Machine Foveation: An Application-Aware Compressive Sensing Framework

Ekdeep Singh Lubana*, Vinayak Aggarwal*, and Robert P. Dick†

*Indian Institute of Technology, Roorkee
Roorkee, Uttarakhand, 247667, India
ekdeeplubana@gmail.com

†University of Michigan
Ann Arbor, Michigan, 48105, USA

Abstract

Embedded vision applications generally face tight resource constraints. Biological vision systems are optimized to operate under similar conditions; they use highly heterogeneous sensing patterns to capture only the most valuable information within scenes. Our exploration of similar approaches in embedded systems has led to the design of Machine Foveation–a general-purpose, application-aware compressive sensing related framework that uses a cascaded network architecture integrating an autoencoder and application network to determine the importance of each pixel. The cascaded structure results in inherent regulation of the autoencoder network, forcing it to learn a representation that retains a given feature only if it is crucial to the overall application. The framework further uses scene awareness for reducing the number of bits necessary to represent the image data. This reduces sensed data at minimal or no decrease in task accuracy and reduces signal communication latency and corresponding energy consumption in embedded systems. For example, when evaluated on the Fashion-MNIST data set, channel bandwidth requirements are reduced by 77.37% and signal communication latency is reduced by 64.6%, with an accuracy loss of only 0.32%.

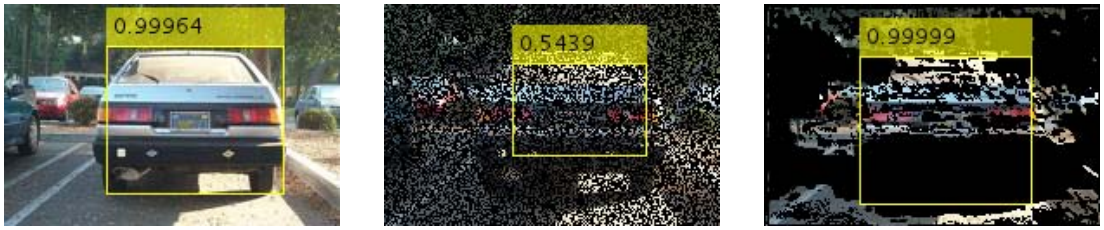
1 Introduction

Advances in deep learning enable machine vision systems to achieve near human accuracy in many applications. However, most machine vision algorithms are resource intensive and expensive to deploy in battery-powered, constrained environments. This is in contrast to highly optimized human vision systems, which achieve resource efficiency by using a heterogeneous sampling pattern to eliminate non-essential data and processing.

Image sensing in humans takes place at the retina. The center of the retina, called the fovea, has dense, color-sensitive photoreceptors (cones) and samples a small part of the scene (approximately 5%) at high resolution; meanwhile, the peripheral region has sparse color-insensitive, low-resolution photoreceptors (rods). Variable photopigment concentration further optimizes data transfer rates across the sensing region [1]. Such an application-aware, heterogeneous sensing pattern optimizes throughput by using low sampling rates at regions that rarely provide any consequential information, while maintaining high application accuracy [2].

Inspired by the high accuracy and efficiency of the human vision system, we have developed Machine Foveation–a compressive sensing related framework that exploits application-awareness as a prior to optimize sampling, thereby reducing memory requirements and communication requirements, while increasing efficiency and maintaining accuracy (see Figure 1). Specifically, Machine Foveation uses a cascaded

2375-0359/19/$31.00 ©2019 IEEE
DOI 10.1109/DCC.2019.00056



(a) Uniform sampling (b) Random sampling (c) Machine Foveation

Figure 1: (a) A uniformly sampled image of a car. (b) Using 40%, randomly located pixels results in significant accuracy degradation. (c) Machine Foveation reduces data processing volume while preserving accuracy via importance-aware pixel selection.

autoencoder–application network architecture to determine the importance of a sample to the given application. Pixels irrelevant or redundant for achieving high accuracy are discarded. Further, due to the sparsity of natural images, we find that pixels form clusters with similar probability distributions, called "typical distributions" (see Section 3.2). These clusters are skewed towards one or more intensity values and not uniformly distributed. Assigning eight bits to such pixels is wasteful, for their intensity distributions can be easily reconstructed using fewer, but more relevant, bits. Machine Foveation uses a search algorithm to assign appropriate bits to such clusters, while ensuring minimal reconstruction error. This enables Machine Foveation to achieve high compression ratios and high quality results. For example, for Fashion-MNIST, Machine Foveation reduces data volume by 77.37%, reduces signal communication latency and accompanying energy consumption by 64.6%, and attains classification accuracy similar (<0.32% loss) to that of the uniformly sampled images. Improvements are similar for other datasets.

This paper makes the following contributions. We introduce an application-aware, compressive sensing related framework that minimizes data volume by discarding irrelevant pixels and optimizing bit representations of the remaining pixels, while preserving decision quality (see Section 3.1). This results in reduced memory and throughput requirements. Information important to the machine vision task is maintained, resulting in minimal to no degradation in accuracy (<1%). Furthermore, adaptive bit assignment is used to decrease sampled and analyzed data by 75–85% (see Section 4).

2 Related Work

Compressive sensing focuses on finding ideal sampling routines for perfect signal reconstruction [3]. Essentially, an encoder forces the input into a high-sparsity domain, called the latent space, and a reconstruction algorithm maps from this latent space to the original, high-dimensionality input signal domain. Conventional approaches to compressive sensing in embedded systems exploit the high reconstruction abilities of such algorithms to draw inferences on the reconstructed signal, than the original input signal. Along these lines, most deep learning research on compressive sensing has also focused on signal reconstruction, e.g., recent work on reconstruction with generative adversarial networks (GANs) [4, 5].

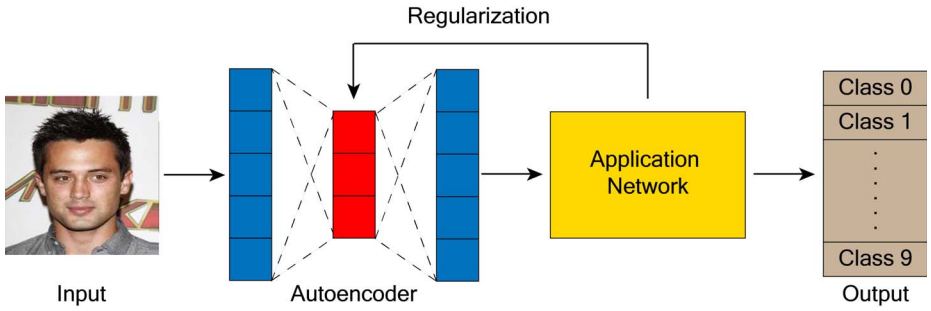


Figure 2: General network architecture for Machine Foveation. An application network acts as a regularizer for the autoencoder.

Our work, however, focuses on directly using the compressed measurements to draw inferences on the input and achieve high accuracy. This is achieved by using application-awareness as an effective prior, as demonstrated in Section 3. Thus, we depart from the conventional compressive sensing goals and literature. Earliest work on such application-driven sample discard was by Avidan and Shamir, who defined heuristic information metrics based on perceptual loss and discarded samples with low information content [6]. Similarly, Kabkab et al. [7] use application-driven compressive sensing too; however, they follow the conventional route in the inference stage, choosing to reconstruct the original input for classification. In contrast, we draw inferences directly from the compressed signal.

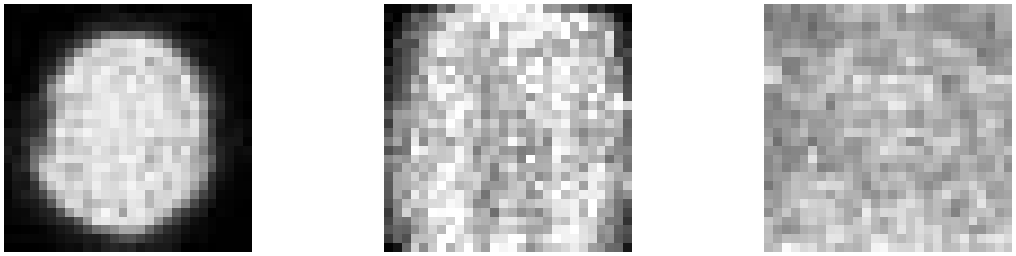
The work by Calderbank et al. [8] is closest to ours. They show that support vector machines trained on a compressed space have similar accuracy to the best trained linear classifiers in the uncompressed space. We extend their analysis by exploring non-linear classifiers, such as neural networks and convolutional networks. Lohit et al. [9] also capture and train a convolutional network to draw inferences on the compressed samples of images. While we are analyzing non-linear classifiers too, our use of an autoencoder to determine the sampling pattern results in much better data compression ratios. In comparison to their 75% data reduction for MNIST with 98.37% classification accuracy, we are able to achieve 93.5% data reduction with 98.89% classification accuracy.

3 Model

Machine Foveation is an autoencoder-based application-aware compressive sensing framework that optimizes sensing patterns to minimize sampled data with minimal reduction in accuracy. This results in reduced data throughput and memory requirements, which are especially important in emerging battery-powered wireless embedded applications. Machine Foveation either discards a pixel or optimizes its bit representation by skipping relatively unimportant bits.

3.1 Calculating pixel importance to overall analysis

The latent space generated by an autoencoder encodes structural information about the data, which is used as an inherent prior by Machine Foveation. Classification



(a) MNIST (b) F-MNIST (c) CelebA

Figure 3: Importance maps (resized to 30×30) corresponding to MNIST (classification), Fashion-MNIST (classification), and CelebA: Faces in the wild (regression).

and regression tasks trained on the latent space of an autoencoder are accurate if the model family incorporates the data generating distribution, e.g., , a neural network classifier trained on the latent space of an MNIST autoencoder has 98.8% accuracy.

Training two networks simultaneously, where one is able to fit the data generating distribution of the input, results in a regularization effect on the other [10]. Thus, we cascade an autoencoder with the application network, i.e., a network designed to draw inferences on the dataset (see Figure 2). The loss function and gradients backpropagate from the application network directly and impose a constraint on the latent space of the autoencoder, forcing it to learn a representation that retains only features crucial to the application. The imposition of a relevance constraint can be used to infer the importances of individual pixels. Specifically, the magnitudes of weights corresponding to connections between the input and the first hidden layers can be used to assign importance factors to pixels.

Consider an autoencoder A with n hidden neurons in its first hidden layer. The autoencoder is trained on a given dataset of input size $m \times n$, alongside an application network N. Thus, each of the n neurons has mn connections to the input layer. We normalize the absolute values of the mn weights for each neuron to a scale of [0, 1]. If a pixel is important (has a large weight) to any hidden neuron, it is vital to the activation of that neuron and is therefore retained. The *importance factor* of the pixel corresponding to index (i, j) is calculated as follows:

$$IF_{i,j} = \max_{1 \leq k \leq n} abs\left(weight_{normalized}\left(h_{i,j,k}\right)\right), \tag{1}$$

where $weight_{normalized}\left(h_{i,j,k}\right)$ denotes the normalized weight corresponding to the connection between the $(i, j)^{th}$ pixel and k^{th} neuron.

An importance matrix, M, is used to generate an importance map, several of which are shown in Figure 3. A map is defined as $M_{i,j} = IF_{i,j}$.

3.2 Adaptive bit assignment

Natural images are generally sparse, resulting in minimal variation in the statistics of groups of (often spatially local) pixels. For example, if the chosen imaging setup resulted in regions of darkness at certain locations, the typical distribution of those pixels will be skewed towards 0. Assignment of 8 bits to such pixels is wasteful,

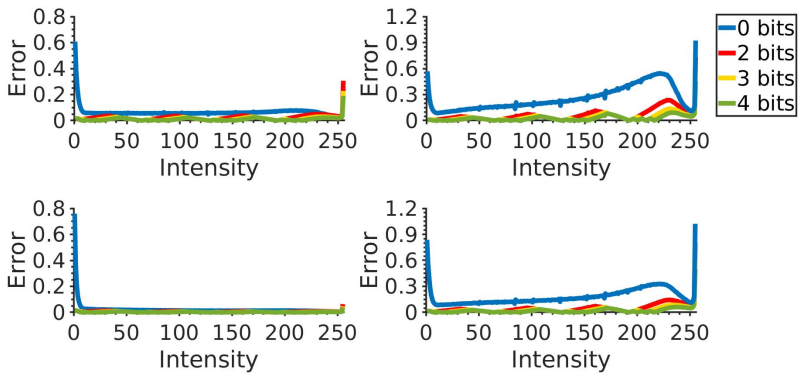


Figure 4: Running an unsupervised clustering algorithm on the probability distributions of the pixels in the Fashion-MNIST dataset results in four categories with similar probability distributions. Reconstruction errors relative to the original distributions for 0-bit, 2-bit, 3-bit, and 4-bit representations are shown.

for they can be well represented using fewer and low-significance bits. Other spatial scene properties common in embedded applications have similar implications, e.g., some regions are more relevant to classification than others.

Each pixel $P_{i,j}$ is a random variable in range [0, 255]. According to the asymptotic equipartition theorem and weak law of large numbers, if n random variables are drawn from an i.i.d. distribution, the resulting series of variables can be used to infer that exact probability distribution. Thus, the pixels across the dataset at location (i, j) can be assumed to be random variables drawn in an i.i.d. manner from the original probability distribution of the pixel $P_{i,j}$. If the dataset is large enough, the original probability distribution of $P_{i,j}$ can be accurately inferred and is called the "typical distribution".

Table 1: Bit Representations vs. Reconstruction Thresholds for Fashion-MNIST.

Category	Reconstruction Error		
	20%	10%	5%
Category 1	8, 7	8, 7, 5	8, 7, 5, 4
Category 2	8, 7	8, 7, 6	8, 7, 6, 4
Category 3	8, 7	8, 7, 5	8, 7, 5, 4
Category 4	8, 7	8, 7, 6	8, 7, 6, 4

To determine a set of pixels with similar typical distributions, we use unsupervised neural network clustering to map pixels to categories. Typical distributions of each category are averaged to determine the category's representative probability distribution.

To find the ideal bit assignment for pixels corresponding to a particular category, we use a greedy search across the available bit resolution (8 bits for images) and reconstruct the representative distribution of that category. The search process identifies the minimal set of bits honoring the following reconstruction bound:

$$\min \left(\sum_{I=1}^{255} P_{category}\left(I\right) \left(I_{1\text{–}8} - I_{rep}\right) \right), \tag{2}$$

where $P_{category}\left(I\right)$ is the probability with which pixels in that particular category are assigned an intensity value of I; while I_{rep} is the reconstruction of I using the

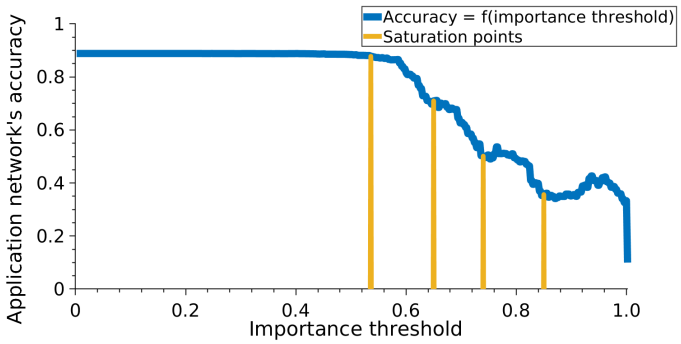


Figure 5: The original classification network trained on Fashion-MNIST saturates in accuracy at several points (shown by yellow intersection lines).

bit-representation "*rep*". The summation starts from 1 because intensity 0 can be represented by any bit representation. We store the representations and their corresponding errors in an array (see Table 1 and Figure 4 for the reconstruction errors of thresholds for Fashion-MNIST categories).

3.3 Segregation into importance sets

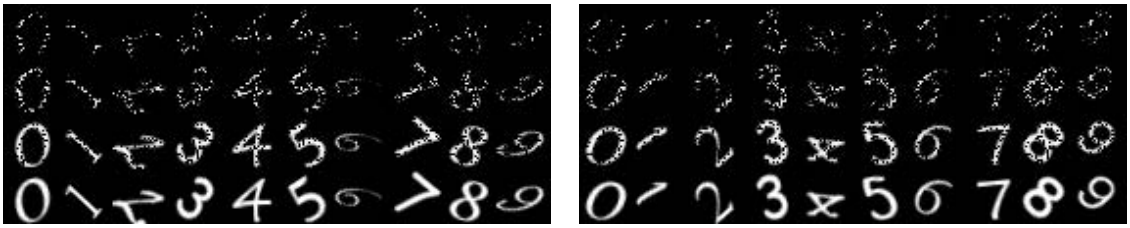
Since the pixels belonging to a category need not share the same importance, we adaptively assign a different reconstruction error to each pixel, based on its importance to the application network. The importance map, M, is used to segregate the pixels into importance sets by considering the relationship between error and accuracy. The importance map is used to create the mask associated with an importance threshold, T ($Mask = M \geq T$).

The pixel sets associated with several importance thresholds are used to define classes that are used to optimize the bit representations. In practice, only a few such thresholds are necessary to achieve high accuracy. We determined these thresholds via the following heuristic.

The threshold is incrementally decreased from 1 to 0. At each step, the test images are multiplied with the mask and evaluated for network accuracy on the application network. The accuracy is plotted as a function of importance thresholds (see Figure 5). As the threshold reduces, there are accuracy plateaus, i.e., regions in which small further decreases in the importance threshold do not improve accuracy. This implies that pixels corresponding to features of a particular structure type have been used; further improvements in accuracy require additional features expected by the network. We call these saturation points and assign pixels ranging between two saturation points to a single importance set. Importance sets corresponding to higher importance factors are assigned a smaller reconstruction error (typically, $<5\%$). Figure 4 shows the representative distribution of the different categories of pixels in Fashion-MNIST.

3.4 Training

The network is first trained in the cascaded structure (Step 1). Then, (Steps 2 and 3) each pixel is assigned a category for bit representation and an importance set. For



(a) Neural network classifier (b) LeNet-5 (Convnet) classifier

Figure 6: Examples of compressively sampled MNIST images corresponding to (a) neural network and (b) convolutional network based classifiers. See Table 2a and Table 2b for details on accuracy and number of samples used.

fine tuning, the autoencoder is removed and the application network is independently trained. Training is done in rounds, corresponding to each saturation point, and terminates when accuracy ceases to increase (see Figure 5). Our experiments (see Section 4) demonstrate that fine-tuning corresponding to just the first saturation point, i.e., the most important pixels, results in network accuracy within <1% of the network trained on uncompressed, uniformly sampled images. Thus, multiple training rounds are generally unnecessary.

4 Experiments

This section evaluates the compression efficiency of Machine Foveation on several classification and regression datasets–MNIST [11], Fashion MNIST [12], and CelebA:Faces-in-the-wild dataset [13]. We use several baseline architectures and compare Machine Foveation against a system using uncompressed, uniformly high resolution images.

4.1 MNIST: digit classification

MNIST contains 60,000 training images and 10,000 test images. We evaluate Machine Foveation for digit classification and compare it with a two-layer neural network (see Table 2a) and LeNet-5 (see Table 2b). In Table 2a, we report the accuracy when using pixels exceeding an importance threshold. These thresholds were determined using the approach described in Section 3.3. The average number of bits per pixel (*BPP*), the number of samples (*n. samp.*) above the chosen importance threshold, and the total reduction in data (*Data red.*) are also indicated. Using the top-14% most important pixels reduces accuracy by 0.22% with 88.25% data reduction; reducing the importance threshold eliminates accuracy reduction with 78.87% data reduction. LeNet-5 achieves similar results, as noted in Table 2b. Examples of compressively sampled images are shown in Figure 6.

4.2 Fashion-MNIST: classification of complex articles

Fashion-MNIST contains 60,000 training images and 10,000 test images. The classes correspond to different clothing articles; the dataset is far more complex than (the

Table 2: MNIST Classification Results

(a) Neural Net Classifier

Imp. thresh.	Accuracy	bits/ pixel	n. samp.	Data red.
0.86	98.58%	0.94	23.72%	88.25%
0.70	98.68%	1.36	37.50%	83%
0.24	98.80%	1.69	54.33%	78.87%
Uniform	98.80%	8.00	100.0%	0%

(b) LeNet-5

Imp. thresh.	Accuracy	bits/ pixel	n. samp.	Data red.
0.84	98.89%	0.52	13.13%	93.50%
0.71	99.16%	1.18	35.07%	85.25%
0.29	99.36%	2.06	78.9%	74.25%
Uniform	99.40%	8.00	100.0%	0%

Table 3: F-MNIST Classification Results

(a) Neural Net Classifier

Imp. thresh.	Accuracy	bits/ pixel	n. samp.	Data red.
0.83	88.10%	1.18	29.59%	85.25%
0.71	88.40%	1.68	54.46%	78.98%
0.24	88.50%	2.18	79.84%	72.64%
Uniform	88.80%	8.00	100.0%	0%

(b) Convnet Classifier

Imp. thresh.	Accuracy	bits/ pixel	n. samp.	Data red.
0.83	89.16%	0.74	18.6%	90.75%
0.69	90.08%	1.26	35.9%	84.25%
0.23	91.49%	1.81	63.3%	77.37%
Uniform	91.81%	8.00	100.0%	0%

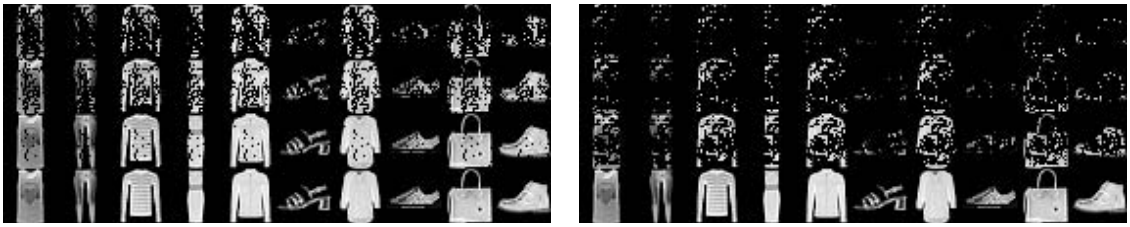
numerical) MNIST. We use the benchmark best-accuracy neural network and convolutional network architectures as base cases for comparison [12]. Tables 3a and 3b show the results. The neural network architecture suffers 0.3% reduction in accuracy with 72.64% reduction in data, while the CNN architecture suffers 0.32% reduction in accuracy with 77.37% reduction in data. Examples of compressively sampled images are shown in Figure 7.

4.3 CelebA: face detection (regression)

Table 4: CelebA Face Detection Results

Imp. thresh.	RMSE	BPP	n. samp.	Data red.
0.68	2.33	1.87	38%	76.62%
0.54	2.25	3.07	74%	61.62%
Uniform	2.21	8.00	100.0%	0%

CelebA:Faces-in-the-wild dataset contains 202,599 images of celebrities' faces. We use a CNN architecture as the base case for comparison. We calculate root mean squared error relative to ground truth bounding box coordinates. As detailed in Table 4, the RMSE increases by



(a) Neural network classifier (b) Convolutional network classifier

Figure 7: Examples of compressively sampled Fashion-MNIST images corresponding to a (a) neural network and (b) convolutional network based classifier. See Table 3a and Table 3b for details on accuracy and number of samples used.

Figure 8: Comparison of bounding boxes determined using uniform (red boundaries), top-74% most important pixels (yellow boundaries) and top-38% most important pixels (green boundaries).

Table 5: Signal Communication Latency. The compression ratio and relative duration values have been calculated w.r.t compression and absolute duration (75.5 sec) values for JPEG compression, respectively.

Imp. thresh.	Compression ratio	Absolute Duration (s)	Relative Duration
0.83	14.14×	14.53	19.25%
0.69	8.43×	22.77	30.19%
0.23	7.10×	26.65	35.44%

0.04 with 61.62% reduction in data. We show the bounding boxes determined using the compressively sampled images and uniform images in Figure 8.

4.4 Impact on signal communication latency

We evaluate the impact of Machine Foveation on signal communication latency in an IoT system. Our platform for evaluation consists of a Raspberry Pi 3 board and an Android mobile phone. The tests are performed on the Fashion-MNIST dataset.

We sample 1,000 randomly selected Fashion-MNIST images using Machine Foveation, assuming the images are to be passed through a ConvNet classifier. This produces a highly sparse set of images (see Figure 7). In order to exploit this sparsity and the fact that the location of important pixels is known to both the transmitter and the receiver, we flatten the images to vectors. This stream of bits is stored in a binary file and communicated from the Raspberry Pi 3 to an Android mobile phone. The file is 7–14× smaller than the JPEG compressed, uniformly sampled images. For the batch of 1,000 images, this results in 65–80% reduction in signal communication latency and accompanying energy consumption. See Table 5 for absolute signal communication time values and Table 3b for corresponding application accuracy.

5 Conclusion

We described Machine Foveation, a compressive sensing framework that assigns an importance factor to each sample in a scene and uses that information to construct an optimized sensing pattern for high task-accuracy at minimal data requirements. Inspired by the heterogeneous sensing pattern in human vision, Machine Foveation uses an adaptive bit assignment methodology that preserves the probability distribution of a pixel, while reducing the number of bits used to represent it. The impact of the framework on accuracy and data quantity was evaluated for three different applications: classification (MNIST and Fashion-MNIST) and regression (Face detection). The results indicate that Machine Foveation reduces data volume by 75–85%, communication latency and accompanying energy consumption by 65%–80%, with less than 1% decrease in accuracy.

References

[1] Eric R Kandel, James H Schwartz, Thomas M Jessell, Department of Biochemistry, Molecular Biophysics Thomas Jessell, Steven Siegelbaum, and AJ Hudspeth, *Principles of Neural Science*, vol. 4, McGraw-Hill, New York, 2000.

[2] Ron Dekel, "Human perception in computer vision," Tech. Rep. arXiv:1701.04674, ArXiV, 2017.

[3] Emmanuel J Candès and Michael B Wakin, "An introduction to compressive sampling," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 21–30, 2008.

[4] Ashish Bora, Eric Price, and Alexandros G. Dimakis, "AmbientGAN: Generative models from lossy measurements," in *Int. Conf. on Learning Representations*, 2018.

[5] Zachary C. Lipton and Subarna Tripathi, "Precise recovery of latent vectors from generative adversarial networks," in *Int. Conf. on Learning Representations*, 2017.

[6] Shai Avidan and Ariel Shamir, "Seam carving for content-aware image resizing," in *ACM Trans. Graphics*, 2007, vol. 26, p. 10.

[7] Maya Kabkab, Pouya Samangouei, and Rama Chellappa, "Task-aware compressed sensing with generative adversarial networks," in *AAAI Conf. on Artificial Intelligence*, 2018.

[8] Robert Calderbank, Sina Jafarpour, and Robert Schapire, "Compressed learning: Universal sparse dimensionality reduction and learning in the measurement domain," Tech. Rep., Princeton University, 2009.

[9] S. Lohit, K. Kulkarni, and P. Turaga, "Direct inference on compressive measurements using convolutional neural networks," in *Int. Conf. on Image Processing*, Sept 2016, pp. 1913–1917.

[10] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio, *Deep Learning*, vol. 1, MIT press Cambridge, 2016.

[11] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov 1998.

[12] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms," *ArXiv e-prints*, Aug. 2017.

[13] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang, "Deep learning face attributes in the wild," in *Proc. of Int. Conf. on Computer Vision*, December 2015.