Sensitivity of sparse codes to image distortions

Kyle Luther[1, 2]
H. Sebastian Seung[2, 3]
[1]Department of Physics, Princeton University, Princeton, New Jersey 08544, USA
[2]Neuroscience Institute, Princeton University, Princeton, New Jersey 08544, USA
[3]Department of Computer Science, Princeton University, Princeton, New Jersey 08544, USA

Keywords: Sparse coding, representation learning, object recognition, generative modeling

Abstract

Sparse coding has been proposed as a theory of visual cortex and as an unsupervised algorithm for learning representations. We show empirically with the MNIST dataset that sparse codes can be very sensitive to image distortions, a behavior that may hinder invariant object recognition. A locally linear analysis suggests that the sensitivity is due to the existence of linear combinations of active dictionary elements with high cancellation. A nearest neighbor classifier is shown to perform worse on sparse codes than original images. For a linear classifier with a sufficiently large number of labeled examples, sparse codes are shown to yield higher accuracy than original images, but no higher than a representation computed by a random feedforward net. Sensitivity to distortions seems to be a basic property of sparse codes, and one should be aware of this property when applying sparse codes to invariant object recognition.

1 Introduction

Given a generative model f for images, one can encode an image $\mathbf{x}$ by a representation vector $\mathbf{r}$ such that $\mathbf{x} \approx f(\mathbf{r})$. While generative models have many possible uses, this paper is concerned with their application to visual object recognition. In this approach, a classifier is trained to predict the object class based on the representation vector rather than the image. It is hoped that the approach will reduce the number of labels required to train the classifier, or reduce the complexity of the classifier.

The approach has a rich history. It has been tried with variational autoencoders (Kingma et al., 2014), generative adversarial networks (Radford et al., 2016), denoising autoencoders (Vincent et al., 2010), sparse coding (Raina et al., 2007), non-negative matrix factorization (Guillamet et al., 2002), and principal components analysis (Turk and Pentland, 1991). However, the approach has largely been abandoned in favor of purely supervised training of classifiers that use the image as input. More recently, progress has been made with a contrastive approach to representation learning in which network output is trained to be invariant to a hand-designed set of augmentations (Dosovitskiy et al., 2014; Chen et al., 2020; He et al., 2020; Caron et al., 2020; Zbontar et al., 2021).

Nevertheless, generative models remain appealing because neither labels nor a hand-designed set of invariances are needed to train them. The goal of this paper is to identify and study an issue that might be hindering the success of the generative model approach to object recognition. For concreteness, we focus our attention on sparse linear generative models, otherwise known as *sparse coding* (Olshausen and Field, 1996). The goal of sparse coding is to learn a dictionary $\mathbf{D} \in \mathbb{R}^{m \times n}$ and for every image $\mathbf{x} \in \mathbb{R}^m$ a sparse representation $\mathbf{r} \in \mathbb{R}^n$ such that $\mathbf{x}$ is approximated by a linear combination of a small subset of dictionary elements: $\mathbf{x} \approx \mathbf{D}\mathbf{r}$.

Sparse coding has been proposed as a theory of visual cortex, because the representations can be computed by a recurrent neural network in which neurons inhibit each other through lateral interactions, the receptive fields of neurons end up resembling Gabor filters, and the representations are sparse (Olshausen and Field, 1997).

The goal of our paper is to point out sparse codes can be very sensitive to image distortions, a property that may ultimately be at odds with the task of object recognition. We demonstrate this empirically using the MNIST images of handwritten digits. Sparse codes turn out to be more sensitive to an image distortion that preserves the digit class than to an image perturbation that can change the digit class. Furthermore, this sensitivity emerges in the sparse codes; it is not present in the original images.

To explain these empirical findings, we use the locally linear relationship between the sparse code and the image, which is based on the dictionary elements or filters of the sparse code. Sensitivity is related to the existence of linear combinations of active filters with high cancellation. These define directions in image space that turn out to be well-aligned with image distortions for MNIST, leading to high sensitivity.

We examine the impact of sensitivity to distortions on downstream classification performance. Nearest neighbor classifiers based on sparse codes significantly underperform those based on raw image pixels. For a linear classifier with a sufficiently large number of labeled examples, we find that sparse codes can yield higher accuracy than raw image pixels, consistent with previous work (Raina et al., 2007). However, the increased linear classifier accuracy is matched by a representation computed by a random feedforward net (Jarrett et al., 2009). For a linear classifier with few labeled examples, sparse codes are outperformed by both the raw image pixels and the random net representation.

Our results seem to suggest that sparse coding is useful for downstream recognition only insofar as it increases linear separability, but also introduces a high sensitivity to distortions which may ultimately be at odds with the task of invariant object recognition.

Some researchers do not take note of the sensitivity issue or do not regard it as problematic (Raina et al., 2007). We have also encountered colleagues who believe that the Restricted Isometry Property (RIP) from compressed sensing (Candes, 2008) guarantees that there is no sensitivity issue. In fact, there is no reason to expect that the RIP should apply to sparse coding, and we will empirically demonstrate RIP violation in the main text.

Other researchers have introduced postprocessing mechanisms to reduce the sensitivity of sparse codes. Spatial pooling has been applied to convolutional sparse codes (Hu et al., 2014; Yang et al., 2009). Chen et al. (2018) used temporal information to learn a linear pooling operator, similar in spirit to applying slow feature analysis to sparse codes, with the goal of linearizing representation trajectories over time. Other work has modified the original formulation of sparse coding to use complex basis functions, where the phase encodes position and the amplitude encodes a local invariance (Cadieu and Olshausen, 2012).

These previous works can be viewed as attempts to reduce the sensitivity behavior. Here we provide an explicit formulation of the behavior: the sparse codes themselves may be more sensitive than the raw pixels to small input distortions. We hope that it will be a helpful starting point for other researchers who hope to make sparse coding competitive with purely supervised approaches to object recognition, or with the popular contrastive approach to self-supervised representation learning (Chen et al., 2020; Caron et al., 2020; Grill et al., 2020).

We also note that the contrastive approach works by training a net to be insensitive to image distortions that preserve the object class. In this sense, our formulation of the sensitivity behavior views sparse coding through a lens provided by the contrastive approach to representation learning.

2 Sparse codes are sensitive to image distortions

In this section, we train sparse coding on the MNIST dataset (Deng, 2012) and measure the sensitivity of the generated representations, defined by the norm of their directional derivative, to image perturbations shown in Figure 1. The motivation for this experiment is to measure the invariance properties of the generated representations. The fundamental challenge of visual object recognition is knowing which of the enormous number of pixel variations are irrelevant for discrimination and which are relevant for discrimination (DiCarlo et al., 2012). If the goal of representation learning is ultimately to generate representations which are useful for downstream recognition, it seems natural to expect them to be less sensitive to this irrelevant variability than they are to the relevant variability. This is known as invariant representation learning, an idea that has been around for decades (Fukushima, 1980).

The distortions represent a source of variability which does not change the class. Swaps, i.e. subtracting off the image and adding in another image, are important. Additive gaussian noise serves as a control in this experiment. The experiments probe responses to infinitesimal perturbations, which can be related to the subsequent theoretical analysis in Sections 3 and 4. Finite perturbations give qualitatively similar results (data not shown).

2.1 Learning sparse codes

Given a dataset $\{\mathbf{x}(t) \in \mathbb{R}^m : t = 1, 2, \ldots, T\}$, a popular objective used to learn the dictionary $\mathbf{D} \in \mathbb{R}^{m\times n}$ and representations $\{\mathbf{r}(t) \in \mathbb{R}^m : t = 1, 2, \ldots, T\}$ was introduced by Olshausen and Field (1996):

$$\min_{\mathbf{D},\mathbf{r}(t)} \sum_{t=1}^{T} \frac{1}{2} \|\mathbf{x}(t) - \mathbf{D}\mathbf{r}(t)\|^2 + \lambda \|\mathbf{r}(t)\|_1 \text{ such that } \|\mathbf{d}_i\| = 1 \;\forall\; i \tag{1}$$

In this paper we use the notation $\|\mathbf{u}\|$ indicates the l_2 norm of the vector $\mathbf{u}$. To indicate the l_1 norm we write $\|\mathbf{u}\|_1$. The hyperparameter $\lambda \geq 0$ controls the degree of sparseness. The dictionary elements are normalized to be unit vectors, $\|\mathbf{d}_i\| = 1$, which prevents the representations from driving to zero to minimize their l_1 norm.

We minimized Equation 1 for the 50,000 handwritten digits from the training split of the MNIST dataset. Both the image and representation dimensionality are 784. We vary $\lambda \in \{0.03, 0.10, 0.30, 0.90, 3.0\}$.

The minimization is performed with an alternating gradient descent-like algorithm. Both the dictionary and representations are initialized from a unit normal distribution. For every training iteration we apply an ISTA update to the representations, and a gradient step to the dictionary, followed by normalization of every dictionary element $\mathbf{d}_i \leftarrow \mathbf{d}_i/\|\mathbf{d}_i\|$. The ISTA updates are explained in Eqs. 1.4,1.4 of Beck and Teboulle (2009).

Separate learning rates for both $\mathbf{D}$ and $\mathbf{r}$ are chosen automatically and are updated at each step. Specifically, if the update decreased the loss, the update is accepted and the learning rate is multiplied by 1.1. If the loss increased, the update is rejected and the learning rate is multiplied by 0.5.

We then freeze the dictionary and continue to run ISTA updates (up to 200k iterations) until the representations are sufficiently close to optimal (defined in the Appendix) given the fixed dictionary. Reconstructions are highly accurate, and closely resemble the original images.

2.2 Sensitivity of sparse codes

If the image $\mathbf{x}$ is perturbed by $\Delta\mathbf{x}$, its representation $\mathbf{r}$ changes by $\Delta\mathbf{r}$ (Figure 1a). We quantify sensitivity by the norm of the directional derivative $\|\Delta\mathbf{r}\|/\|\Delta\mathbf{x}\|$, to three types of infinitesimal perturbations $\Delta\mathbf{x}$: adding gaussian noise, swapping to another image, and distorting the image (Figure 1b). To compute the directional derivative $\|\Delta\mathbf{r}\|/\|\Delta\mathbf{x}\|$, we compute the Jacobian matrix $d\mathbf{r}/d\mathbf{x}$ (see Section 4 for equation). We then compute $\|d\mathbf{r}/d\mathbf{x} \cdot \Delta\mathbf{x}\|/\|\Delta\mathbf{x}\|$. This is repeated for 4000 samples and we show the distribution of the directional derivatives in Figure 1c.

For noise perturbations we set $\Delta\mathbf{x} = \mathcal{N}(0, I)$. For swap perturbations, we sample another image $\mathbf{x}_{swapped}$ and set $\Delta\mathbf{x} = \mathbf{x}_{swapped} - \mathbf{x}$. For distortion perturbations, we use the *elasticdeform* augmentation module (van Tulder, 2021) which elastically warps the images as described in Ronneberger et al. (2015). For this experiment, a 5×5 grid is generated with displacements sampled from a Gaussian distribution with $1/4$ pixel standard deviation. A 28×28 displacement field is then generated by upsampling

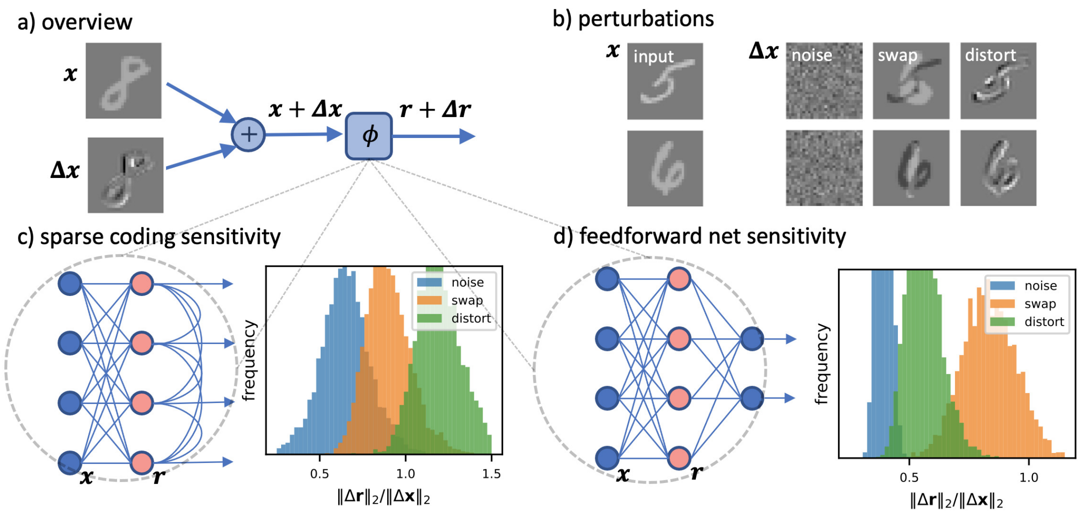

Figure 1: Representations generated by sparse coding are more sensitive to image distortions than to swapping the image. The the opposite behavior when examining the hidden layer of a fully connected network trained with supervised learning to discriminate digits. The sensitivity is measured by the norm of the directional derivative $\|\Delta \mathbf{r}\|/\|\Delta \mathbf{x}\|$ where $\Delta \mathbf{x}$ is an infinitesimal perturbation. Both representations are relatively less sensitive to noise than either warps or swaps.

with bicubic interpolation and the image is distorted with this displacement field. The distortions are small but observable if one carefully compares two different distortions of the same image.

The sparse codes turn out to be most sensitive to distortions, followed by swaps, and finally noise (Figure 1c). Intuitively the noise invariance property makes sense: the network does not model noise, and this result suggests the network is at least just partially ignoring the noise. This is likely related to the well-known image denoising properties of sparse codes (Elad and Aharon, 2006). The important behavior here is the greater sensitivity to distortions than swaps. This behavior is opposite that of a digit classifier, which should be less sensitive to distortions than swaps. (Swaps change the image to another randomly drawn image, usually that of another class altogether.) Therefore, the sparse codes seem to be making the classification task harder.

We note that the directional derivative $\|\Delta \mathbf{r}\|/\|\Delta \mathbf{x}\|$ is normalized, so that the differences between the histograms in Figure 1c do not arise from differences in the size of perturbations $\Delta \mathbf{x}$. It is instructive to consider the baseline case where the image pixels themselves are used as the representation, $\mathbf{r} = \mathbf{x}$. In this case, $\|\Delta \mathbf{r}\|/\|\Delta \mathbf{x}\| = 1$ for all kinds of perturbations (distortion, swap, and noise). In other words, the image itself is equally sensitive to all kinds of perturbations.

In summary, we have found that sparse codes are more sensitive to distortions than to swaps. This sensitivity emerges in the sparse code and is not inherited from the image, which is equally sensitive to all kinds of perturbations. In other words, representing an image by a sparse code appears to take a step away from invariant object recognition, rather than a step towards it. The Appendix repeats the experiment of Fig. 1 for multiple values of the sparsity parameter λ, and shows that greater sensitivity to distortions than swaps is a robust finding.

2.3 Comparison with hidden layer of supervised feedforward net

For comparison we also train a two layer fully connected network with supervised backpropagation to classify digits. The hidden layer is 784 dimensional, the same as our sparse representations, and the output is 10 dimensional. The hidden layer uses ReLU nonlinearity. We optimize the cross-entropy loss with stochastic gradient descent using a batch size of 64. We do not augment the images so the images are exactly the same ones used to train the sparse coding model. The network is trained for 1000 steps with a learning rate of 0.1 then 1000 more steps with a learning rate of 0.01. It achieves 98.7% training accuracy and 97.7% validation accuracy. This not intended to be a state-of-the-art network, but is sufficient for our purpose of comparison.

We consider the hidden layer of this network as a representation, and quantify its sensitivity to the same perturbations as before (Figure 1 d). This representation turns out to be less sensitive to distortions than to swaps. In this sense, the hidden layer representation is taking a step towards invariant object recognition. This may not be surprising, as the network was trained to classify digits. Still, the hidden layer serves as an informative contrast with the sparse codes.

3 Locally linear representations

To help explain the preceding empirical findings, we provide a locally linear analysis of sparse codes in the following. For a fixed dictionary $\mathbf{D}$, the representation $\mathbf{r}$ of an image $\mathbf{x}$ is defined by minimizing the objective in Eq. (1) with respect to $\mathbf{r}$. The solution satisfies

$$r_i = S_\lambda\left(\mathbf{d}_i \cdot \mathbf{x} - \sum_{j, j\neq i} (\mathbf{D}^\top \mathbf{D})_{ij} r_j\right) \tag{2}$$

We will assume that the argument of the shrinkage function

$$S_\lambda(u) = \begin{cases} u - \lambda, & u \geq \lambda \\ 0, & |u| \leq \lambda \\ u + \lambda, & u \leq -\lambda \end{cases} \tag{3}$$

is "not exactly at threshold," i.e., not equal to $\pm\lambda$ for any i. Violations of this assumption occur only for nongeneric images. Equation (2) has the form of a neural net in which $\mathbf{x}$ is the input and $\mathbf{r}$ is the output. This is the opposite of the generative model, in which $\mathbf{r}$ is the input and $\mathbf{x}$ is the output. In the neural net, $\mathbf{d}_i \cdot \mathbf{x}$ is a linear filtering of the image, so we will use the terms "filter" and "dictionary element" interchangeably to refer to $\mathbf{d}_i$.

The nonzero elements of $\mathbf{r}$, the active representation $\mathbf{r}_+$, can be written as a locally linear function of $\mathbf{x}$

$$\mathbf{r}_+ = (\mathbf{D}_+^\top \mathbf{D}_+)^{-1} \left(\mathbf{D}_+^\top \mathbf{x} - \lambda \mathbf{s}_+\right) \tag{4}$$

Here $\mathbf{D}_+$ is the active dictionary and $\mathbf{s}_+$ contains the signs of the elements in $\mathbf{r}_+$.

A sufficient condition for uniqueness of $\mathbf{r}$ is that no dictionary elements are duplicates or antipodal pairs. (This follows from Lemma 3 of Tibshirani (2013), assuming that all dictionary elements are unit vectors.) The network used for Figure 1 satisfies this condition. Another sufficient (and almost necessary) condition for uniqueness is that $\mathbf{D}_+$ should be nonsingular (Tibshirani, 2013), in which case the matrix inverse in Eq. (4) is well-defined.

4 Non-orthogonality of filters can lead to sensitivity

This sensitivity to distortions arises from the non-orthogonality of the dictionary. We'll argue that this sensitivity can partially be understood by considering the response to image perturbations which align with the difference $\mathbf{d}_a - \mathbf{d}_b$ between overlapping filters. We'll show that the representations generated by lasso inference are more sensitive to this difference than to either filter individually. For MNIST digits, the difference between the most overlapping pairs of filters often more closely resembles a spatial derivative of a filter, than the filter itself.

4.1 Representations are more sensitive to filter differences to the filters themselves

We start by considering two simple perturbations: one in the direction of an active filter $\Delta\mathbf{x}_1 = \epsilon\mathbf{d}_1$ and one in the direction of the difference between active filters $\Delta\mathbf{x}_{1,2} = \epsilon[\mathbf{d}_1 - \mathbf{d}_2]$. Using Eq. (4), it can be shown that the responses to these perturbations are $\Delta\mathbf{r}_1 = \{+\epsilon, 0, 0, \ldots, 0\}$ and $\Delta\mathbf{r}_{1,2} = \{+\epsilon, -\epsilon, 0, \ldots, 0\}$ for sufficiently small ϵ, given the "not exactly at threshold" assumption. So the directional derivatives for the perturbations are

$$\frac{\|\Delta\mathbf{r}_1\|}{\|\Delta\mathbf{x}_1\|} = 1 \qquad \frac{\|\Delta\mathbf{r}_{1,2}\|}{\|\Delta\mathbf{x}_{1,2}\|} = \frac{1}{\sqrt{1 - \mathbf{d}_1 \cdot \mathbf{d}_2}} \tag{5}$$

using the fact that the filters are unit vectors. According to the second equation, $\mathbf{r}$ becomes more sensitive to difference perturbations as the overlap $\mathbf{d}_1 \cdot \mathbf{d}_2$ increases. The sensitivity diverges as the overlap approaches 1. When the overlap is exactly one, there is a continuum of minima for Eq. (1) and the representation is not unique. In this regard, the high sensitivity to difference perturbations can be regarded as a soft version of the non-uniqueness seen when there are two identical filters.

4.2 Filter differences resemble spatial derivatives for the MNIST experiment

Given Eq. (5), it is natural to ask whether in practice there exist filter pairs with high overlap. Figure 2 returns to MNIST, and shows pairs of active filters with overlap greater than 0.5. The filters resemble parts (or negative parts) of digits. The differences between filters resemble distortions of the filters. Notably most filters have a significant mean value. For the 10 filter pairs in Fig. 2c, the average of the normalized means $\frac{1}{10}\sum_{i=1}^{10}\sum_m d^a_{i,m}/\sum_m |d^a_{i,m}|$ is 0.66. The average of the mean of the difference

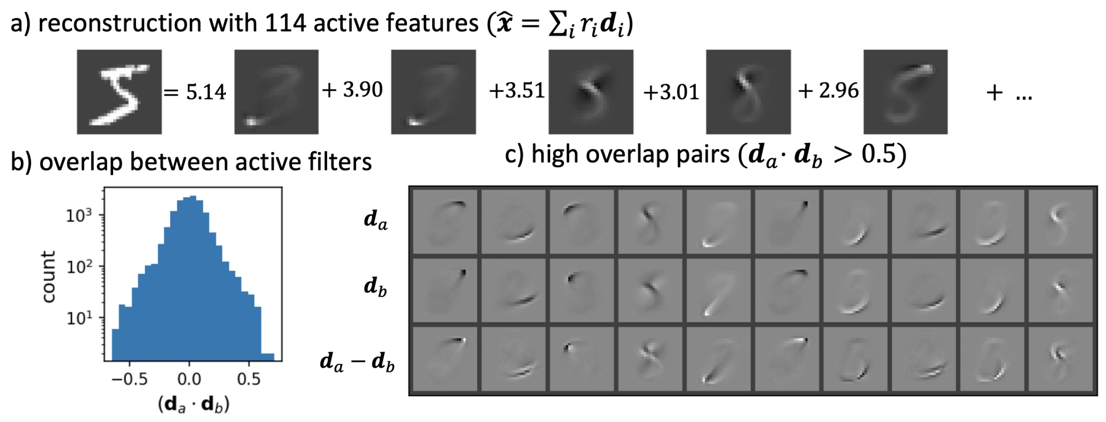

Figure 2: Sparse codes learned from MNIST lead to non-orthogonal filters. In (a) we show the reconstruction of the first MNIST digit and 5 of the 122 filters used in the reconstruction. In (b) we show the overlap distribution between all pairs of the 122 active filters. In (c) we show 10 pairs of filters with high overlap and their differences. The differences often more closely resemble spatial derivatives of the filters. The representations are more sensitive to image perturbations which look like these differences than perturbations which look like the filters themselves.

$\frac{1}{10}\sum_{i=1}^{10}\sum_m (d^a_{i,m} - d^b_{i,m})/\sum_m |d^a_{i,m} - d^b_{i,m}|$ is only 0.09. This suggests that adding the filter difference is more like a transport/displacement of some of the pixels, compared to adding one of the filters which is more like an addition or removal of pixels.

4.3 Why are the most similar filters distorted versions of one another?

The observation that the most similar filters appear as distorted versions of one another is consistent with the predictions made in the sparse manifold transform framework (Chen et al., 2018). Specifically it is postulated that the dictionary learned by sparse coding has "has an organization that is a discrete sampling of a low-dimensional, smooth manifold" (Chen et al., 2018). Additionally it is postulated that the representations and dictionary act as a set of steerable filter which act to approximate continuous variations by. The idea is that in general, an oriented edge can show up at any continuous position in the image and will never perfectly align with an oriented edge in the dictionary. Weighted averages of dictionary elements can be used to approximate the continuous position of this edge.

The novelty of our work is to consider the implications of steerability when the filters being used to steer are non-orthogonal. The key theoretical insight is that when these filters have some positive overlap (i.e. they are discretely but sufficiently close along a manifold) the representations are extremely sensitive to small displacements of the image along the manifold.

5 High sensitivity due to a highly cancelling combination of active filters

In the previous section we showed that sparse coding can be sensitive to an image perturbation that is a linear combination of two filters that overlap, or equivalently tend to cancel each other. We now show that the sensitivity can be even higher if there exists a linear combination of more than two filters with more cancellation.

5.1 Singular values and gain of amplification

From the locally linear Eq. (4), we can write the "active Jacobian" of the $\mathbf{x} \to \mathbf{r}$ map as the $k \times m$ matrix

$$\mathbf{J}_+ := \frac{d\mathbf{r}_+}{d\mathbf{x}} = (\mathbf{D}_+^\top \mathbf{D}_+)^{-1} \mathbf{D}_+^\top \tag{6}$$

The $n - k$ rows of the Jacobian that are outside the active set have been omitted because they vanish. We will also refer to the active Jacobian as the "gain matrix," because it helps us conceptualize the linear response as amplification by different gains for different directions in the image space.

The largest singular value of the gain matrix is the reciprocal of the smallest singular value of the active dictionary $\mathbf{D}_+$ Therefore, the representation $\mathbf{r}$ has the potential to be sensitive to changes in the image $\mathbf{x}$ if the active dictionary has a small singular value. We can write the smallest singular value σ^* in variational form as

$$\sigma^* = \min_{\mathbf{v}} \|\mathbf{D}_+ \mathbf{v}\| \text{ such that } \|\mathbf{v}\| = 1 \tag{7}$$

The solution $\mathbf{v}^*$ is a vector in the representation space, and can be interpreted as the coefficients of a linear combination of dictionary elements with maximum cancellation. The corresponding vector $\mathbf{u}^* \propto D_+ \mathbf{v}^*$ in the image space is the direction that exhibits amplification with maximum gain $1/\sigma^*$. Both $\mathbf{v}^*$ and $\mathbf{u}^*$ are columns of the orthogonal matrices in the singular value decomposition (SVD).

5.2 Image distortions align with high gain directions

Returning to our MNIST experiment, we can find the image space direction of "maximum cancellation" by computing the SVD of the active dictionary or active Jacobian. The results are displayed in Fig. 3a for the first MNIST digit. The smallest singular value of the active dictionary is 0.158, corresponding to a maximum gain of 6.33.

For comparison, achieving the same gain with a filter pair as in Eq. (5) would require overlap of 0.975. In fact, the maximum overlap between filters in the active dictionary is about 0.6 (see Fig. 2b), which amounts to a gain of only about 1.6 according to Eq. (5). The noisy appearance of the corresponding singular vector is consistent with the idea of maximum cancellation between active dictionary elements as in Eq. (7).

We define the power spectrum of an image perturbation as the square of its overlaps on the singular vectors in the image space. The power spectrum always sums to one, assuming that the image perturbation is a unit vector. However, the distribution of power over the singular vectors differs for the different kinds of perturbations that were

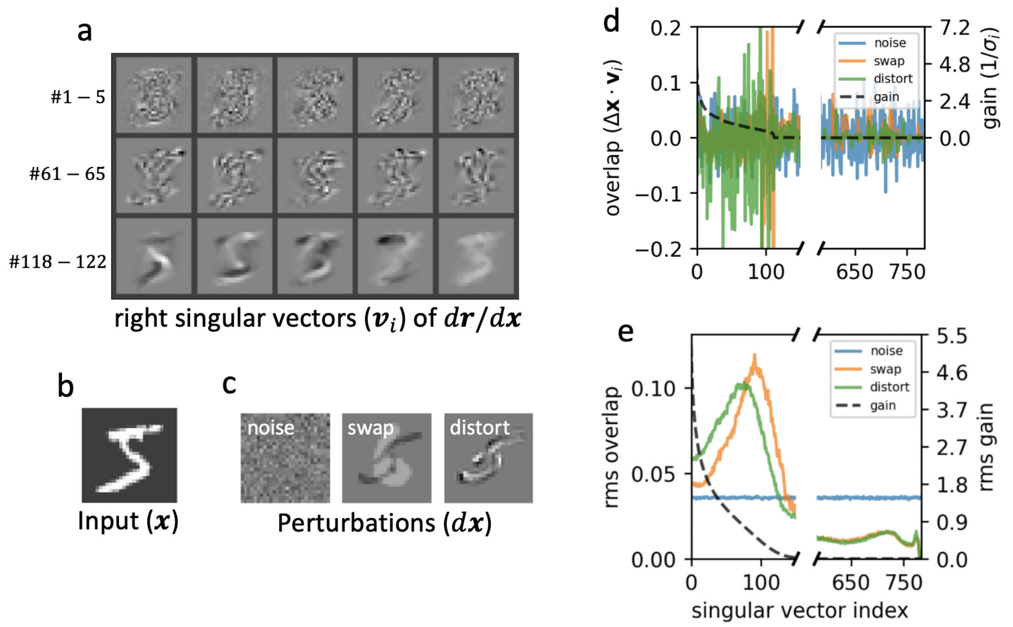

Figure 3: Singular value decomposition of the active dictionary, $\mathbf{D}_+ = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$, for the MNIST digit shown in b. In (a) we show singular vectors in the image space (columns of $\mathbf{V}$), displayed as 28×28 images in order of decreasing gain $1/\sigma$, where σ denotes singular value. The vector with highest gain (top left) appears to be a high frequency noisy image, and the vector with lowest gain (bottom right) appears to be a blurred version of the original image. Only the top 122 singular vectors with nonzero gain are shown; they span the same subspace as the active dictionary. In (c) we show examples of three types of input perturbations. In (d) we plot the dot product between the singular vectors and the three input perturbations. We also plot the gain ($1/\sigma_i$) for each of these singular values. The gains vary by $20\times$ from the largest to the smallest nonzero gain. The perturbations of Fig. 1 can be expressed as linear combinations of the singular vectors (colored lines). (e) Root-mean-squared overlap ("amplitude spectrum") computed with the same 4000 perturbations and digits used to generate Fig. 3c. From noise to swap to distortions, the amplitude spectrum shifts towards the high gain vectors, explaining why the sensitivity of the representation increases.

studied in Fig. 1. A noise perturbation has equal power on average over all singular vectors (Fig. 3c). That means most of the power of a noise perturbation is in the numerous singular vectors with zero gain, consistent with the experimental observation in Fig. 1 that sparse coding is robust to noise perturbations.

In contrast, both swaps and distortions have relatively little power in the singular vectors with zero gain, and more power in the singular vectors with nonzero gain. Furthermore, the power spectrum for distortions is shifted to the left, towards the singular vectors with higher gain. This is consistent with the experimental observation that sparse coding is more sensitive to distortions than swaps. Subjectively, the singular vectors with lowest nonzero gain do have more resemblance to swaps than distortions (Fig. 3), consistent with the difference between swaps and distortions in the average power spectrum.

5.3 Violation of Restricted Isometry Property

In the field of compressed sensing, the dictionary $\mathbf{D}$ is often designed to satisfy the Restricted Isometry Property (RIP) (Candes, 2008), meaning that the norms of $\mathbf{Dr}$ and $\mathbf{r}$ are approximately equal for sparse $\mathbf{r}$. In sparse coding, the dictionary is learned rather than designed. Figure 3 shows that the singular values of the active dictionary $\mathbf{D}_+$ can span a large range, so that the RIP is far from being satisfied.

6 Supervised evaluation of sparse representations

While generative models have many applications, here we are specifically interested in using the learned representations for classification. We evaluate four different sets of representations using nearest neighbor and linear classifiers. The representations we evaluate are given by 1) raw pixels $(\mathbf{x})$ 2) sparse coding $(\mathbf{r})$ with $\lambda = 0.3$ 3) the hidden layer of a 2 layer net trained to recognize digits $(\mathbf{h})$ and 4) the 2nd layer of a wide network with random weights $(\mathbf{z})$. The first 3 are described in section 2. We describe the random net representations below. The results for $\lambda = 0.03, 0.1, 0.9, 3.0\}$ are similar and shown in the Appendix.

6.1 Random feature baseline

We evaluate our classifiers on a set of representations generated by a random 2-layer feedforward network with rectifying nonlinearity. Specifically, we create two matrices $\mathbf{W}_1$ of size 7840×784 and $\mathbf{W}_2$ of size 7840×7840 by sampling all elements iid from a unit normal distribution $W_{ij} \in \mathcal{N}(0, 1)$. Defining $f(x)$ as the rectified linear unit (ReLU) nonlinearity, our representation $\mathbf{z}$ is generated by applying the 2 layer network to the image:

$$\mathbf{z}(\mathbf{x}) = f(\mathbf{W}_2 \cdot f(\mathbf{W}_1 \cdot \mathbf{x})) \tag{8}$$

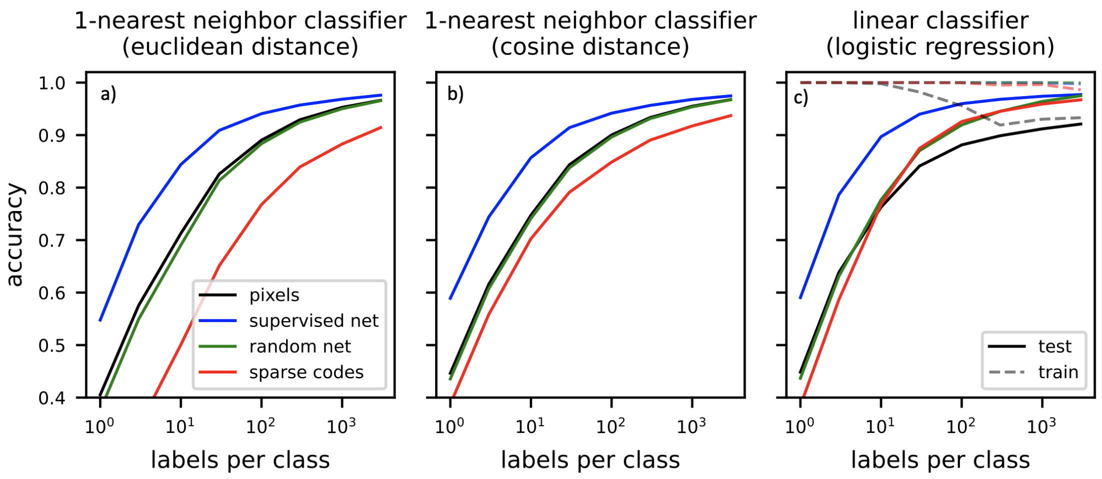

Figure 4: Supervised evaluation of four representations: pixels, sparse coding with $\lambda = 0.3$, the hidden layer of a two-layer net trained via backpropagation to recognize digits, and a wide network with two layers of random weights (Eq. 8). For nearest neighbor classification (Euclidean or cosine distance), pixels and the random net yield nearly identical performance. The sparse representation is uniformly worse than pixels, and the hidden layer of the trained net is uniformly better. For linear classification, the sparse representation is worse than pixels with few labels and better than pixels with many labels. However, the sparse representation never outperforms the random net. In all cases, the hidden layer of the backprop-trained net is uniformly better than all other representations.

6.2 Nearest Neighbor Classification

We train a 1-nearest neighbor classifier on the representations. We compare euclidean and cosine distance. We train every configuration using $k \in \{1, 3, 10, 30, 100, 300, 1000, 3000\}$ labels per class. We average the accuracy for every configuration over 5 random seeds. In Figure 4, we show the results for both Euclidean and cosine distance.

As expected the random features and pixel representations have nearly identical performance (the random representations do not change the ordering of the distances so long as the network is sufficiently wide). However, the sparse representations are markedly worse for all numbers of labels. This makes sense in light of our previous experiments: the representations are more sensitive to small translations than rescaling and swapping images. Intuitively, translations and deformations are precisely the sorts of variability that a representation should be insensitive to if a nearest neighbor classifier is to be trained on top of it.

Interestingly, using cosine distance instead of Euclidean distance seems to have the largest positive impact on the sparse representations, though they still lag behind the other representations for all numbers of labels. This suggests that the norm of the sparse representations may be relatively unstable compared to pixel representations and by using cosine distance, we ignore variations in the length of the representation vector.

It is easy to dismiss the results of this nearest neighbor classification because it is not a particularly "smart" classifier: it makes few assumptions about the data and in many practical datasets it requires too many labels to achieve acceptably high generalization performance. However we believe this strong reduction in generalization performance shows a very real and important property of the sparse representations.

First, the criticism of the classifier as being too weak is not as relevant for MNIST digits: the classifier achieves 97.5% validation accuracy when trained on pixels using Euclidean distance. Essentially this classifier works well on pixels, and the sparse representations "break" this classifier. This reduction in performance can partially be mitigated by normalizing the representations (i.e. using cosine instead of Euclidean distance) although not completely.

Second, this classifier provides a somewhat direct evaluation of the geometry of representations. This classifier gives us a formal method for measuring the degree of invariance of a set of representations. If the representations are in fact invariant to transformations of the image which are not important for recognition, while being sensitive to transformations which are important, then a nearest neighbor classifier will perform quite well. If the classifier does worse, this suggests the representations are "less invariant" than the original image.

Third, this classifier is in fact used to evaluate modern successful self-supervised learning systems (Caron et al., 2020). Pretraining networks to be distortion invariant often results in substantial improvements in nearest neighbor classification even on much more challenging datasets like IMAGENET.

Fourth, the issue of training accuracy is not relevant for a nearest neighbor classifier. This makes the results simpler to interpret than most other classifiers. Even for a linear classifier, understanding generalization can be complicated in part because the representation may be improving linear separability, an effect easily achieved with random nonlinear features as we'll show in the next section.

6.3 Linear Classification

We train a standard linear classifier with logistic regression on the representations. We use a variable number of labels and 5 seeds for configuration like we did with for nearest neighbor classifiers. Additionally, we have an another parameter λ_w controlling the strength of weight decay $\|\mathbf{w}\|$. For every configuration, we find the $\lambda_w \in \{1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1\}$ which yields the highest test accuracy.

With more than 10 labels per class, both the random-feature and sparse representations improve the train and test accuracy of the linear classifier over raw pixels. With fewer than 10 labels per class all 3 representations perform comparably in train accuracy since they all perfectly fit the training set. This time, the random-feature and pixel representations perform similarly and the sparse representations perform worse. This reduced performance is not substantial, but it is robust across random seeds.

This experiment suggests a pessimistic explanation for the seemingly encouraging observation that the sparse representations improve generalization performance of a linear classifier. This experiment suggests the sparse representations are merely more linearly separable than pixels and the linear classifier is able to overcome the instabilities introduced by the sparse coding representation if it has enough labels. When you don't have enough labels, the linear classifier is outperformed by the raw pixels, which we argue is because the representations are so sensitive to distortions. The random-feature representations similarly improve the linear separability but they never hurt the linear classifier. And for the nearest neighbor classifier, the sparse representations are always worse than for pixels and random-feature representations. The Appendix shows that the above findings regarding classification are robust to the specific choice of the sparsity hyperparameter λ.

7 Discussion

A priori, there is no reason to expect that optimizing for reconstruction error will produce sparse representations that help with the downstream task of object recognition. We have argued that by some measures sparse coding produces representations which are less suited for subsequent recognition than the original pixel representations. We argued empirically that the representations can be particularly sensitive to image distortions that do not change the object class.

A linear classifier trained on top of sparse codes has been shown to exhibit more "adversarial robustness" than a backprop-trained feedforward net (Paiton et al., 2020). This robustness seems related to our work in two ways. First, adversarial perturbations of the feedforward net resemble noise, and sparse codes are robust to noise (Fig. 1). Second, the robustness is claimed for the output of the linear classifier, not the sparse code itself.

We also train a linear classifier on top of sparse codes, and its output can be robust to distortions if trained with many class labels. However we observe worse than pixel performance with few labels, which we argue is an outcome of the distortion sensitivity of the underlying sparse code.

This sensitivity arises from nonorthogonality of the learned dictionary elements. A possible remedy is to add a term to the objective function that encourages the dictionary elements to be orthogonal (Sulam et al., 2020), though this may be less effective as the dictionary size grows large.

It is interesting to contrast sparse coding with contemporary self-supervised learning techniques (Dosovitskiy et al., 2014; Chen et al., 2020; He et al., 2020; Caron et al., 2020; Zbontar et al., 2021). Networks are trained to yield similar representations for two distorted versions of the same image. Sparse codes, in contrast, are sensitive to image distortions.

Beyond sparse coding, there are other generative models such as VAEs (Kingma and Welling, 2014), GANs (Goodfellow et al., 2014), and autoregressive models (Oord et al., 2016). These models generate beautifully, but have not seen the same level of success for object recognition as supervised training with lots of labels or contrastive pretraining. These generative models are more complex and harder to understand than sparse coding but we can speculate that perhaps these are other examples where reconstructing the original image is simply not an optimal pretraining task. This speculation might be testable via experiments similar to those in Figures 1 and 3.

Acknowledgements

We would like to thank L. And J. Jackel, Lawrence Saul, and Runzhe Yang for their helpful insights and discussions. We would also like to Dylan Paiton and Bruno Olshausen for their feedback which helped to both make the paper's claims more precise and the experimental analysis more rigorous.

This research was supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior/ Interior Business Center (DoI/IBC) contract number D16PC0005, NIH/NIMH RF1MH117815, RF1MH123400. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoI/IBC, or the U.S. Government.

Appendix

7.1 Learned features for various λ

We train networks with $\lambda \in \{0.03, 0.10, 0.30, 0.90, 3.0\}$, keeping the number of features the same in each case ($n = 784$). We use the same alternating gradient-based optimization procedure described in Section 3.2. The only difference is we use more iterations for smaller λ and fewer iterations for higher λ. Figure 5 shows the learned dictionaries and the fractions of active neurons (over the training set) for the exact representations derived from each dictionary.

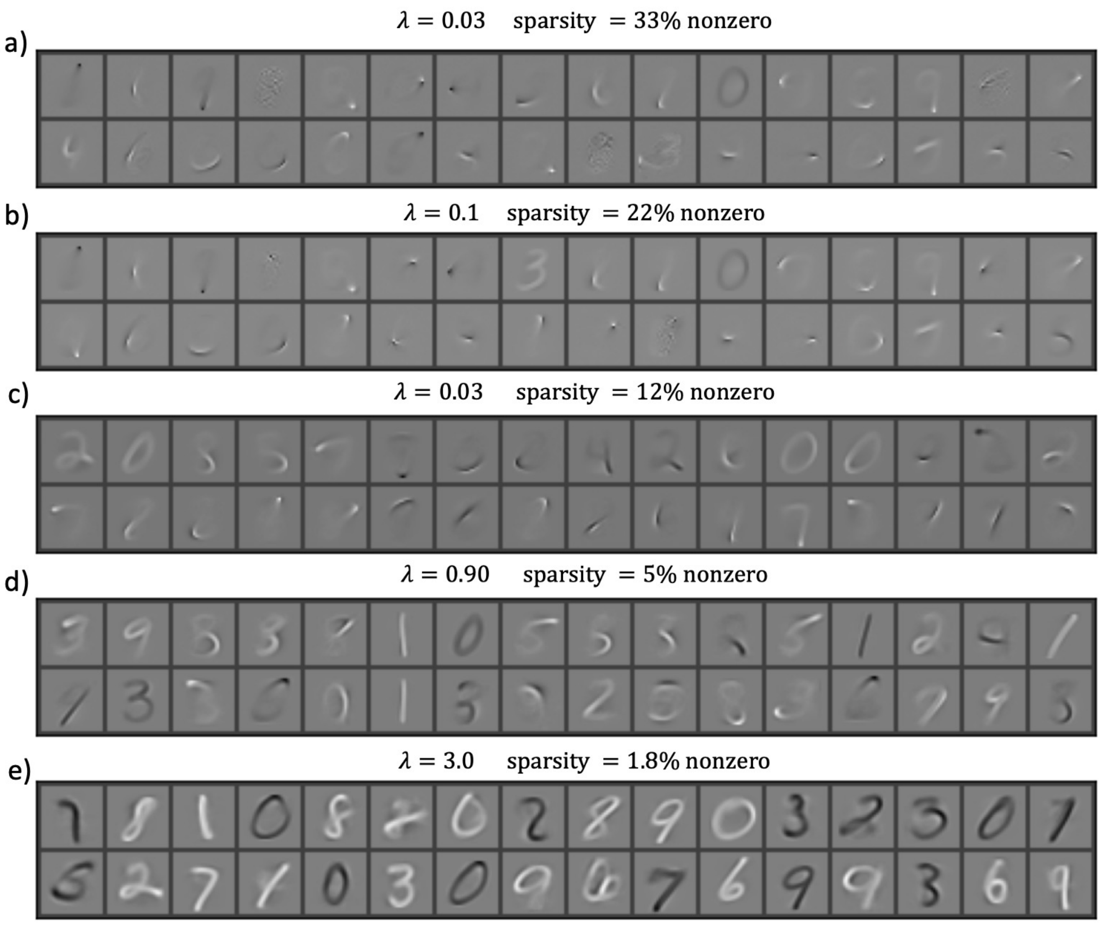
a)
$\lambda = 0.03$ sparsity = 33% nonzero
b)
$\lambda = 0.1$ sparsity = 22% nonzero
c)
$\lambda = 0.03$ sparsity = 12% nonzero
d)
$\lambda = 0.90$ sparsity = 5% nonzero
e)
$\lambda = 3.0$ sparsity = 1.8% nonzero

Figure 5: Elements of the dictionary learned for $\lambda \in \{0.03, 0.10, 0.30, 0.90, 3.0\}$. As λ increases, the fraction of nonzero neurons decreases, and the dictionary elements move from resembling small spots and strokes ($\lambda = 0.0$ up to whole digits $\lambda = 3.0$)

7.2 Computing the representations for every image

At the end of the training process described in Section 3.2, we freeze the dictionary and continue to update the representations so we nearly have the exact unique representations for every image. As discussed in the main text, uniqueness of representations is guaranteed because the learned dictionary elements turn out to be unique (the largest cosine similarity between any pair is 0.9).

These representations are generated by running enough ISTA steps until Eq. (2) is satisfied to high numerical precision. For each image, we require:

$$\begin{gathered} \text{if } r_i = 0 \text{ then } |\mathbf{d}_i \cdot \mathbf{x} - \sum_j \mathbf{d}_i \cdot \mathbf{d}_j r_j| - \lambda < 10^{-5} \\ \left\|\mathbf{r}_+ - (\mathbf{D}_+^\top \mathbf{D}_+)^{-1}(\mathbf{D}_+^\top \mathbf{x} - \lambda \mathbf{s}_+)\right\| / \|\mathbf{r}_+\| < 10^{-4} \end{gathered} \tag{9}$$

where $\mathbf{r}_+$ is the active representation for a given image and $\mathbf{s}_+$ contains the signs of these elements $(s_+)_i = \text{sign}((r_+)_i)$.

Finding the representations which achieve these conditions is time consuming. We start with single precision ISTA updates, and switch to double precision if the energy fails to decrease. The learning rate parameter is adjusted automatically as described in Section 3.2. We check the conditions Eq. 9 for each image every 10,000 iterations. Aapproximately 200,000 ISTA updates were required for convergence. In total it took approximately 12 GPU hours to compute all the representations for $\lambda = \{0.03, 0.10, 0.30, 0.90, 3.0\}$.

7.3 Sensitivity histograms for various λ

Figure 6 shows the sensitivity of the sparse codes to swaps, distortions, and noise. This histogram was shown for $\lambda = 0.3$ in Figure 1c. We now show the histograms for $\lambda \in \{0.03, 0.10, 0.30, 0.90, 3.0\}$.

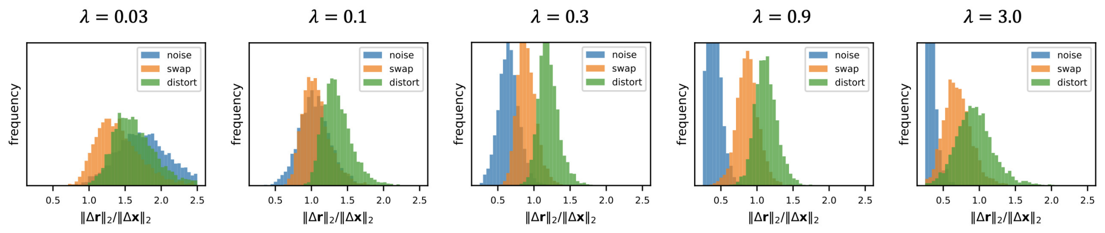

Figure 6: Sensitivity of the sparse representations to noise, swaps, distortion perturbations (examples are shown in Fig. 1). For every λ we test, the sparse representations are more sensitive to distortions than to swaps. This difference seems most pronounced for the intermediate values of $\lambda = 0.1, 0.3, 0.9$. The noise sensitivity appears to be most strongly impacted by the sparsity level, and the representations are less noise sensitive as the sparsity level increases.

7.4 Supervised evaluation for various λ

Figure 7 shows the supervised evaluation metrics for various λ.

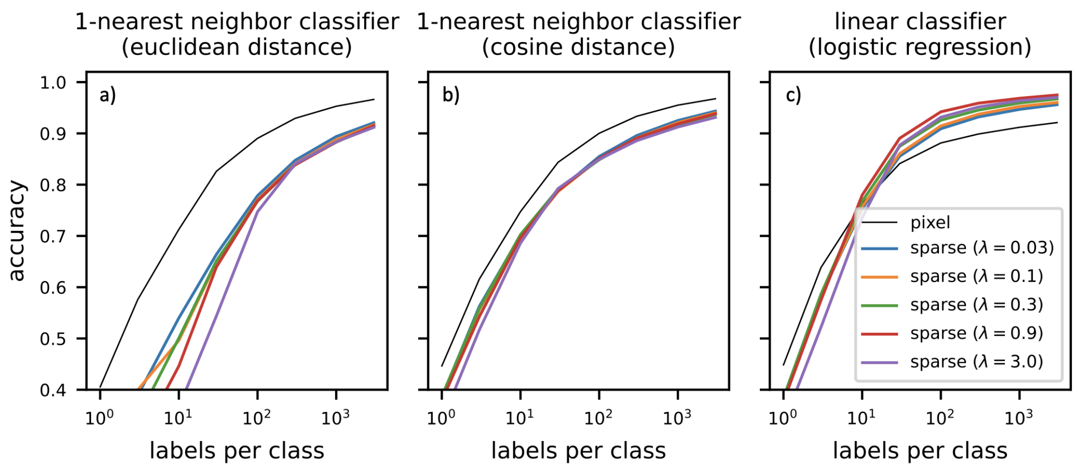

Figure 7: Supervised evaluation of the representations with various λ. The behavior of the representations is remarkably similar, given the difference in sparsity and learned features (Fig. 5) between the various λ. We do observe that an increase in sparsity is somewhat worse for both the nearest neighbor classifiers and the linear classifier when the number of training labels is low.

References

Beck, A. and Teboulle, M. (2009). A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sciences*, 2:183–202.

Cadieu, C. F. and Olshausen, B. A. (2012). Learning intermediate-level representations of form and motion from natural movies. *Neural computation*, 24(4):827–866.

Candes, E. J. (2008). The restricted isometry property and its implications for compressed sensing. *Comptes rendus mathematique*, 346(9-10):589–592.

Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., and Joulin, A. (2020). Unsupervised learning of visual features by contrasting cluster assignments. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 9912–9924. Curran Associates, Inc.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. (2020). A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Chen, Y., Paiton, D., and Olshausen, B. (2018). The sparse manifold transform. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Deng, L. (2012). The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, 29(6):141–142.

DiCarlo, J., Zoccolan, D., and Rust, N. (2012). How does the brain solve visual object recognition? *Neuron*, 73:415–434.

Dosovitskiy, A., Springenberg, J. T., Riedmiller, M., and Brox, T. (2014). Discriminative unsupervised feature learning with convolutional neural networks. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Elad, M. and Aharon, M. (2006). Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 15(12):3736–3745.

Fukushima, K. (1980). Neocognitron: a self organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4):193—202.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P. H., Buchatskaya, E., Doersch, C., Pires, B. A., Guo, Z. D., Azar, M. G., et al. (2020). Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*.

Guillamet, D., Schiele, B., and Vitrià, J. (2002). Analyzing non-negative matrix factorization for image classification.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. (2020). Momentum contrast for unsupervised visual representation learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9726–9735.

Hu, X., Zhang, J., Li, J., and Zhang, B. (2014). Sparsity-regularized hmax for visual recognition. *PloS one*, 9(1):e81813.

Jarrett, K., Kavukcuoglu, K., Ranzato, M., and LeCun, Y. (2009). What is the best multi-stage architecture for object recognition? In *2009 IEEE 12th International Conference on Computer Vision*, pages 2146–2153.

Kingma, D. P., Mohamed, S., Jimenez Rezende, D., and Welling, M. (2014). Semi-supervised learning with deep generative models. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Kingma, D. P. and Welling, M. (2014). Auto-encoding variational bayes. In Bengio, Y. and LeCun, Y., editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Olshausen, B. and Field, D. (1996). Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381:607–9.

Olshausen, B. A. and Field, D. J. (1997). Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision Research*, 37(23):3311 – 3325.

Oord, A. v. d., Kalchbrenner, N., Vinyals, O., Espeholt, L., Graves, A., and Kavukcuoglu, K. (2016). Conditional image generation with pixelcnn decoders. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 4797–4805, Red Hook, NY, USA. Curran Associates Inc.

Paiton, D., Frye, C., Lundquist, S., Bowen, J., Zarcone, R., and Olshausen, B. (2020). Selectivity and robustness of sparse coding networks. *Journal of Vision*, 20:10.

Radford, A., Metz, L., and Chintala, S. (2016). Unsupervised representation learning with deep convolutional generative adversarial networks. *CoRR*, abs/1511.06434.

Raina, R., Battle, A., Lee, H., Packer, B., and Ng, A. Y. (2007). Self-taught learning: Transfer learning from unlabeled data. New York, NY, USA. Association for Computing Machinery.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In Navab, N., Hornegger, J., Wells, W. M., and Frangi, A. F., editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham. Springer International Publishing.

Sulam, J., Muthukumar, R., and Arora, R. (2020). Adversarial robustness of supervised sparse coding. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 2110–2121. Curran Associates, Inc.

Tibshirani, R. J. (2013). The lasso problem and uniqueness. *Electronic Journal of Statistics*, 7(none):1456 – 1490.

Turk, M. and Pentland, A. (1991). Face recognition using eigenfaces. In *Proceedings. 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 586–591.

van Tulder, G. (2021). elasticdeform: Elastic deformations for N-dimensional images.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.-A., and Bottou, L. (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(12).

Yang, J., Yu, K., Gong, Y., and Huang, T. (2009). Linear spatial pyramid matching using sparse coding for image classification. In *2009 IEEE Conference on computer vision and pattern recognition*, pages 1794–1801. IEEE.

Zbontar, J., Jing, L., Misra, I., LeCun, Y., and Deny, S. (2021). Barlow twins: Self-supervised learning via redundancy reduction. *CoRR*, abs/2103.03230.