (1-\epsilon)\Big] = 1 + \frac{\epsilon - 2}{d} < 1 - \frac{2}{d+1} \tag{15}$$

Thus $s_k > s_{k'}$: the prediction score for the correct class k exceeds the prediction score for any other class k'.

This toy example of vertex coding is meant to demonstrate the ideal logarithmic dependence of the latent dimension on the number of latent classes that one wishes to discriminate between.

5.3 Datasets

In our studies we used the following datasets: Street View House Numbers (SVHN), STL-10, and the MNIST database of handwritten digits.

SVHN is a real-world image dataset collected from house numbers in Google Street View images. It contains 10 classes, 1 for each digit. There are approximately 80,000 images. Each image is 32-by-32 centered around a single character. [16]

STL-10 is inspired by the CIFAR-10 dataset with a few modifications. Each image, which was acquired from labeled examples on ImageNet, is 96-by-96 pixels and is drawn from 10 mutually exclusive classes. STL-10 differs from CIFAR-10 in two main ways: (1) STL-10 is higher resolution and (2) each class has fewer labeled training examples (500) than in CIFAR-10, but 100,000 unlabeled examples from a broader subject distribution is provided for semi-supervised training. [17]

The MNIST dataset contains 60,000 images of handwritten digits. The digits have been size-normalized and centered in a fixed-size image. In the past this dataset has been used as a benchmarking dataset for many pattern recognition methods [20]. For preprocessing, we scaled each input dimension to $[0, 1]$ by dividing the pixel intensity by 255.