# Theoretical Insights into Contrastive Unsupervised Representation Learning

Student: Hrishikesh Khandeparkar
Advisor: Prof. Sanjeev Arora

April 2019

**Abstract**

Unsupervised Representation Learning has had a tremendous impact in several domains like computer vision and natural language processing (NLP). *Contrastive unsupervised* methods to learn *representations* of data points have led to a simplification of downstream classification tasks which use these representations. However, a theoretical understanding of when these methods will succeed is lacking. In this report, we use the framework presented in [Arora et al., 2019] to guide the construction of better training objectives which we further test with experiments. Furthermore, we highlight the limitations of unsupervised learning in the form of counter examples showing when unsupervised learning is doomed to fail. We verify our results by conducting experiments on commonly used function classes for classification in the domains of computer vision.

## 1 Introduction

Machine Learning as a field of research has had a tremendous impact on several applications of practical interest lately. Supervised learning – learning to classify using *labeled* samples – has become a staple task in domains like computer vision and natural language processing. While supervised learning is well studied empirically [Caruana and Niculescu-Mizil, 2006] as well as theoretically [Valiant, 1984], its counterpart, unsupervised learning, is much less understood. Unsupervised representation learning in particular, which

has enjoyed empirical success recently [Mikolov et al., 2013, Wang and Gupta, 2015], lacks the theoretical grounding that supervised learning enjoys.

In this report, we consider a particular paradigm of unsupervised learning which we refer to as *contrastive unsupervised representation learning* (CURL). Contrastive methods are in essence similar to word2vec Mikolov et al. [2013], a method used to learn low dimensional vector representations for words. These methods share a common feature – they assume access to unlabelled pairs of points $x, x^+$ that are drawn from a distribution $\mathcal{D}_{sim}(x, x^+)$ of *semantically similar* pairs of points. Along with this, these methods also use access to a *negative sample* $x^-$ drawn from a distribution $\mathcal{D}_{neg}(x^-)$. It is presumed that $x^-$ is dissimilar from $x, x^+$. Given this, contrastive methods optimize the following loss function

$$L(f) = \mathop{\mathbb{E}}_{\substack{x,x^+\sim D_{sim} \\ x^-\sim D_{neg}}} \left[ -\log \left( \frac{e^{f(x)^T f(x^+)}}{e^{f(x)^T f(x^+)} + e^{f(x)^T f(x^-)}} \right) \right]$$

Concretely, the problem resolves to finding a function $f^* = \arg\min_{f\in\mathcal{F}} L(f)$ where $\mathcal{F}$ is a class of *representation functions*. A priori, it is not clear why optimizing this loss function should lead to good representations for downstream tasks. In fact, it is not even clear whether this method should always work or if there are conditions on distributions that can lead it to fail. Consequently, despite being similar in flavor to several other paradigms like multitask learning and similarity learning, [Bellet et al., 2012, Maurer et al., 2016] contrastive learning lacks a theoretical understanding.

In this report, we use the framework of [Arora et al., 2019] to resolve two questions. First, we highlight instances where contrastive unsupervised learning can, in fact, fail to recover a 'good' representation function. In particular we show that increasing the number of negative samples can lead to the algorithm picking suboptimal representations – this is contrary to empirical evidence that increasing negative samples indeed should help. Understanding these instances leads us to insights into what possible conditions might hold for real world distributions for $\mathcal{D}_{sim}, \mathcal{D}_{neg}$. Second, we use a simple insight to provide an objective function that is a tighter guarantee of performance (in a formal sense) when one has access to *blocks* of similar data. We support both our findings with experiments in the domain of computer vision using widely used representation function classes $\mathcal{F}$. We hope that such insights can guide future theoretical work to make the correct assumptions about

real-world distributions as well guide the design of principled objectives for unsupervised learning.

The report is organized as follows. In Section 2 we present the theoretical framework of Arora et al. [2019]. In Section 3 we present the generic contrastive unsupervised representation learning algorithm. In Section 4 we describe insights that one can derive from the framework and highlight our main results. Section 5 presents our experimental work to verify the results on real world function classes of interest. Finally, we conclude in Section 6 with summary of our work and point towards further research directions.

# 2 A Theoretical Framework

We now proceed to describe the theoretical framework presented in Arora et al. [2019]. Let $\mathcal{X}$ be the set of all input data points. Let $\mathcal{F}$ be a class of representation functions $f$ such that every $f : \mathcal{X} \to \mathbb{R}^d$ and $||f(\cdot)|| \leq R$ for some $R > 0$. Recall from the introduction that contrastive unsupervised representation learning assumes access to two distributions:

i) $D_{sim}(x, x^+)$, a distribution over pairs of *semantically similar* points

ii) $D_{neg}(x^-)$ a distribution over negative samples that are *random*.

### Latent Classes and Semantic Similarity

To formalize the notion of semantic similarity Arora et al. [2019] introduce the concept of *latent classes*. This notion will be used to define the structure of $\mathcal{D}_{sim}$ as well as the supervised tasks of interest.

**Definition 2.1** (Latent Classes)**.** *Let $\mathcal{C}$ denote the set of all latent classes. Associated with each class $c \in \mathcal{C}$ is a distribution $\mathcal{D}_c$ over $\mathcal{X}$. There exists a distribution $\rho$ over $\mathcal{C}$ that characterizes how likely a class is to appear in the unlabelled data.*

$\mathcal{D}_c(x)$ roughly captures how related $x$ is to class $c$. For example if $\mathcal{X}$ is the set of plausible natural images, $\mathcal{D}_{dog}(x)$ would assign high probability to all images $x$ that are pictures of dogs, and low probability to other images.

A distribution $\rho$ is assumed over classes in order to define a notion of *unlabelled data*. Thus, $\rho$ characterizes how likely samples from these classes are to appear in unlabelled data. Now we can formalize the structure of $\mathcal{D}_{sim}, \mathcal{D}_{neg}$.

**Definition 2.2** (Semantic Similarity)**.** *Semantically similar points are points drawn i.i.d from a random class $c \sim \rho(c)$. The distribution of negative samples is the marginal of the distribution over semantically similar pairs. Formally:*

$$\mathcal{D}_{sim}(x, x^+) = \mathop{\mathbb{E}}_{x\sim\rho}[\mathcal{D}_c(x)D_c(x^+)] \tag{1}$$

$$\mathcal{D}_{neg}(x^-) = \mathop{\mathbb{E}}_{c\sim\rho}[D_c(x^-)] \tag{2}$$

This definition of $\mathcal{D}_{sim}$ is plausible because the class distributions are allowed to overlap arbitrarily. Assuming the structure of $\mathcal{D}_{sim}$ lets us formalize a notion of classification tasks of interest.

### Supervised Tasks

To define a notion of *downstream classification tasks* we restrict ourselves to binary classification tasks. Thus, a 2-way task $\mathcal{T}$ consists of a pair of classes $\{c_1, c_2\} \subseteq \mathcal{C}$. The labelled dataset is simply i.i.d. draws from each of these classes.

**Definition 2.3** (Supervised Task)**.** *A supervised tasks consists of two classes $\{c_1, c_2\}$ with $c_1 \neq c_2$ and $2m$ labelled samples. $m$ samples are i.i.d draws from $\mathcal{D}_{c_1}$ and labelled as $c_1$ and similarly for $c_2$.*

The key idea in this definition is that the $\mathcal{D}_c$ which defines how unlabelled samples appear from a class is the same as that of the distribution of labelled points in classification tasks that involve $\mathcal{D}_c$. This lets us *connect* downstream tasks to the distribution of unlabelled data.

### Evaluation Metric for Representations

In order to define a notion of *performance* on downstream tasks of interest, we now proceed to define how representations are used in downstream tasks. Let the task be $\mathcal{T} = \{c_1, c_2\}$. Then, a classifier for $\mathcal{T}$ is a function $g : \mathcal{X} : \mathbb{R}^2$ whose output coordinates are indexed by the classes $c_1, c_2$ in $\mathcal{T}$.

The supervised loss of a classifier on a labelled sample $(x, c_1) \in (\mathcal{X}, \mathcal{T})$ (and analogously for $c_2$) is given by $\ell(\{g(x)_{c_1} - g(x)_{c_2}\})$ where $\ell(z) = \log_2(1 + exp(-z))$ is either the logistic loss function or $\ell(z) = \max\{0, 1 - z\}$ is the hinge loss function. Thus, the supervised loss of the classifier function $g$ is defined as

**Definition 2.4** (Supervised Loss of Classifier)**.** *The supervised loss of a classifier $g$ on task $\mathcal{T}$ is given by $L_{sup}(\mathcal{T}, g)$ where*

$$L_{sup}(\mathcal{T}, g) = \mathop{\mathbb{E}}_{c\sim U(\{c_1,c_2\})} \mathop{\mathbb{E}}_{x\sim\mathcal{D}_c} [\ell(g(x)_c - g(x)_{c'})]$$

*Here $U(\cdot)$ is the uniform distribution over the set and $c' = \mathcal{T}/c$.*

We note that the loss function $\ell$ will be used in its more general form when used in unsupervised learning with $k$ negative samples. In particular, let $\ell : \mathbb{R}^k \to \mathbb{R}$ be given as a function on a $k$ dimensional vector as $\ell(\mathbf{z}) = \ell(\{\mathbf{z_i}\}_{i=1}^k)$ where $\ell(\mathbf{z}) = \log_2(1 + \sum_i \exp(-\mathbf{z_i}))$ or $\ell(\mathbf{z}) = \max\{0, 1 + \max_i -\mathbf{z_i}\}$.

Now, in order to use a representation function $f : \mathcal{X} \to \mathbb{R}^d$ to create a classification function we restrict ourselves to the setting of *linear classification*. Thus, a matrix $W \in \mathbb{R}^{2\times d}$ is trained and $g(x) = Wf(x)$ is used to evaluate the loss. The problem of finding the best $W$ given a fixed $f$ is just the vanilla linear classification task, so we define the notion of performance of the representation function $f$ on a task as the performance of the *best* linear classifier.

**Definition 2.5** (Supervised Performance)**.** *The performance of a representation function $f : \mathcal{X} \to \mathbb{R}^d$ on a classification task $\mathcal{T}$ is given by*

$$L_{sup}(\mathcal{T}, f) = \inf_{W\in\mathbb{R}^{2\times d}} L_{sup}(\mathcal{T}, Wf)$$

Finally, the key idea in considering an evaluation metric is to consider the *average* performance of the representation function over all binary tasks. The distribution $\rho$ lends itself naturally to this definition

**Definition 2.6** (Average Supervised Performance)**.** *The average supervised performance of a representation function $f$ is defined as*

$$L_{sup}(f) = \mathop{\mathbb{E}}_{c_1,c_2\sim\rho^2} [L_{sup}(\{c_1, c_2\}, f) | c_1 \neq c_2]$$

The key idea in Arora et al. [2019] involves using contrastive unsupervised loss $L_{un}(f)$ (defined in Section 3) to bound $L_{sup}(f)$ – the performance of the representation function learned by the unsupervised algorithm on the average binary task.

*Remark.* We note that the performance on the average binary task is hard to relate to the performance on the all-way task in cases where $C$ is a finite set.

# 3 Unsupervised Representation Learning

We now proceed to describe the contrastive unsupervised representation learning algorithm. In order to learn a *good* representation function using unlabelled data we define the unsupervised loss function.

**Definition 3.1** (Unsupervised Loss)**.** *The unsupervised loss $L_{un}(f)$ with $k$ negative samples of a representation function $f$ is given by*

$$L_{un}(f) = \mathop{\mathbb{E}}_{\substack{(x,x^+)\sim\mathcal{D}_{sim} \\ x_1^-,..x_k^-\sim\mathcal{D}_{neg}}} \left[\ell\left(\{f(x^T)\left(f(x^+) - f(x_i^-)\right)\}_{i=1}^k\right)\right]$$

*and its emprical loss $\hat{L}_{un}(f)$ is given by*

$$\hat{L}_{un}(f) = \frac{1}{M}\sum_{i=1}^{M} \ell\left(\{f(x_i^T)\left(f(x_i^+) - f(x_{ij}^-)\right)\}_{i=1}^k\right)$$

*where the $M$ samples are tuples of $(x_i, x_i^+, x_{i1}^-, ..x_{ik}^-)$ similar pairs and $k$ negative samples.*

Finally, the unsupervised algorithm to learn a representation function is simply to find the best $\hat{f} \in \arg\min_{f\in\mathcal{F}}(\hat{L}_{un}(f))$ that minimizes the empirical unsupervised loss. For the sake of this paper, we will ignore issues of generalization of the empirical unsupervised loss. For a more thorough treatment of the generalization error of the unsupervised loss, see Arora et al. [2019].

# 4 Algorithmic Insights

In the following section, we present our main results. First we explain how under the minimal assumptions in [Arora et al., 2019], excessive negative sampling can be detrimental. Then, we present a slightly modified loss function for learning when one has acess to blocks of semantically similar data. We support these insights with experiments in the Section 5.

## 4.1 The Negative Effects of Excessive Negative Sampling

Recall that the unsupervised loss function (3.1) with $k$ negative samples is given by

$$L_{un}(f) = \mathbb{E}\left[\ell(\{f(x)^T(f(x^+) - f(x_i^-))\}_{i=1}^k)\right]$$

where $x, x^+ \sim D_{sim}(x, x^+)$ and $x_i^-$ are i.i.d. draws from $D_{neg}(x^-)$. Empirically, it is widely believed that increasing the number of negative samples always helps in learning better objective functions. In fact, in Noise Contrastive Estimation (NCE) [Gutmann and Hyvärinen, 2010] which is considered a theoretical justification, increasing the number of negative samples helps provably by improving the asymptotic variance of the learned parameters. Contrary to this, we show that this phenomenon does not hold for contrastive learning. In fact, increasing the number of negative samples can hurt. The following Lemma formalizes this.

**Lemma 4.1.** *There exists a function class $\mathcal{F}$, and distributions $\mathcal{D}_c$ for $c \in \mathcal{C}$ such that $\mathcal{F}$ contains an optimal representation $f^*$ for classification tasks defined using $\mathcal{D}_c$ yet $f^* \neq \arg\min_{f \in \mathcal{F}} L_{un}(f)$ when the number of negative samples $k = \Omega(|\mathcal{C}|)$. Consequently, unsupervised learning can find a suboptimal representation function.*

*Proof.* Let $\mathcal{C} = \{c_i\}_{i=1}^n$ and let $D_{c_i}(x)$ be uniform over the set $\{x_i^1, x_i^2\}$. Namely, each class distribution $D_c$ is effectively a distribution over a pair of points. Let $\mathcal{F} = \{f_0, f_1\}$ where $f_0(x) = \mathbf{0}$ for all $x \in \mathcal{X}$ and $f_1(x_i^1) = 3/2r\mathbf{e_i}$, $f_1(x_i^2) = 1/2r\mathbf{e_i}$ where $\mathbf{e_i}$ denote the standard basis vectors in $\mathbb{R}^n$. Finally, let $\rho$ be uniform over $\mathcal{C}$.

Firstly, note that $f_1$ can perfectly separate all pairs of classes $\{c_i, c_j\}$ using the classifier $\mathbf{e_i} - \mathbf{e_j}$ and thus has $L_{sup}(f_1) = 0$. On the other hand, $f_0$ trivially has loss $L_{sup}(f_0) = 1$. However, consider the case where the number of negative samples $k = \Omega(|C|)$. This means that there is a constant probability that $\exists i, c_i^- = c^+$. Furthermore, given this, with constant probability, $x, x^+ = x_i^2$ and $x^- = x_i^1$. In this case,

$$\begin{aligned} L_{un}(f_1) &= \Omega\left(log\left(1 + e^{1/2r\mathbf{e_i}(3/2r\mathbf{e_i} - 1/2r\mathbf{e_i})}\right)\right) \\ &\approx \Omega(log(1 + e^{\Omega(r^2)})) = \Omega(r^2) \end{aligned}$$

On the other hand, the $L_{un}(f_0) = O(1)$. Therefore, $\arg\min_{f \in \mathcal{F}} L_{un}(f) = f_0 \neq f_1$ despite having $L_{sup}(f_1) = 0$ and $L_{sup}(f_0) = 1$ and so the unsupervised algorithm will pick the suboptimal representation. □

Naively, one might think that the problem happens because $k = \Omega(|\mathcal{C}|)$ and so it is inevitable that points from the same class will occur both as positive samples and negative samples. The next example shows that even

when $k = o(|\mathcal{C}|)$, the same phenomenon can occur. The example is essentially an extension of Lemma 4.1 done to promote collision of points that lead to high unsupervised loss.

**Lemma 4.2.** *The statement of Lemma 4.1 holds in the case when* $k = o(|\mathcal{C}|)$

*Proof.* Let $\mathcal{C} = \{c_{ij}\}_{i,j=0}^{n}$ such that $|\mathcal{C}| = n^2$. Let each class $c_{ij}$ be uniform over the set $\{x_{ij}^1, x_{ij}^2\}$ and thus $|\mathcal{X}| = 2n^2$. As in Lemma 4.1, let $\mathcal{F} = \{f_0, f_1\}$ with $f_0 = \mathbf{0}$ and $f_1(x_{ij}^1) = 3/2r\mathbf{e_i}$, $f_1(x_{ij}^2) = 1/2r\mathbf{e_i}$. Thus, $f_1$ 'clusters' the $n^2$ classes into $n$ clusters.

Note that again, $L_{sup}(f_0) = 1$ on the average 2-way task and $L_{sup}(f_1) = 1/n = o(1)$ because the classification problem will consist of two classes from the same cluster only with probability $1/n$. Now, when $k = o(n)$, the probability of having a class in the negative samples be the same as the class of the positive samples is $1 - (1 - 1/n)^k$ and so $L_{un}(f_1) = o(1) < L_{un}(f_0) = \Omega(1)$. However, consider when $k = \Omega(n)$. Then, with constant probability a class $c_i^-$ from the same cluster as $c^+$ will be picked as a negative sample. Again, as above, $L_{un}(f_1) = \Omega(r^2)$ while $L_{un}(f_0) = O(1)$. This means that the algorithm will pick $f_0$ despite $L_{sup}(f_1) = o(1) < 1 = L_{sup}(f_0)$ □

Thus, we see that even when $k = O(\sqrt{|\mathcal{C}|})$, the unsupervised algorithm can pick suboptimal representations. The above example can easily be extended to the case when $k = O(poly(|\mathcal{C}|))$ for any fractional power too.

*Remark.* Note that while the example in Lemma 4.1 seems rather artificial, the example in Lemma 4.2 is plausible when the function class $\mathcal{F}$ is not very expressive and clumps certain inputs from related classes together.

Furthermore, notice that the number of negative samples at which negative sampling can have a detrimental effect scales with the number of *effective clusters* imposed by the function class $\mathcal{F}$. This hints that while the distributions $\mathcal{D}_{sim}, \mathcal{D}_{neg}$ can inherently prevent the unsupervised loss from being low, the function class being fit also plays a role.

Thus, we see that a large number of negative samples does not necessarily help in contrastive learning. Consequently, the number of negative samples to use becomes a hyper parameter to optimize over. In Section 5 we explore how the choice of negative samples affects performance empirically.

## 4.2 Utilizing Blocks of Similar Data

A dataset can often contain *blocks* of similar data instead of just pairs. Here, according to the framework, a block refers to $b+1$ i.i.d. draws of points $x, x_1^+, ..x_b^+ \sim \mathcal{D}_c(x)$ where $c \sim \rho$. For instance, in natural language processing, a paragraph of text can be seen as samples from the same class. For computer vision, frames within a certain window in a video can be thought of as samples from the same class too. How can one utilize this additional structure in data?

To do so, we propose a slightly modified loss function: One that uses a block $x, x_1^+, ..x_b^+$ of iid samples from $c^+ \sim \rho$ as a positive sample and a block $x_1^-, ..x_b^-$ of iid samples from $c^- \sim \rho$ as a negative sample.

**Definition 4.1.** *The unsupervised block loss function is given by*

$$L_{un}^{block}(f) = \mathbb{E}\left[\ell\left(f(x)^T\left(\frac{\sum_i f(x_i^+)}{b} - \frac{\sum_i f(x_i^-)}{b}\right)\right)\right]$$

This is reminiscent of the average of embeddings used in *word2vec*, where blocks correspond to windows of consecutive words.

To understand why this loss function make sense, recall that the connection between $L_{sup}$ and $L_{un}$ is made via applying Jensen's inequality (see [Arora et al., 2019]). Thus, the algorithm that uses the average of the positive and negative samples in blocks as a proxy for the classifier instead of just one point each should have a strictly better bound owing to the Jensen's inequality getting tighter.

We formalize this intuition below. Let $\tau$ be the probability that the class in the positive block and the negative block are the same. Then:

**Lemma 4.3.** $\forall f \in \mathcal{F}$

$$L_{sup}(f) \leq \frac{1}{1-\tau}\left(L_{un}^{block}(f) - \tau\right) \leq \frac{1}{1-\tau}\left(L_{un}(f) - \tau\right)$$

This bound tells us that $L_{un}^{block}$ is a better surrogate for $L_{sup}$, making it a more attractive choice than $L_{un}$ when larger blocks are available[1].

[1]Note that we don't compare the generalization error of the two loss functions.

*Proof.* By convexity of $\ell$,

$$\ell\left(f(x)^T\left(\frac{\sum_i f(x_i^+)}{b}-\frac{\sum_i f(x_i^-)}{b}\right)\right)=\ell\left(\frac{1}{b}\sum_i f(x)^T\left(f(x_i^+)-f(x_i^-)\right)\right)$$
$$\leq\frac{1}{b}\sum_i \ell\left(f(x)^T\left(f(x_i^+)-f(x_i^-)\right)\right)$$

Thus,

$$L_{un}^{block}(f)=\mathop{\mathbb{E}}_{\substack{x,x_i^+\\x_i^-}}\left[\ell\left(f(x)^T\left(\frac{\sum_i f(x_i^+)}{b}-\frac{\sum_i f(x_i^-)}{b}\right)\right)\right]$$
$$\leq\mathop{\mathbb{E}}_{\substack{x,x_i^+\\x_i^-}}\left[\frac{1}{b}\sum_i \ell\left(f(x)^T\left(f(x_i^+)-f(x_i^-)\right)\right)\right]$$
$$=L_{un}(f)$$

The proof of the lower bound is analogous to that of Lemma 4.3 in [Arora et al., 2019].

□

We verify the tightness of this bound by experimenting with this objective using function classes of practical interest and find that minimizing $L_{un}^{block}$ instead of $L_{un}$ can lead to better performance and our results are summarized in Section 5.

# 5 Experiments

While the above theory applies generally, it ignores issues of optimization over function classes as well as generalization. Thus, to emprically verify the insights, we conduct experiments using function classes of practical interest in computer vision to verify the above results. We now highlight the setup.

### Toy Dataset

For our experiments, use the CIFAR-100 dataset [Krizhevsky and Hinton, 2009]. CIFAR-100 consists of 100 classes containing 600 images each. We

use a train test split of 500/100. Each image is a $32 \times 32$ RGB image $x$ such that $x \in [0, 255]^{3072}$. We use a standard normalization scheme to normalize the pixel values around their means for all images. For training, we augment the dataset by randomly flipping the image and randomly cropping a $32 \times 32$ sub-image from a padded version with a padding of 4. The test set is not augmented. CIFAR-100 is a popular multiclass dataset used for training neural networks on computer vision tasks.

### Experiment Set Up

In order to create a toy unsupervise dataset, we recreate the settings of the framework. Specifically, to generate pairs of similar points, we pick a class uniformly at random from the 100 classes and then sample two pairs of images from that class. For the function class $\mathcal{F}$ we use the VGG-16 architechture [Simonyan and Zisserman, 2014]. We construct a representation function class by taking away the final classification layer and instead adding a $522 \times 100$ linear layer at the end to make 100 dimensional representations. VGG-16 is considered to be a standard baseline for classification on CIFAR-100. We train the network on the unsupervised loss stochastic gradient descent.

### Increasing Negative Samples

We study the effect of the number of negative samples. Figure 1 summarizes our findings. We tested with $k = 1, 2, 4, 10$ along with $M = 50, 250, 500, 1000$ random samples per class. Note that when we use $M$ samples per class we effectively use $M * 100$ unlabelled samples. The reason we sampled a fixed number of samples from each class was to prevent lack of diversity of unsupervised data for lower values of $M$.

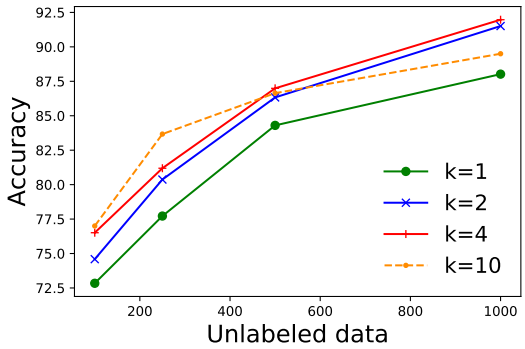


Figure 1: The effect on increasing negative samples

We find that increase in the number of negative samples leads to an increase in accuracy at first, but can eventually be detrimental. We observe such effects at $k = 10$ which is roughly $\sqrt{|\mathcal{C}|}$. At this level, the probability of collision is roughly $1/10$ and the probability of having two of the same classes as a negative sample is constant.

We note that this effect was expected when the number of negative samples increases to become comparable to the number of classes by Lemmas 4.1, 4.2. Negative results of this form are rare in empirical work and we hope that study of the effect of increasing negative samples leads to future empirical and theoretical work towards *negative* results. Table 1 presents the final accuracies of trained models on the test set. We note importantly that the accuracy is measured on representations of previously unseen images. This hints towards the fact that the inductive bias of the function class plays a role too (since the number of parameters in the net far exceed the number of training samples).

### Increasing Block Size

We now present our results on increasing the size of blocks. For this experiment, we considered the set up with just 1 negative sample (as in Lemma 4.3. Our results are summarized in Table 5. We find that increasing the block size does lead to an increase in accuracy. For this particular experiment, the accuracy was computed by training the best classifier on a small set of labelled data.

We note that for the sake of this experiment, we maintained that the *total*

| | Data | | | |
|---|---|---|---|---|
| Neg. Samples | $M = 50$ | $M = 250$ | $M = 500$ | $M = 1000$ |
| k=1 | 72.84 | 86.18 | 84.30 | 88.02 |
| k=2 | 74.59 | 80.37 | 86.33 | 91.51 |
| k=4 | 76.51 | 81.19 | 86.99 | 91.97 |
| k=10 | 77.00 | 83.67 | 86.64 | 88.01 |

Table 1: Effect of increase in negative samples with different amounts of data

| | Block Size | | |
|---|---|---|---|
| Method | $b = 2$ | $b = 5$ | $b = 10$ |
| CURL | 88.12 | 89.62 | 89.72 |

Table 2: Effect of increase in block size

*amount* of data used was constant. In particular, we ensured that $b \times M$ was a constant accross all experiments. For this experiment we used 500 images per class for total of 500000 randomly sampled unlabelled points. Further experiments inspired by this insights were conducted in Arora et al. [2019] in the domain of natural language processing. Arora et al. [2019] find an increase in performance of the state-of-the-art sentence embeddings on the IMDB Dataset.

# 6 Conclusion

Contrastive learning methods have been empirically successful at learning useful feature representations. The framework of Arora et al. [2019] gives fresh insights into what guarantees are possible and impossible, and shapes the search for new assumptions to add to the framework that allow tighter guarantees. The framework currently ignores issues of efficient minimization of various loss functions, and instead studies the interrelationships of their minimizers as well as sample complexity requirements for training to generalize. However, this report is unable to show a full generalization result

in the case of modified objective as in Lemme 4.3 which is left for future work. While the insights from the framework are simple, one experiment on sentence embeddings already illustrates how fresh insights derived from our framework can lead to improvements upon state-of-the-art models in this active area. We hope that further progress will follow, and that our theoretical insights will begin to influence practice, including design of new heuristics to identify semantically similar/dissimilar pairs.

# References


Sanjeev Arora, Hrishikesh Khandeparkar, Mikhail Khodak, Orestis Plevrakis, and Nikunj Saunshi. A theoretical analysis of contrastive unsupervised representation learning. *arXiv preprint arXiv:1902.09229*, 2019.

Aurélien Bellet, Amaury Habrard, and Marc Sebban. Similarity learning for provably accurate sparse linear classification. *arXiv preprint arXiv:1206.6476*, 2012.

Rich Caruana and Alexandru Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd international conference on Machine learning*, pages 161–168. ACM, 2006.

Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 297–304, 2010.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Andreas Maurer, Massimiliano Pontil, and Bernardino Romera-Paredes. The benefit of multitask representation learning. *The Journal of Machine Learning Research*, 17(1):2853–2884, 2016.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

L. G. Valiant. A theory of the learnable. In *Proceedings of the Sixteenth Annual ACM Symposium on Theory of Computing*, STOC '84, pages 436–445, New York, NY, USA, 1984. ACM. ISBN 0-89791-133-4. doi: 10.1145/800057.808710. URL http://doi.acm.org/10.1145/800057.808710.

Xiaolong Wang and Abhinav Gupta. Unsupervised learning of visual representations using videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2794–2802, 2015.