A Neuromorphic Sparse Coding Defense to Adversarial Images

Edward Kim
Villanova University
Villanova, Pennsylvania
edward.kim@villanova.edu

Jessica Yarnall
Villanova University
Villanova, Pennsylvania
jyarnal1@villanova.edu

Priya Shah
Villanova University
Villanova, Pennsylvania
pshah26@villanova.edu

Garrett T. Kenyon
Los Alamos National Laboratory
Los Alamos, New Mexico
gkenyon@lanl.gov

ABSTRACT

Adversarial images are a class of images that have been slightly altered by very specific noise to change the way a deep learning neural network classifies the image. In many cases, this particular noise is imperceptible to the human vision system and thus presents a vulnerability of significant concern to the machine learning and artificial intelligence community. Research towards mitigating this type of attack has taken many forms, one of which is to filter or post process the image before classifying the image with a deep neural network. Techniques such as smoothing, filtering, and compression have been used with varying levels of success.

In our work, we explored the use of a neuromorphic software and hardware approach as a protection against adversarial image attack. The algorithm governing our neuromorphic approach is based upon sparse coding. Our sparse coding approach is solved using a dynamic system of equations that models biological low level vision. Our quantitative and qualitative results show that a sparse coding reconstruction is remarkably invariant to changes in sparsity and reconstruction error with respect to classification accuracy. Furthermore, our approach is able to maintain low reconstruction errors without sacrificing classification performance.

CCS CONCEPTS

• **Computing methodologies** → **Computer vision**; **Machine learning approaches**; • **Hardware** → **Neural systems**.

KEYWORDS

neuromorphic computing, sparse coding, adversarial image attack

ACM Reference Format:
Edward Kim, Jessica Yarnall, Priya Shah, and Garrett T. Kenyon. 2019. A Neuromorphic Sparse Coding Defense to Adversarial Images. In *International Conference on Neuromorphic Systems (ICONS '19), July 23–25, 2019, Knoxville, TN, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3354265.3354277

ICONS '19, July 23–25, 2019, Knoxville, TN, USA

ACM ISBN 978-1-4503-7680-8/19/07…$15.00
https://doi.org/10.1145/3354265.3354277

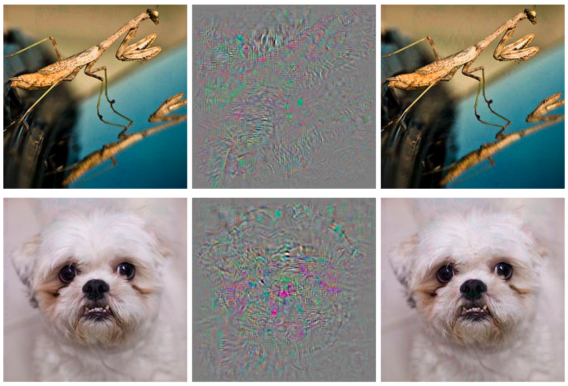


Figure 1: Examples of images next to the adversarial perturbation noise, and the resulting adversarial image (original image plus noise) from Szegedy et al. [26]. The original images are correctly classified; however, with the addition of noise, the adversarial image is classified as an "ostrich". As seen in the example, the adversarial image is nearly identical to the original image with no detectable semantic difference.

1 INTRODUCTION

Deep learning has revolutionized virtually all computer vision and machine learning tasks. Convolutional neural networks (CNNs) are now the standard in image classification, detection, and segmentation. In some settings, deep learning artificial neural networks can perform at super human levels [10, 29]. But even though these networks perform specific tasks very well, they have a surprising and disturbing property that they are susceptible to attack using adversarial images [13, 20, 25, 26]. An example of a specific type of adversarial image can be seen in Figure 1. Adversarial images are a class of images that have been slightly altered by very specific noise. This noise has the effect of changing the way a deep learning neural network classifies the image. Although general adversarial noise alone is not a particularly critical issue, i.e. one can imagine continually adding noise to an image until it becomes unrecognizable, the main problem is that, in many cases, this particular noise is imperceptible to the human vision system. Thus, it is clear that a deep learning classification system is learning some subset of features that are principally different from what humans learn and

use for object recognition, and these features can be easily manipulated by high frequency noise. As deep learning becomes more and more pervasive in autonomous vehicles, voice recognition systems, biometric security, etc., this vulnerability is of significant concern to the machine learning and artificial intelligence community.

In our work, we explored the possibilities of creating a neuromorphic software and hardware defense to adversarial image attacks. We define the term *neuromorphic* as technology that implements models of neural systems. These neuromorphic technologies mimic particular forms and functions of mammalian brains. If we are able to more closely model the process of human perception, it is conceivable that such a system would not be susceptible to small, noisy perturbations and would mitigate the effects of this attack. Towards this goal, we perform a systematic evaluation of a neurmorphic software and hardware approach based upon sparse coding. We compare our approach to other similar strategies to defending against adversarial image attack and present both qualitative and quantitative results that highlight the benefits of our neuromorphic approach.

2 BACKGROUND

2.1 Background in Adversarial Image Attack

Adversarial image attacks can be performed in different ways and can be categorized by methodology and information. One category of attack are *white box attacks* where the attack algorithm has access to the classification model parameters and can use this information to perturb the input image pixels. In many of these cases, the gradient of the loss function can be observed for the attack process. Examples of attacks that use gradient information include Fast Gradient Sign Method (FGSM) [8], Iterative-FGSM [28], and Deepfool [18].

Another category of attack are the *black box attacks* where the attack algorithm does not have access to the classifier. These attacks must find perturbations with nearly zero knowledge of the underlying classifier. An example of this type of attack was used in the one pixel attack method [24]. This black box attack uses differential evolution, where a population of candidate solutions generate offspring which compete with the rest of the population according to their fitness. New offspring are generated by combining (mutating) individuals in the population, and replacing worse-performing individuals with better candidates.

Just as there are multiple ways to attack a classifier, there are multiple ways to defend against attack. One approach, adversarial training, incorporates the attack images into the training set. This introduces the attack pattern to the learning algorithm and attempts to make the classifier more robust. Alternatively, a different kind of defense involves transforming the input signal. The basic idea of input transformation defenses is to alter the input image to eliminate the adversarial noise. Some examples of input transformation include spatial smoothing or blurring methods [30], compression techniques [4], and total variation minimization [9]. We note that this list is not exhaustive and there are many more attacks and defenses. We presented several approaches in order to provide some context into the field of adversarial image attack. For our work, we will use some of the more common methods of input transformation as comparisons to our neuromorphic approach.

2.2 Background in Neuromorphic Software and Sparse Coding

The principle of sparsity is central to how we perceive and understand the world in which we live. The brain's neural representations are sparse and highly recurrent with many feedback connections. Strong evidence demonstrates that the neural code is both explicit and sparse [6] where neurons fire selectively to specific stimuli. Olshausen and Field [19] have shown that sparsity is a desirable property as our natural environment can be described by a small number of structural primitives. Sparse codes have a high representational capacity in associative memory, far surpassing the number of input-output pairs that can be stored by a more dense code [2], and biologically, the sparsity of neural codes are more metabolically efficient and reduce the cost of code transmission [1]. The advantages of sparsity in a neural network are supported by the research and support the interpretability of sparse codes. These include: information disentangling, efficient variable-size representation, and evidence that sparse representations are more linearly separable [7].

The algorithm governing our neuromorphic approach is based upon sparse coding. Mathematically, sparse coding is a reconstruction minimization problem which can be defined as follows. In the sparse coding model, we have some input variable $x^{(n)}$ from which we are attempting to find a latent representation $a^{(n)}$ (we refer to as "activations") such that $a^{(n)}$ is sparse, e.g. contains many zeros, and we can reconstruct the original input, $x^{(n)}$ with high fidelity. A single layer of sparse coding can be defined as,

$$\min_{\Phi} \sum_{n=1}^{N} \min_{a^{(n)}} \frac{1}{2} \|x^{(n)} - \Phi a^{(n)}\|_2^2 + \lambda \|a^{(n)}\|_1 \tag{1}$$

Where Φ is the overcomplete dictionary, and $\Phi a^{(n)} = \hat{x}^{(n)}$, or the reconstruction of $x^{(n)}$. The λ term controls the sparsity penalty, balancing the reconstruction versus sparsity term. N is the total training set, where n is one element of training. Φ represents a dictionary composed of small kernels that share features across the input signal.

Recent work has started looking at the adversarial problem with components of sparse coding as a potential solution. Dictionary denoising on image samples improves classification [17] and sparsity has shown some robustness to adversarial attacks on linear classifiers [16]. Additionally, the use of sparse coding has amazing image denoising properties, which inherently provides some robustness to pixel perturbations [17]. Sparse coding is also unsupervised, which provides support against gradient based attacks on supervised neural networks. Deep learning networks tend to learn surface statistical regularities [13] in the data, whereas, sparse coding learns feature representations robust to variances in the data.

2.3 Background in Neuromorphic Hardware and Spiking Neural Networks

If we consider that the language of the brain is, "spikes", then the next frontier of neuromorphic computation is spiking hardware. There have been a small number of major efforts in creating neuromorphic hardware systems such as IBM's True North (2014), SpiNNaker (2009), and Intel's Loihi (2018). Intel's Loihi is the latest

chip developed and was released for research in 2018. Loihi is a manycore spiking neural network, where a single chip contains 128 neuromorphic cores, 128k neurons, 128M synapses.

In our work, we utilized Intel's Loihi chip and thus describe this hardware and dynamics of this system in more detail. The chip is created in a 14nm process with scalable on-chip learning capabilities. The chip is fully asynchronous, fully digital, and deterministic. At any given time, the neurons may send out a spike to its neighbors by use of virtual synapses. Neurons have two internal state variables - a synaptic response current, a weighted sum of the input spikes and a constant bias, and membrane potential that leaks over time and will send out a spike when the potential crosses the firing threshold. Once enough spikes accumulate and exceed some threshold, a spike message is generated and sent to other connected neurons. In the following section, we will describe how the sparse coding problem can be transformed into a dynamic system.

3 METHODOLOGY

3.1 Sparse Coding as a Dynamic System

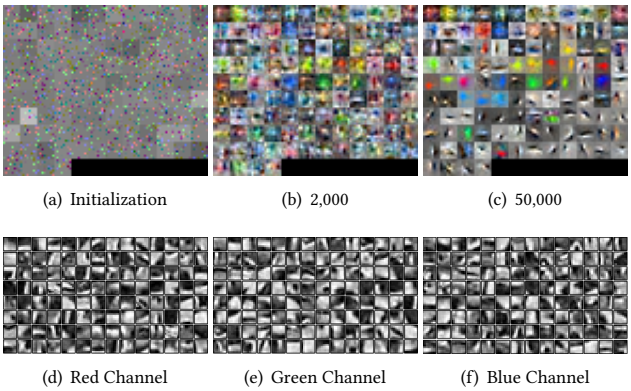
We use the Locally Competitive Algorithm (LCA) [21] to minimize the mean-squared error (MSE) with sparsity cost function as described in Equation 1. The LCA algorithm is a biologically informed sparse solver that uses principles of *thresholding* and *local competition* between neurons. The LCA model is governed by dynamics that evolve the neuron's internal state when presented with some input image. The internal state, i.e. "membrane potential", charges up like a leaky integrator and when it exceeds a certain threshold, will activate that neuron. This activation will then send out inhibitory responses to units within the layer to prevent them from firing. The input potential to the state is proportional to how well the image matches the neuron's dictionary element, while the inhibitory strength is proportional to the activation and the similarity of the current neuron and competing neuron's convolutional patches, forcing the neurons to be decorrelated.

We utilize a modified LCA that uses a deconvolutional approach [22] called OpenPV [1] . OpenPV is an open source, object oriented neural simulation toolbox optimized for high-performance multicore computer architectures. Equation 1 is transformed into an energy function, whereby minimizing the energy with respect to Φ generates a sparse code that faithfully reconstructs the image with few feature vectors. The feature vectors, or dictionary Φ, is learned using a Hebbian weight update rule using a residual (error) layer. Some examples of learned dictionary elements can be seen in Figure 2.

3.2 Dataset

In our work, we worked on the CIFAR-10 dataset. This dataset consists of 32x32 color images, 10 classes, and 6,000 images per class. For our experiments, we randomly select 1,000 test images that have not been seen during training. There are several reasons why we decided to evaluate on CIFAR. First, the images are relatively small. This was a requirement as we are working with emerging technology, where the size and scale of an ImageNet-like dataset would be computationally prohibitive at this time. Second, the one-pixel black box attack described in Su et al. [24] has been extensively tested on CIFAR-10. This provides a replicable benchmark for our research. Other gradient based methods on CIFAR frequently create high distortion, unrecognizable images, see Figure 3, and therefore not evaluated in this work.

[1]https://github.com/PetaVision/OpenPV



(a) Initialization (b) 2,000 (c) 50,000

(d) Red Channel (e) Green Channel (f) Blue Channel

Figure 2: Examples of 112 dictionary elements used for sparse coding CIFAR images. (a-c) show the progression of learning an RGB dictionary from (a) a random initialization to the (c) resulting dictionary after viewing 50,000 CIFAR training images using OpenPV. The figures (d-f) show separated red, green, and blue dictionary channels learned using a patch-based least angle regression method.

(a) (b) (c)

Figure 3: High distortion effects are visible in many gradient based attacks such as FGSM, iterative FGSM, and Deepfool on CIFAR 32x32 images. In (a)-(c) we illustrate how the noise effects CIFAR sized images using a Deepfool attack.

3.3 Neural Network Models

We evaluated our method on the following neural network architectures.

ResNet [11] - ResNet is a residual learning framework that won 1st place in the ILSVRC 2015 classification task. ResNet uses residual functions to learn and has shown this is an effective way of training networks that are substantially deeper than previous models. The ResNet instance that we use has a total of 32 layers and achieves a 92.3% accuracy on the CIFAR-10 dataset. After being attacked, the baseline accuracy of ResNet is **67.5% when perturbing one pixel**.

The baseline accuracy drops to **27.6% when using three pixels**. The use of five pixels has a nearly equivalent baseline accuracy around 27%.

LeNet [15] - LeNet-5 was one of the earliest developed deep convolutional neural networks. The instance of LeNet-5 that we use is a five layer network consisting of two convolutional layers and three densely connected layers. LeNet-5 achieves a 74.9% accuracy on the CIFAR-10 dataset. After being attacked, we observed the baseline accuracy of LeNet-5 is **41.0% when perturbing one pixel**.

DenseNet [12] - DenseNet refers to a convolutional neural network that is densely connected, i.e. every layer is connected to every other layer in a feed-foward fashion. DenseNet was able to achieve the highest test accuracy of the networks shown on CIFAR-10 with a test accuracy of 94.7%. After being attacked, the baseline accuracy of DenseNet dropped to **74.3% when perturbing one pixel.**.

3.4 Defense Methods

For the defense against adversarial image attack, we compared against several common input transformation defense methods and some variants of sparse coding.

Spatial Sampling [30] - One of the most basic approaches to removing adversarial noise is through the use of image filters. A common filter is a gaussian smoothing operation where the blurring radius is parameter that must be chosen.

JPG compression [4] - The process of removing adversarial noise can be viewed as a lossy image compression problem where the noise is idealy removed in the encoded image. Some have used JPG compression as a technique to quantize away the imperceptible elements within an image. The quality of compression is a parameter that should be selected. For example in [4] a quality of 75 is chosen.

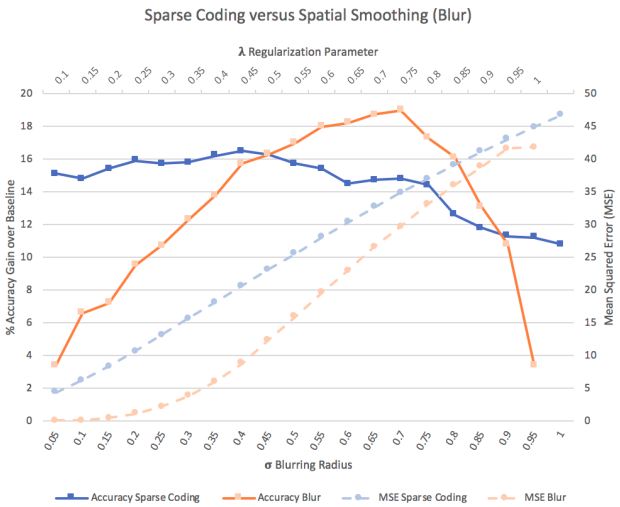
LARS Lasso Sparse Coding [5] - The LARS Lasso method is a least angle regression method that provides an estimate of which basis vectors (dictionary elements) should be used as well as their coffieicients. The Lasso problem introduces the L1 penalty as a convex relaxation to the L0 norm which is used to induce sparsity within the solution. For the LARS methods, we experimented with the number of dictionary elements: 112, 224, and 336.

OpenPV LCA Sparse Coding [22] -The OpenPV LCA sparse coding solver is a neurmorphic software implementation that utilizes concepts of dynamics, leaky integrate and fire neurons, and lateral inhibition in solving the sparse coding problem. The sparsity of the solution can be controlled by the regularization parameter, λ.

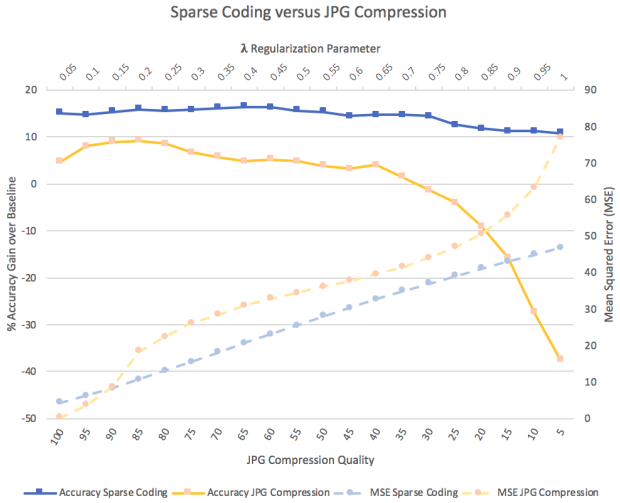
Loihi LCA Sparse Coding [27] - This method utilizes both a neuromorphic software and hardware approach to solving the sparse coding optimization problem. The Intel Loihi chip is a spiking neural network and the implementation by Tang et al. [27] describes how the rate-encoded LCA algorithm can be converted and computed using a spiking implementation.

3.5 Evaluation Metrics

The metrics that we use to evaluate our results measure both the quality of the input after transformation, as well as the accuracy of transformed images on different neural networks. Specifically, we measure accuracy of classification over the baseline accuracy (ACC),



(a) Effect of Parameterization on Sparse Coding λ and Blur radius σ on accuracy over baseline (ResNet one pixel attack) and mean squared error to the input image.



(b) Effect of Parameterization on Sparse Coding λ and JPG Quality on accuracy over baseline (ResNet one pixel attack) and mean squared error to the input image.

Figure 4: Comparison between sparse coding and (a) spatial smoothing and (b) jpg compression. The solid color lines are plotted with the primary axis (left Y axis) and represent the accuracy increase over the baseline. The dotted, and lighter shaded lines are plotted with the secondary axis (right Y axis) and represent the mean squared error of the transformed image and input image.

mean squared error (MSE), peak signal-to-noise ratio (PSNR), and distortion to accuracy ratio (DistR). For the random test images, we ensure that every one of the 1000 test images that we attack were originally correctly classified by the model.

For the metric calculations, we define the MSE as the measure of the squared sum of square differences between the attack image

and the reconstruction image, divided by the total number of pixels. The PSNR measure uses the ratio of error to the maximum intensity of the image. We calculate PSNR by the following equation,

$$PSNR = 10 \cdot log_{10} \frac{MAX_I^2}{MSE} \tag{2}$$

PSNR is most commonly used to measure the quality of reconstruction of lossy compression codecs (e.g., for image compression). The signal in this case is the original data, and the noise is the error introduced by compression. Typical values for the PSNR in lossy image and video compression are between 30 and 50 dB (the higher the number the better), provided the bit depth is 8 bits.

We will define the distortion to accuracy ratio as the following equation,

$$DistR = \frac{\frac{1}{n}\sum_{i=1}^{n}(I_i - \hat{I}_i)^2}{ACC} = \frac{MSE}{ACC} \tag{3}$$

At a high level this describes how much distortion is necessary to the original image (measured in mean squared error) in order to gain a percentage increase in overall classification accuracy.

4 RESULTS

4.1 The Effect of Parameterization

All of the input transformation defenses that we tested required the selection of some parameter. In the spatial sampling (blurring) defense, we were required to select the blurring radius. In the JPG compression defense, we needed to select the quality of the encoding. For the sparse coding solutions, we need to select the weight of the L1 penalty in the OpenPV implementation, which controls the sparsity of the solution. In an experimental setting, we are able to show the effect of the parameter selection; however, in the real world, one would not have that luxury. A robust defense should be as invariant as possible to parameterization. In Figure 4 (a)(b) we plot the accuracy (on ResNet using a one-pixel attack) and mean squared error metrics of common defenses versus sparse coding.

As can be seen in Figure 4 (a)(b), the accuracy of both the Blurring and JPG defenses are highly dependent on their respective parameters. We controlled for the parameter space by using the MSE measurement, such that there are relatively similar profiles for MSE in the compared methods. In contrast, *sparse coding accuracy is relatively invariant to the parameter space*, even as the MSE continues to rise, and the sparsity of the solution varies significantly. At λ = 0.05, the sparsity of the solution is 24.65% while the sparsity at λ = 1.0 is nearly an order of magnitude lower at 2.83%.

4.2 Qualitative and Quantitative Results

When observing the absolute gain in accuracy in Figure 4 (a), blurring with a radius of 0.75 has the best overall accuracy gain over the baseline. However, this accuracy gain comes at significant cost to MSE. Ideally, a robust defense would result in high accuracy and minimal error (distortion) to the original input. We capture this ratio in the DistR metric show in Tables 1, 2, 3, 4. These tables quantitatively list the accuracy over the baseline, peak signal to noise ratio, mean squared error, and distortion to accuracy ratio, respectively. Ideally, the ACC and PSNR should be as high as possible, and the MSE and DistR should be as low as possible. As seen in the results, some version of sparse coding consistently has the best (lowest) DistRs and maintains a high overall accuracy.

Qualitative results of a one pixel attack are shown in Figure 5 and a three pixel attack in Figure 6. The visual effects of the input transformations can be seen at different parameterizations of the defense algorithm.

5 DISCUSSION

5.1 Neuromorphic Hardware

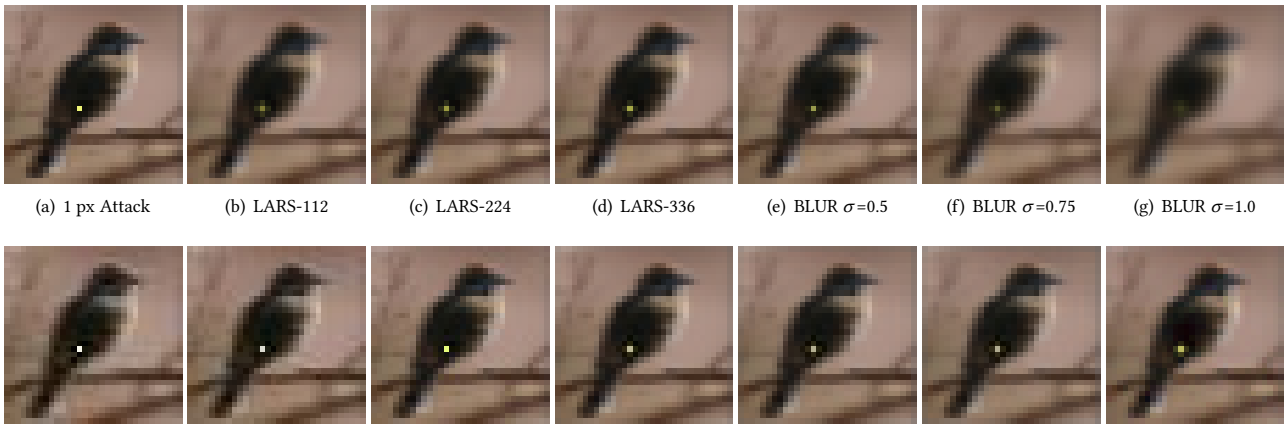
Although there are benefits of neuromorphic software approaches to sparse coding e.g. generative model, better features, better interpretability, optimizes an overcomplete basis, etc., the main drawback is the computational cost. Thus, we are especially excited with developments the Intel Loihi in spiking neural network hardware capabilities. Currently, the implementation of a spiking LCA is running with 112 dictionary elements, a patch size of 8, and stride of 4. Exponential memory and connectivity costs prohibit a smaller stride which would help the fidelity of reconstruction.

Although a smaller stride is not possible, we are able to solve larger input images with the same dictionary and patch sizes. Thus, to emulate a smaller stride (from 4 to 2), we double the size of the input image, sparse code with the default stride of four, then resize the image back to the original size. This sizing trick has a dramatic effect on the reconstruction quality of the result, often resulting in MSE improvements of 2.5x-3x.

Qualitative and quantitative results demosntrate that the spiking LCA hardware implementation has competitive performance, and can further improve as the hardware develops. Davies et al. [3] has shown significant improvements in solving the sparse coding problem both in energy consumption and speed. As of December 2018, the implementation of LCA on Loihi is 10-50x faster for a non-convolutional sparse coding approach, 100-1000x faster in a convolutional algorithm, and 10,000x-100,000x times more energy efficient. The numbers range based upon a range in the number of unknowns required to solve on an Intel Core i7 - 4790 3.6Ghz w/32 GB of ram CPU implemented FISTA (fast iterative shrinking-thresholding algorithm).

5.2 Neuromorphic Software

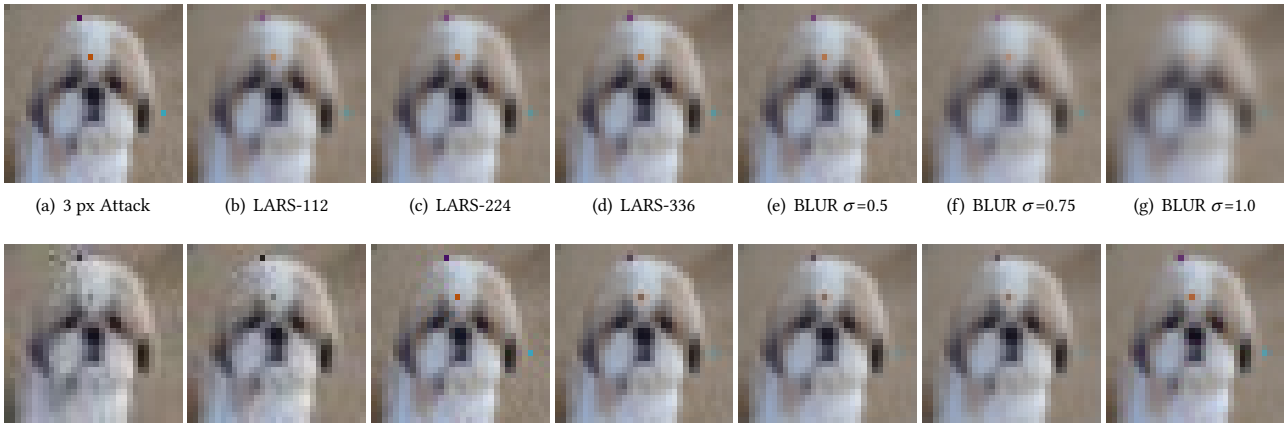
For this project, our sparse coding neuromorphic approach transforms the input and the transformed input is used for classification in a deep neural network. The software approach was coded in the OpenPV framework and the sparse coding optimization was solved by simulating neuronal membrane potentials, inhibition, and dynamics. Our results show that a single layer of sparse coding is a promising direction in protecting against adversarial attack. However, we believe that true robustness in classification will not be achieved by an approach that uses the features extracted from a deep neural network. As stated in our introduction, a deep learning classification system is learning some subset of features that are different from what humans learn and use for object recognition, and these features can be easily manipulated by high frequency noise. Ideally, we do not go back to the original image, but instead work directly with the sparse features to perform inference and image understanding.



(a) 1 px Attack (b) LARS-112 (c) LARS-224 (d) LARS-336 (e) BLUR σ=0.5 (f) BLUR σ=0.75 (g) BLUR σ=1.0

(h) JPG comp=75 (i) JPG comp=85 (j) JPG comp=95 (k) OpenPV λ=0.05 (l) OpenPV λ=0.15 (m) OpenPV λ=0.25 (n) Loihi-112

Figure 5: (a) One pixel attack and (b-n) qualitative results of the resulting gaussian blur filter, compression algorithm, or sparse reconstruction. Details of the reconstruction best viewed with antialiasing turned off.



(a) 3 px Attack (b) LARS-112 (c) LARS-224 (d) LARS-336 (e) BLUR σ=0.5 (f) BLUR σ=0.75 (g) BLUR σ=1.0

(h) JPG comp=75 (i) JPG comp=85 (j) JPG comp=95 (k) OpenPV λ=0.05 (l) OpenPV λ=0.15 (m) OpenPV λ=0.25 (n) Loihi-112

Figure 6: (a) Three pixel attack and (b-n) qualitative results of the resulting gaussian blur filter, compression algorithm, or sparse reconstruction. Details of the reconstruction best viewed with antialiasing turned off.

As an example, we have shown that deep sparse coding [14] is immune to adversarial transfer attacks [23] that work across different deep learning networks. A deep sparse architecture was invariant to small perturbations in the input image. This robustness stemmed from the model learning general features corresponding to generators of the dataset as a whole, rather than highly discriminative features for distinguishing specific classes. The resulting classifiers using these features were less dependent on idiosyncrasies that might be more easily exploited. We also note that our deep sparse coding models utilize fixed point attractor dynamics with top-down feedback, making it more difficult to find small changes to the input that drive the resulting representations out of the correct attractor basin.

Although no model is immune to all adversarial noise, we were able to demonstrate that the deep sparse coding model relies on features that are *different than those relied upon by other deep learning models*. This immunity to adversarial examples has significant implications for the field of transferrable adversarial machine learning. Figure 7 shows that an image generated to attack the DenseNet121 model is also an effective adversarial image on other CNNs including ResNet50, InceptionV3, VGG16 and MobileNetV2, but has no effect on our deep sparse coding model (DSC).

Method on ResNet (1px)	ACC	PSNR	MSE	DistR
LARS-112	14.6%	37.87	10.59	0.72
LARS-224	13.6%	40.40	5.92	0.43
LARS-336	13.4%	41.91	4.18	0.31
BLUR σ=0.5	16.3%	37.20	12.36	0.75
BLUR σ=0.75	**19.0%**	33.40	29.75	1.56
BLUR σ=1.0	5.0%	31.92	41.77	8.35
JPG compression=75	6.7%	33.96	26.09	3.89
JPG compression=85	9.2%	35.45	18.50	2.01
JPG compression=95	8.1%	**42.56**	**3.60**	0.44
*OpenPV-112 λ=0.05	15.1%	41.72	4.37	**0.28**
*OpenPV-112 λ=0.15	15.4%	38.91	8.34	0.54
*OpenPV-112 λ=0.25	15.7%	36.96	13.08	0.83
†Loihi-112	8.7%	37.31	12.07	1.38

Table 1: Results of various input transformation defense on a ResNet one pixel attack. ACC indicates the improvement over the baseline accuracy. The baseline accuracy of ResNet one pixel is 67.5%. (*software neuromorphic implementaiton. †hardware neuromorphic implementation.)

Method on LeNet (1px)	ACC	PSNR	MSE	DistR
LARS-112	**36.7%**	37.80	10.77	0.29
LARS-224	31.7%	40.30	6.05	0.19
LARS-336	28.6%	41.76	4.32	**0.15**
BLUR σ=0.5	32.9%	37.04	12.85	0.39
BLUR σ=0.75	34.1%	33.29	30.42	0.89
BLUR σ=1.0	20.2%	31.84	42.48	2.10
JPG compression=75	13.8%	33.96	26.06	1.88
JPG compression=85	13.8%	35.42	18.64	1.35
JPG compression=95	13.8%	**42.72**	**3.47**	0.25
*OpenPV-112 λ=0.05	14.2%	41.53	4.56	0.32
*OpenPV-112 λ=0.10	16.0%	40.02	6.47	0.40
*OpenPV-112 λ=0.15	17.6%	38.77	8.62	0.48
†Loihi-112	34.3%	32.45	36.96	1.07

Table 3: Results of various input transformation defense on a LeNet one pixel attack. ACC indicates the improvement over the baseline accuracy. The baseline accuracy of LeNet is 41.0%. (*software neuromorphic implementaiton. †hardware neuromorphic implementation.)

Method on ResNet (3px)	ACC	PSNR	MSE	DistR
LARS-112	44.8%	37.58	11.33	0.25
LARS-224	39.0%	39.94	6.58	0.16
LARS-336	35.8%	41.30	4.81	**0.13**
BLUR σ=0.5	43.3%	36.91	13.21	0.30
BLUR σ=0.75	**51.3%**	33.25	30.70	0.59
BLUR σ=1.0	44.7%	31.82	42.66	0.95
JPG compression=75	41.0%	33.76	27.34	0.66
JPG compression=85	43.1%	35.21	19.58	0.45
JPG compression=95	28.6%	**42.06**	**4.04**	0.14
*OpenPV-112 λ=0.05	38.2%	40.80	5.40	0.14
*OpenPV-112 λ=0.15	43.0%	38.42	9.34	0.21
*OpenPV-112 λ=0.25	45.2%	36.65	14.05	0.31
†Loihi-112	39.3%	36.94	13.13	0.35

Table 2: Results of various input transformation defense on a ResNet three pixel attack. ACC indicates the improvement over the baseline accuracy. The baseline accuracy of ResNet three pixel is 27.6%. (*software neuromorphic implementaiton. † hardware neuromorphic implementation.)

Method on DenseNet (1px)	ACC	PSNR	MSE	DistR
LARS-112	7.7%	37.89	10.56	1.37
LARS-224	8.3%	40.43	5.88	0.70
LARS-336	7.9%	41.93	4.16	**0.52**
BLUR σ=0.5	7.9%	37.14	12.54	1.58
BLUR σ=0.75	8.5%	33.33	30.17	3.54
BLUR σ=1.0	-9.6%	31.85	42.46	inf
JPG compression=75	1.1%	33.98	25.95	23.59
JPG compression=85	5.4%	35.38	18.82	3.4
JPG compression=95	2.8%	**42.56**	**3.60**	1.28
*OpenPV-112 λ=0.05	7.4%	41.69	4.40	0.59
*OpenPV-112 λ=0.10	7.8%	40.14	6.28	0.80
*OpenPV-112 λ=0.15	**9.4%**	38.97	8.41	0.89

Table 4: Results of various input transformation defense on a DenseNet one pixel attack. ACC indicates the improvement over the baseline accuracy. The baseline accuracy of DenseNet is 74.3%. (*software neuromorphic implementaiton. †hardware neuromorphic implementation.)

6 CONCLUSION

In conclusion, adversarial image attacks exploit a critical vulnerability in deep learning systems. In computer vision, these attacks often add imperceptible noise to an image in order to change the classification result. Previous research has been performed on defending against these attacks. One type of defense is transforming the input image through the use of filters, compression, or smoothing. However, since these attacks are often imperceptible to the human vision system, we explored the use of neuromorphic software and hardware approaches as a protection against adversarial image attack.

Our sparse coding approach is solved using a dynamic system of equations that models biological low level vision. This approach was further tested on a spiking neural network chip, Loihi. Our results show feasibility and competitiveness in accuracy of performance, with significant benefits in time to solution and energy consumption. From both the quantitative and qualitative results, we come to the following conclusions.

1. A resulting sparse coding reconstruction is remarkably robust to changes in sparsity and MSE reconstruction error with respect to classification accuracy. The fact that sparse coding reconstruction classification is relatively invariant to large

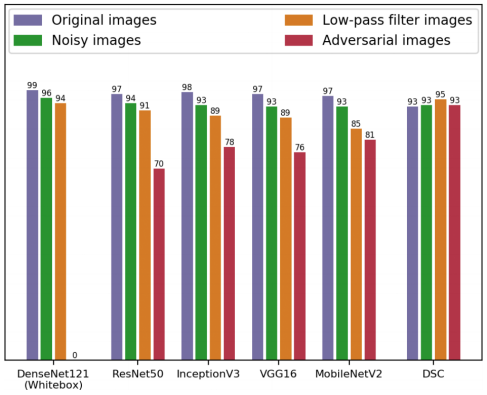


Figure 7: Accuracy scores of various deep learning CNNs. DenseNet121 is attacked leading to an accuracy of 0 on the adversarial examples. The performance of all models drops significantly on transferred adversarial examples while our deep sparse coding model (DSC) is immune. Details on these results can be found in [23]

changes in the parameter space supports the idea that this neuromorphic approach is a viable defense to adversarial images and increases one's confidence that the reconstruction will be both accurate and high quality.

2. Sparse coding has the best distortion to accuracy ratio while maintaining a high level of classification. Sparse coding is able to maintain a low reconstruction error and also successfully denoise adversarial perturbations from an input signal.

7 ACKNOWLEDGEMENTS

This material is based upon work supported by the Intel Corporation and the National Science Foundation under Grant No. 1846023.

REFERENCES

[1] Roland Baddeley. 1996. Visual-Perception-An Efficient Code in V1. *Nature* 381, 6583 (1996), 560–561.
[2] Eric B Baum, John Moody, and Frank Wilczek. 1988. Internal representations for associative memory. *Biological Cybernetics* 59, 4-5 (1988), 217–228.
[3] Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, et al. 2018. Loihi: A Neuromorphic Manycore Processor with On-Chip Learning. *IEEE Micro* 38, 1 (2018), 82–99.
[4] Gintare Karolina Dziugaite, Zoubin Ghahramani, and Daniel M Roy. 2016. A study of the effect of jpg compression on adversarial images. *arXiv preprint arXiv:1608.00853* (2016).
[5] Bradley Efron, Trevor Hastie, Iain Johnstone, Robert Tibshirani, et al. 2004. Least angle regression. *The Annals of statistics* 32, 2 (2004), 407–499.
[6] Peter Foldiak. 2003. Sparse coding in the primate cortex. *The handbook of brain theory and neural networks* (2003).
[7] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. 315–323.
[8] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014).
[9] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens van der Maaten. 2017. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117* (2017).
[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*. 1026–1034.
[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*. 770–778.
[12] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4700–4708.
[13] Jason Jo and Yoshua Bengio. 2017. Measuring the tendency of CNNs to learn surface statistical regularities. *arXiv preprint arXiv:1711.11561* (2017).
[14] Edward Kim, Darryl Hannan, and Garrett Kenyon. 2018. Deep Sparse Coding for Invariant Multimodal Halle Berry Neurons. *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)* (2018).
[15] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324.
[16] Zhinus Marzi, Soorya Gopalakrishnan, Upamanyu Madhow, and Ramtin Pedarsani. 2018. Sparsity-based Defense against Adversarial Attacks on Linear Classifiers. *arXiv preprint arXiv:1801.04695* (2018).
[17] John Mitro, Derek Bridge, and Steven Prestwich. 2018. Denoising Dictionary Learning Against Adversarial Perturbations. *arXiv preprint arXiv:1801.02257* (2018).
[18] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. 2016. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2574–2582.
[19] Bruno A Olshausen and David J Field. 1997. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision research* 37, 23 (1997), 3311–3325.
[20] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. 2016. The limitations of deep learning in adversarial settings. In *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on*. IEEE, 372–387.
[21] Christopher Rozell, Don Johnson, Richard Baraniuk, and Bruno Olshausen. 2007. Locally competitive algorithms for sparse approximation. In *IEEE International Conference on Image Processing*, Vol. 4. IEEE, IV–169.
[22] Peter F Schultz, Dylan M Paiton, Wei Lu, and Garrett T Kenyon. 2014. Replicating kernels with a short stride allows sparse reconstructions with fewer independent kernels. *arXiv preprint arXiv:1406.4205* (2014).
[23] Jacob M Springer, Charles S Strauss, Austin M Thresher, Edward Kim, and Garrett T Kenyon. 2018. Classifiers Based on Deep Sparse Coding Architectures are Robust to Deep Learning Transferable Examples. *arXiv preprint arXiv:1811.07211* (2018).
[24] Jiawei Su, Danilo Vasconcellos Vargas, and Sakurai Kouichi. 2017. One pixel attack for fooling deep neural networks. *arXiv preprint arXiv:1710.08864* (2017).
[25] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. 2019. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation* (2019).
[26] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199* (2013).
[27] Ping Tak Peter Tang, Tsung-Han Lin, and Mike Davies. 2017. Sparse coding by spiking neural networks: Convergence theory and computational results. *arXiv preprint arXiv:1705.05475* (2017).
[28] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. 2017. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204* (2017).
[29] Tobias Weyand, Ilya Kostrikov, and James Philbin. 2016. Planet-photo geolocation with convolutional neural networks. In *European Conference on Computer Vision*. Springer, 37–55.
[30] Weilin Xu, David Evans, and Yanjun Qi. 2017. Feature squeezing mitigates and detects carlini/wagner adversarial examples. *arXiv preprint arXiv:1705.10686* (2017).