https://doi.org/10.2352/ISSN.2470-1173.2020.10.IPAS-310

CNN performance dependence on linear image processing

Khalid Omer; Wyant College of Optical Sciences, The University of Arizona; Tucson, Arizona, USA
Luca Caucci; Department of Medical Imaging,The University of Arizona; Tucson, Arizona, USA
Meredith Kupinski; Wyant College of Optical Sciences, The University of Arizona; Tucson, Arizona, USA

Abstract

This work reports on convolutional neural network (CNN) performance on an image texture classification task as a function of linear image processing and number of training images. Detection performance of single and multi-layer CNNs (sCNN/mCNN) are compared to optimal observers. Performance is quantified by the area under the receiver operating characteristic (ROC) curve, also known as the AUC. For perfect detection AUC = 1.0 and AUC = 0.5 for guessing. The Ideal Observer (IO) maximizes AUC but is prohibitive in practice because it depends on high-dimensional image likelihoods. The IO performance is invariant to any full-rank, invertible linear image processing. This work demonstrates the existence of full-rank, invertible linear transforms that can degrade both sCNN and mCNN even in the limit of large quantities of training data. A subsequent invertible linear transform changes the images' correlation structure again and can improve this AUC. Stationary textures sampled from zero mean and unequal covariance Gaussian distributions allow closed-form analytic expressions for the IO and optimal linear compression. Linear compression is a mitigation technique for high-dimension low sample size (HDLSS) applications. By definition, compression strictly decreases or maintains IO detection performance. For small quantities of training data, linear image compression prior to the sCNN architecture can increase AUC from 0.56 to 0.93. Results indicate an optimal compression ratio for CNN based on task difficulty, compression method, and number of training images.

Introduction

This work compares the detection performance of single and multi-layer CNNs (sCNN/mCNN) with the Bayesian ideal observer (IO). Performance is quantified by the area under (AUC) the Receiver Operating Characteristic (ROC) curve. The IO maximizes AUC as well as other detection task figures of merit [1]. The AUC of the IO is used as a benchmark for task difficulty and a comparative tool for evaluating observer-model detection performance [2]. For many tasks the IO can be well approximated by a CNN given an adequate quantity of training data [3]. In this work, the relationship between linear transforms on the image data and classification performance is investigated. This work demonstrates the existence of full-rank linear transforms that degrade CNN performance even for large quantities of training data. A subsequent full-rank linear transform can improve the CNN performance, presumably by restoring the correlation structure to a pattern recognized by the CNN. The disparity between CNN and IO is also evaluated for varying quantities of training images.

An image detection task from Gaussian distributed, zero mean image data allows closed-form analytic expressions for the IO and optimal linear compression. When the covariance matrices of these two classes are unequal the data is called heteroscedastic and the images have the appearance of different textures. Two image classes are simulated on a 64×64 pixel grid with dimensionality $M = 64^2$. A given image is denoted by the $M \times 1$ vector $\mathbf{g}$. The covariance matrix of each class is chosen to be circulant with a Gaussian kernel and parameterized by a scalar-valued correlation length, denoted σ. Changing the correlation length yields images of varying spatial textures. Prior work demonstrates that CNN learning is biased towards image texture, as opposed to image shape [4].

An $L \times M$ compression matrix multiplied by $\mathbf{g}$ yields a compressed image when $L < M$. With certain eigenspectrum assumptions, the Fukunaga Koontz transform (FKT) is the low-rank approximation to the optimal classifier for zero mean, heteroscadastic, and normally distributed data [5, 6] . When the quantity of training data is limited, linear compression using FKT can increase detection performance of the CNN. The "curse of dimensionality" is evident from a performance peak at an optimal compression, followed by a decrease [7]. The CNN performance on the linearly compressed images is heavily dependent upon the compression matrix. An adaptation of FKT widely used in pattern recognition, is called a tuned basis function (TBF) [8]. An optimal solution to $\mathbf{T}_0$ for the quadratic test statistic $\mathbf{g}^\dagger \mathbf{T}_0 \mathbf{g}$.

Mathematical Methods

Consider the relationship between an image and an object as

$$\mathbf{g} = \mathscr{H} f + \mathbf{n}. \tag{1}$$

Here $\mathbf{g}$ is an $M \times 1$ vector of measurements made by an imaging system represented as a continuous-to-discrete operator $\mathscr{H}$; the measurements of the continuous object f are corrupted by measurement noise $\mathbf{n}$. We will consider post-processing signal detection. That is to say the forward imaging model $\mathscr{H}$ is fixed and can even be unknown since only the statistics of the image data will be used.

The M-dimensional vector $\mathbf{g}$ will represent the input to the classifier, which will classify this vector as either belonging to the population corresponding to the probability density function (PDF) $pr_1(\mathbf{g})$ or the population corresponding to the PDF $pr_2(\mathbf{g})$. This vector may be a direct image, a reconstructed image, or the raw data being produced by an imaging system.

The IO uses the log-likelihood ratio

$$\lambda(\mathbf{g}) = \ln[\Lambda(\mathbf{g})] = \ln[pr_1(\mathbf{g})] - \ln[pr_2(\mathbf{g})] \tag{2}$$

as a decision variable and compares the result to a threshold. Here $\Lambda(g)$ is the likelihood ratio $pr_1(\mathbf{g})/pr_2(\mathbf{g})$. If the decision variable is above the threshold, then the data vector is assigned to $pr_1(\mathbf{g})$, and otherwise it is assigned to $pr_2(\mathbf{g})$.

In imaging applications the dimension M of the vector $\mathbf{g}$ (*e.g.* the number of pixels) is very large. Assuming for $i = 1,2$ that $pr_i(\mathbf{g})$ is a Gaussian PDF, with zero mean and covariance matrix $\mathbf{K}_i$, the log-likelihood ratio is

$$\lambda(\mathbf{g}) = \mathbf{g}\mathbf{K}_2^{-1}\mathbf{g}^\dagger - \mathbf{g}\mathbf{K}_1^{-1}\mathbf{g}^\dagger. \tag{3}$$

Here scale factors and terms with do no depend on $\mathbf{g}$ have been ignored. Implementing the log-likelihood ratio, even with these PDF assumptions, incurs two major challenges. The first challenge is computational; even for Gaussian PDFs we will have to invert two $M \times M$ covariance matrices $\mathbf{K}_i$, which may not be feasible if, for example, the input images contain millions of pixels. The second challenge is that, if we are estimating the image statistics from data, which is often the case, we will need a very large number of samples to get reliable estimates. For example, in the Gaussian case, the number of samples needs to be at least M to get invertible estimates of the $\mathbf{K}_i$, and typically needs to be an order of magnitude greater to get reliable estimates. This provides a motivation for trying to reduce the dimension of the data vector before implementing the log-likelihood ratio.

Linear data transformation is implemented by an $L \times M$ dimensional matrix $\mathbf{T}$ via the equation $\mathbf{t} = \mathbf{T}\mathbf{g}$. Using terminology from the perception literature, each row of $\mathbf{T}$ is referred to as a channel [9]. In this paper $\mathbf{t}$ will be called a channelized image or a channelized data vector. The number L is the dimension of the channelized data and satisfies $L \ll M$. Channelized data are preferable for mathematical observers since calculating a decision variable usually involves the estimation of parametric likelihoods [10]. In the channelized representation this estimation can be much more accurate given common constraints on finite training data [11, 12]. Computational needs are also lower because the inverse of a covariance matrix, required for likelihood evaluations, is now $L \times L$ instead of $M \times M$.

We will always assume that $\mathbf{T}$ is a full rank matrix so that the channels are linearly independent. We will assume that the PDFs for the channelized data for the two populations are given by zero-mean normal distributions

$$pr_i(\mathbf{t}) = \frac{\exp\left[-\frac{1}{2}\mathbf{t}^\dagger\left(\mathbf{T}\mathbf{K}_i\mathbf{T}^\dagger\right)^{-1}\mathbf{t}\right]}{\sqrt{(2\pi)^L \det\left(\mathbf{T}\mathbf{K}_i\mathbf{T}^\dagger\right)}} \tag{4}$$

for $i = 1,2$. Note that the covariance of the channelized data are related to the covariance of the image data by $\mathbf{C}_i = \mathbf{T}\mathbf{K}_i\mathbf{T}^\dagger$ where $\mathbf{C}_i$ is the $L \times L$ covariance matrix of the channelized data.

Covariance matrix eigenanalysis is rarely practical for modern imaging systems since an image is comprised of several million elements. Fukunaga and Koontz were the first to suggest covariance matrix eigendecomposition for detection and classification tasks [13]. This FKT method uses a matrix $\mathbf{T}$ to transform the data so that $\mathbf{T}(\mathbf{K}_1 + \mathbf{K}_2)\mathbf{T}^\dagger$ equals the identity matrix. This equality guarantees that both covariance matrices of the transformed data will have the same eigenvectors. Furthermore, the sum of the two eigenspectra, when eigenvalues associated with the same eigenvector are added, is equal to one. Consequently the transformation by $\mathbf{T}$ makes the *strongest* eigenvectors of one class the *weakest* eigenvectors of the other.

Numerical Methods

Formation of the CNN architecture is done through the TensorFlow Keras API in Python. Tensor operations are executed utilizing 4 Nividia 1080Ti graphic cards. Network construction is a heuristic model based on prior knowledge regarding the image ensemble. For example, kernel size selection can be estimated based on the correlation length. As the correlation length increases, the spatial variability in the image decreases, allowing for a larger kernel size to be used without affecting detection performance. Architecture design consist of a first layer containing 32 non-zero padded convolutional kernels of size 2×2 of stride 1, followed by max-pooling with the same stride and kernel size. All architectures utilize binary-cross-entropy as the loss function. Dropout is set to 0.75, batch normalization [14], and LeakyRelu set to 0.03. For the mCNN, there are three additional convolutional layers with 12 non-zero padded kernels of size 4×4 . The other additional layer hyperparameters are the same as for the sCNN (excluding dropout, as dropout is only applied to the first layer and fully connected layer). For the mCNN, dropout on the first and last layer are zero for $\sigma_1 = 1.0$ and $\sigma_2 = 0.4$ pixels. These large differences between correlation length require adjustments to optimize the network architecture. All network nodes were activated to increase AUC for this long correlation length difference. Batch size is approximately 10% of the training set but is slightly varied to increase performance depending on task difficulty and number of training images.

When trained, the channelized array $(L \times 1)$ is reshaped into a $(\sqrt{L} \times \sqrt{L})$ matrix. This transformation is done to allow for 2D convolution to occur, and to conserve the sCNN architecture. L varies from 16 - 4096, where L must be integer squares. The value of 16 was chosen to be the minimum L value in order to preserve the sCNN throughout the experiment, values smaller than 16 would require the kernel size of 2×2 to become a 1×1 to allow for the minimum stride = 1.

The number of trainable parameters for the sCNN is equal to

$$N_s = 737 + 4096\left(\sqrt{L} - 2\right)^2 \tag{5}$$

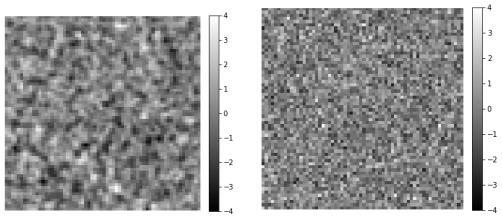
for mCNN equals

$$N_m = 3533 + 1536\left(\sqrt{L} - 14\right)^2. \tag{6}$$

Here L is the number of elements in an image or compressed image. Image compression decreases the number of parameters utilized in the CNN. For the sCNN, L values between 16 to 4096 have a trainable parameter range between $N_s = 803,553$ to $N_s = 15,745,761$. The mCNN is trained only when $L = M$, and uses $N_m = 3,843,533$ parameters. When $L > 44$, sCNN has more trainable parameters that mCNN. The sCNN has more parameters than mCNN due to the flattening on the last convolutional layer followed by a fully connected layer the number of trainable parameters in the CNN is related to the product of: number of parameters in last layer, number of kernels, and number of fully connected parameters.

Generating Images

A simple covariance model is chosen to provide an analytic solution for the IO and FKT. Two classes of images are simulated on a 64×64 pixel grid; $M = 64^2$. The covariance matrix of each



(a) Sample from $pr_1(\mathbf{g})$ (b) Sample from $pr_2(\mathbf{g})$

Figure 1: Example images from classes: (a) $\sigma_1 = 1.00$ and (b) $\sigma_2 = 0.40$ pixels. The visual difference between the two classes is obvious.

class is chosen to be circulant with a Gaussian kernel and parameterized by a correlation length as in

$$[\mathbf{K}_i]_{\mathbf{n},\mathbf{m}} = \exp\left(\frac{-\left((n_x - m_x)\text{mod}64\right)^2 - \left((n_y - m_y)\text{mod}64\right)^2}{2\sigma_i^2}\right) \tag{7}$$

where σ_i is the correlation length of the i^{th} class in units of pixels. The two-dimensional vectors $\mathbf{n}$ and $\mathbf{m}$ are pixel indices. Both classes are zero mean, *i.e.* all 64 elements of $\bar{\mathbf{g}}_1$ and $\bar{\mathbf{g}}_2$ equal 0. A short correlation length yields images of higher frequency content then a longer correlation length; see Figure 1.

Channelized Images: Linear Data Transformation

Multiplying an $L \times M$ matrix and the $M \times 1$ vector $\mathbf{g}$ reduces the dimension of the data to $L \times 1$ and the compression ratio is L/M. The matrix is fully populated of rank L. In this work the IO and CNN detection performance are compared for three linear data reduction methods: 1) FKT compression denoted by the matrix $\mathbf{T}$, 2) linear compression without any prior knowledge denoted $\mathbf{R}$, and 3) FKT compression subsequent to uncompressed $\mathbf{R}$ (*i.e.* $L = M$) denoted $\tilde{\mathbf{T}}$. FTK is the optimal linear compression method given the constraint that images are Gaussian distributed, zero-mean, and heteroscadastic. On the other extreme, linear compression with no prior knowledge is implemented by populating a matrix with samples from a uniform distribution. The channelized data are still zero-mean Gaussian distributed with covariance matrices: $\mathbf{TK_iT}^\dagger$, $\mathbf{RK_iR}^\dagger$, or $\tilde{\mathbf{T}}\mathbf{RK_iR}^\dagger\tilde{\mathbf{T}}^\dagger$.

For $L = M$ the matrices $\mathbf{T}$, $\mathbf{R}$, and $\tilde{\mathbf{T}}$ are full-rank, invertible, and $\mathbf{g}$ has not been compressed. The data-processing inequality guarantees that information to discriminate the classes has not been increased by post-processing [15]. For $L = M$ these operators are invertible and therefore information has not been lost when the images are not compressed. Figures 1, 2, 3, and 4 are example images with two difference correlation lengths and uncompressed channelized images using matrices $\mathbf{T}$, $\mathbf{R}$, and $\tilde{\mathbf{T}}$; respectively.

The FKT matrix is populated by L eigenvectors of $\mathbf{K}_2^{-1}\mathbf{K}_1$ with corresponding eigenvalues κ_l. IO AUC can be maximized when these eigenvectors are chosen to have the L largest values of $\kappa_l + \kappa_l^{-1}$ [6]. Then the compressed data is

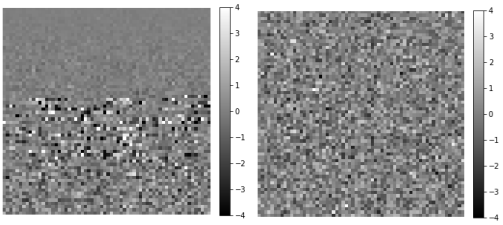
$$\mathbf{t} = \mathbf{Tg} \tag{8}$$

where $\mathbf{T}$ is an $L \times M$ matrix and FKT makes the *strongest* eigenvectors of one class the *weakest* eigenvectors of the other. The FKT ensures that $\mathbf{TK_1T}^\dagger + \mathbf{TK_2T}^\dagger = \mathbf{I}$ where $\mathbf{I}$ is the identity matrix. Consequently, when the variance in a given pixel is low for one class it is high for the other. This variance difference is visible in samples of $\mathbf{t}$ for $L = M$ case given in Figure 2.

Without an prior knowledge of $pr(\mathbf{g})$ compression can be implemented by

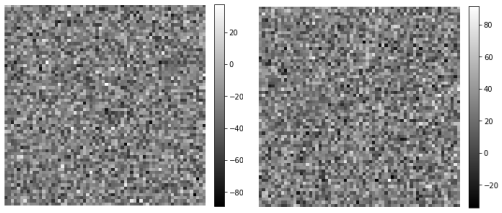
$$\mathbf{r} = \mathbf{Rg}. \tag{9}$$

Here $\mathbf{R}$ is a $L \times M$ matrix populated by elements sampled from a uniform distribution over $[0, 1)$. For the special case of $L = M$ no



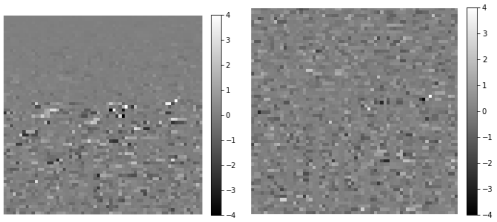
(a) Sample from $pr_1(\mathbf{t})$ (b) Sample from $pr_2(\mathbf{t})$

Figure 2: Example channelized images $\mathbf{t}$ from FKT for classes: (a) $\sigma_1 = 1.00$ and (b) $\sigma_2 = 0.40$ pixels. The FKT constrains pixels of high variance in one class to have low variance in the other. Here $\mathbf{t} = \mathbf{Tg}$ and example images $\mathbf{g}$ are shown in Figure 1. Visible differences between the classes are present in both $\mathbf{g}$ and $\mathbf{t}$ but the correlation structure is different.



(a) Sample from $pr_1(\tilde{\mathbf{g}})$ (b) Sample from $pr_2(\tilde{\mathbf{g}})$

Figure 3: Example channelized images $\tilde{\mathbf{g}}$ from full-rank linear transform for classes: (a) $\sigma_1 = 1.00$ pixels and (b) $\sigma_2 = 0.40$ pixels. Here $\tilde{\mathbf{g}} = \mathbf{Rg}$ and images $\mathbf{g}$ are shown in Figure 1. There is no visible difference between the two classes after this full-rank linear transform $\mathbf{R}$.



(a) Sample from $pr_1(\tilde{\mathbf{t}})$ (b) Sample from $pr_2(\tilde{\mathbf{t}})$

Figure 4: Example channelized images $\tilde{\mathbf{t}}$ from FKT for classes: (a) $\sigma_1 = 1.00$ pixels and (b) $\sigma_2 = 0.40$ pixels. Here $\tilde{\mathbf{t}} = \tilde{\mathbf{T}}\tilde{\mathbf{g}}$ and channelized images $\tilde{\mathbf{g}}$ are shown in Figure 3. The visible differences between the images $\mathbf{g}$ (see Figure 1) which was lost in the uncompressed channelized images $\tilde{\mathbf{g}}$ (see Figure 3) has been restored by FTK channelized images shown in this figure.

compression takes place which is denoted $\tilde{\mathbf{g}} = \mathbf{R}\mathbf{g}$. Uncompressed channelized images $\tilde{\mathbf{g}}$ are shown in Figure 3.

The FTK applied to the $\tilde{\mathbf{g}}$ image set is denoted

$$\tilde{\mathbf{t}} = \tilde{\mathbf{T}}\tilde{\mathbf{g}}. \tag{10}$$

Here, $\mathbf{R}$ is now full rank of size $M \times M$ and $\tilde{\mathbf{T}}\left[\mathbf{R}\mathbf{K}_1\mathbf{R}^\dagger + \mathbf{R}\mathbf{K}_2\mathbf{R}^\dagger\right]\tilde{\mathbf{T}}^\dagger = \mathbf{I}$ is the FKT constraint. Compression is achieved via the $L \times M$ matrix $\tilde{\mathbf{T}}$ when $L < M$. Once again, when the variance in a given pixel is low for one class it is high for another. This variance difference is visible in Figure 4.

Estimating Observer Performance

The last node of the network consist of a sigmoid activation function. This sigmoid produces an estimate of the posteriors $pr(i|\mathbf{g})$ [16]. For the case of equal prevalence $pr(i=1) = pr(i=2)$, the posteriors are

$$pr(i=1|\mathbf{g}) = \frac{1}{1+\Lambda(\mathbf{g})} \tag{11}$$

where $\Lambda(\mathbf{g})$ is the likelihood-ratio defined in Equation 2 and $pr(i=2|\mathbf{g}) = 1 - pr(i=1|\mathbf{g})$. The posteriors are evaluated on a set of testing images to generate an ROC curve; the AUC is estimated by trapezoidal integration. The average values of AUC estimates did not change for more than 20,000 testing images (10,000 $pr(i=1|\mathbf{g})$ and 10,000 $pr(i=2|\mathbf{g})$) so this quantity was selected as an asymptotic estimate.

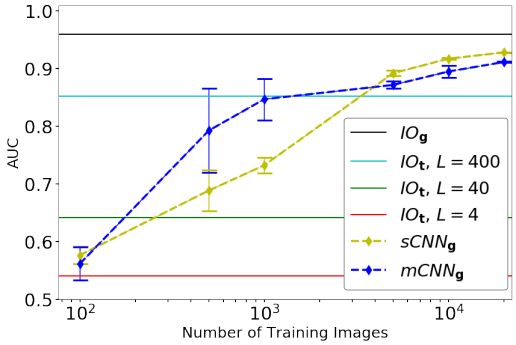
Results

In this work three observer models are compared: IO, sCNN, and mCNN. The detection task performance of these observers are computed from four image data sets: original image data $\mathbf{g}$, image data compressed by FKT $\mathbf{t}$, linear compression with no prior knowledge $\mathbf{r}$, and FKT compression subsequent to a non-compressive (*i.e.* $L = M$) invertible linear transform denoted $\tilde{\mathbf{t}}$. Detection performance for a specific observer model and type of data set is denoted, for example $\text{sCNN}_\mathbf{t}$ for a single layer CNN on an FKT channelized image set.

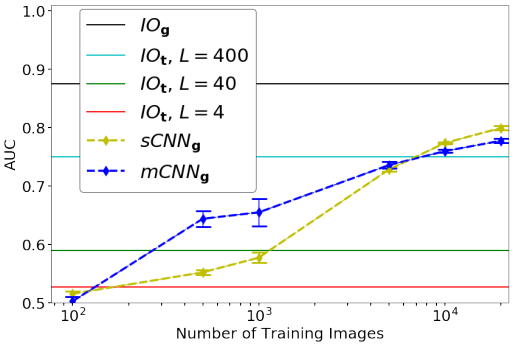
In Figure 5 the AUCs are compared for: $\text{IO}_\mathbf{g}$, $\text{IO}_\mathbf{t}$, $\text{sCNN}_\mathbf{g}$, and $\text{mCNN}_\mathbf{g}$ as a function of number of training images. An easy task is evaluated in Figure 5a and a more difficult task, where the correlation lengths are more similar, is reported in Figure 5b. IO does not depend on the number of training images because the true value of the covariance matrices are used to estimate AUC. The AUC for $IO_\mathbf{t}$ is compared for different numbers of FKT channels: $L = 4$, 40, and 400. As expected, the AUC of $IO_\mathbf{t}$ increases with the number of channels. At a given L the AUC of IO_t is lower for the more difficult task in Figure 5b as compared to Figure 5a.

The gap between the AUCs of $IO_\mathbf{t}$ for the easy and more difficult task closes as the number of channels decreases. At the maximum compression, AUC for $IO_\mathbf{t} : L = 4$ is 0.54 for the easy task and 0.52 for the more difficult task; see Figures 5a and 5b, respectively. Notably, for 100 training images $\text{mCNN}_\mathbf{g}$ and $\text{sCNN}_\mathbf{g}$ are outperformed by $\text{IO}_t : L = 40$ for the easy task and $\text{IO}_t : L = 4$ for the more difficult task. For a given L the CNN requires a greater quantity of training data to meet IO_t as the task difficulty increases.

The number of trainable parameters in the sCNN and mCNN depends on the image size, see Equation 5 and 6. Both the sCNN and mCNN have the same hyperparameters in the first layer. The number of trainable parameters for $\text{sCNN}_\mathbf{g}$ and $\text{mCNN}_\mathbf{g}$ are: $N_s = 15,745,761$ and $N_m = 3,843,533$; respectively. Detection performance depends on both the number of independent samples and the number of trainable network parameters. The number of independent samples is the product of the number of training images and the size of an image. Above 1,000 training images, the number of independent samples exceeds the number of trainable parameters in $\text{mCNN}_\mathbf{g}$ but is less then the number of trainable parameters in $\text{sCNN}_\mathbf{g}$. Also above 1,000 training images, the AUC of $\text{sCNN}_\mathbf{g}$ meets or exceeds the AUC of $\text{mCNN}_\mathbf{g}$; see Figure 5. At and below 1,000 training images $mCNN_g$ outperforms or equals $sCNN_g$ where the reduction of trainable parameters increases detection performance.



(a) $\sigma_1 = 0.38$ and $\sigma_2 = 0.30$ pixels



(b) $\sigma_1 = 0.38$ and $\sigma_2 = 0.34$ pixels

Figure 5: AUC dependence on quantity of training images for: (a) an easy task where $IO_\mathbf{g}$ $AUC = 0.98$ and (b) a task of moderate difficulty where $IO_\mathbf{g}$ $AUC = 0.88$. Here $\sigma_1 = 0.38$ pixels and in (a) $\sigma_2 = 0.30$ pixels and (b) $\sigma_2 = 0.34$ pixels. The upper bound on AUC, for a given number of channels (L), is given by $IO_\mathbf{t}$, which is always lower for the more difficult task in (b). Here the CNN performance strictly increases with quantity of training data. $mCNN_g$ outperforms $sCNN_g$ for 500 and 1,000 training images. For larger quantities of training images $sCNN_g$ performance is greater.

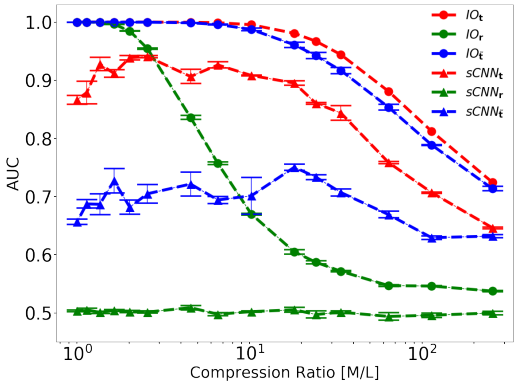
Figure 6 reports AUC as a function of compression ratio M/L for an easy task of: (a) short correlation lengths and (b) longer correlation lengths. AUC mean and standard deviation are estimated from 100 training images. IO performance is denoted by circle markers. Triangle markers denote performance of sCNN. The colors denotes linear processing methods: red ($\mathbf{t}$), green ($\mathbf{r}$), and blue ($\tilde{\mathbf{t}}$) defined in Equations 8, 9, and 10; respec-

tively. The IO is invariant to multiplication by a nonsingular matrix. Therefore, IO AUC is equal for all three full-rank compression matrices when $L = M$. As the compression ratio increases, the IO AUC decreases at different rates for each compression matrix. Compression from FKT (red) is optimal linear compression for this zero-mean Gaussian heteroscadastic data. The AUC=1.0 for the FKT compressed image (denoted by IO_t and red circles in Figure 6) until $M/L \approx 10$ where the AUC decreases monotonically with compression ratio. The performance trend of $sCNN_t$ matches IO_t for large compression ratios. However, given small compression $sCNN_t$ AUC is convex and performance peaks at $M/L \approx 6$. Further compression degrades the detection performance due to the trade-off between information content and estimation from finite sample statistics [7]. This performance peak is less pronounced in Figure 6a as compared to Figure 6b. The correlation lengths are shorter and therefore image data contains higher frequency content in Figure 6a compared to Figure 6b. Both detection tasks are of similar difficulty level given by IO AUC=1.0 for uncompressed images. However, even for uncompressed images the $sCNN_t$ AUC for these tasks deviates; see Figure 6b compared to Figure 6a. The longer correlation length structure is more challenging for the sCNN to learn given a low quantity of training data. This indicates that although IO AUC is a metric for detection task difficulty it is not necessarily a predictor of the quantity of training images necessary for a IO performance approximation. The correlation structure of the data also needs to be considered.

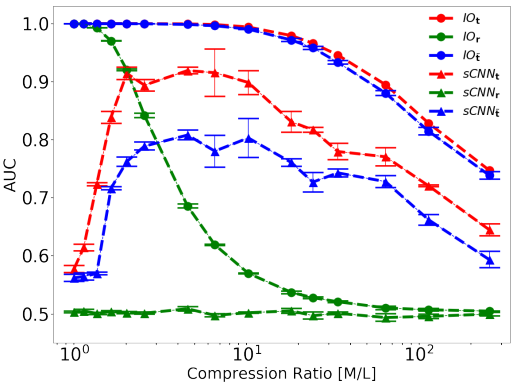
The IO AUC decreases most rapidly when no prior knowledge is used in the compression matrix IO_r (green circles in Figure 6). Compression is implemented by populating a $L \times M$ matrix with uniformly-distributed random samples; see Equation 9. Even at low compression ratios where IO_r AUC=1.0 the $sCNN_r$ AUC$\approx$ 0.50. This result is the largest disparity between CNN and IO reported in this work. The non-singular linear transform $\tilde{\mathbf{g}} = \mathbf{R}\mathbf{g}$ does not change the information content of the image data since the original image data can be recovered by $\mathbf{g} = \mathbf{R}^{-1}\tilde{\mathbf{g}}$. However, the sCNN trained on $\tilde{\mathbf{g}}$ no longer recognizes the correlation differences between the two classes.

The correlation structure change due to matrix-multiplication by $\mathbf{R}$ can be changed again by an FKT compression matrix applied to $\tilde{\mathbf{g}}$ images; see Equation 10. This is denoted $IO_{\tilde{t}}$ (blue circles in Figure 6) and closely matches the curve for IO_t (red circles in Figure 6), with a maximum AUC difference of less than 0.01 which begins to occur when $M/L = 10$. The AUC of $sCNN_t$ and $sCNN_{\tilde{t}}$ differs by ≈ 0.10 at $M/L = 1$ for the shorter correlation length images in Figure 6a. For the longer correlation length images in Figure 6b AUC of $sCNN_t$ and $sCNN_{\tilde{t}}$ differs by ≈ 0.01 at $M/L = 1$. Therefore, the effectiveness of FKT to restore a correlation structure which sCNN can recognize depends on the correlation structure of the original data set. For the shorter correlation length case an increase in FKT compression on $\tilde{\mathbf{g}}$ results in AUC fluctuating near 0.7 and peaking at $M/L \approx 200$; see $sCNN_{\tilde{t}}$ denoted by blue triangles in Figure 6a. For the longer correlation case the AUC of $sCNN_{\tilde{t}}$ is more similar to $sCNN_t$ including the performance peak observed near $M/L = 10$.

In Figure 7 the AUC of IO and mCNN are compared for an extremely easy detection task ($\sigma_1 = 1.00$, $\sigma_2 = 0.40$ in magenta) and a more difficult detection task ($\sigma_1 = 0.38$, $\sigma_2 = 0.34$ in green). See Figure 1 for example images of the extremely easy task. The mCNN performance is reported as a function of number



(a) $\sigma_1 = 0.45$, $\sigma_2 = 0.35$ pixels



(b) $\sigma_1 = 1.00$, $\sigma_2 = 0.98$ pixels

Figure 6: AUC mean and standard deviation are estimated from 100 training images as a function of compression ratio for an easy task of: (a) short correlation lengths and (b) longer correlation lengths. IO performance is denoted by circle markers. Triangle markers denote performance of sCNN. The colors denotes linear processing methods: red ($\mathbf{t}$), green ($\mathbf{r}$), and blue ($\tilde{\mathbf{t}}$) defined in Equations 8, 9, and 10; respectively. The AUC improvement with compression of $sCNN_{\mathbf{t}}$ and $sCNN_{\tilde{\mathbf{t}}}$ is less appreciable for the shorter correlation length images (a) compared to longer correlation length images (b).

of training images for $\mathbf{g}$ and $\tilde{\mathbf{g}} = \mathbf{R}\mathbf{g}$. The IO performance does not depend on number of training images or non-singular linear transform. For both the extremely easy and more difficult detection task $mCNN_{\mathbf{g}}$ improves monotonically with increased quantity of training images. Even if the correlation length differences are small the CNN is learning to distinguish the images.

For the easier task $mCNN_{\tilde{\mathbf{g}}}$ AUC (magenta circles in Figure 7) improves monotonically with increased quantity of training images. For the more difficult task $mCNN_{\tilde{\mathbf{g}}}$ AUC$\approx$ 0.5 (green circles in Figure 7) and does not improve with increased quantity of training images. This result indicates that the non-singular linear transform $\tilde{\mathbf{g}} = \mathbf{R}\mathbf{g}$ can render the correlation differences undetectable to the CNN unless the original images $\mathbf{g}$ differ in correlation length by a sufficiently large magnitude.

Conclusion

Unlike the CNN, the IO detection performance is invariant to any full-rank, invertible linear image processing and strictly de-

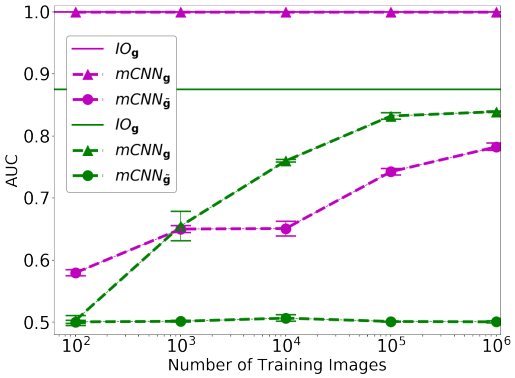


Figure 7: AUC as a function of number of training images for two classification tasks of varying difficulty. In magenta $IO_{\mathbf{g}} = 1.0$ for $\sigma_1 = 1.00$, $\sigma_2 = 0.40$ (see Figure 1 for example images) and in green $IO_{\mathbf{g}} = 0.87$ for $\sigma_1 = 0.38$, $\sigma_2 = 0.34$. Performance on image data $\mathbf{g}$ denoted by circle markers, performance on the linear transformed images $\tilde{\mathbf{g}} = \mathbf{R}\mathbf{g}$ denoted by triangles. For the easier task $mCNN_{\tilde{\mathbf{g}}}$ AUC (magenta cirlces) improves with increased quantity of training images.

creases with compression. This work demonstrates the existence of full-rank linear transforms that degrade CNN performance to nearly guessing even given large quantities of training data. A subsequent full-rank linear transform can improve the CNN performance. These results indicate the inability of the CNN to learn certain full-rank linear transforms regardless of training data quantity. For small quantities of training data, linear image compression prior to the CNN architecture can increase the AUC of the CNN discriminant. Our results demonstrate that the optimal compression ratio depends upon task difficulty, compression method, and quantity of training images.

References

[1] H.H. Barrett and K.J. Myers. *Foundations of Image Science*. John Wiley & Sons, 2013.

[2] Wilson S. Geisler. Contributions of ideal observer theory to vision research. *Vision Research*, 51(7):771 – 781, 2011. Vision Research 50th Anniversary Issue: Part 1.

[3] Weimin Zhou, Hua Li, and Mark A. Anastasio. Approximating the Ideal Observer and Hotelling Observer for binary signal detection tasks by use of supervised learning methods. *arXiv e-prints*, page arXiv:1905.06330, May 2019.

[4] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations*, 2019.

[5] Xiaoming Huo. A statistical analysis of fukunaga-koontz transform. *IEEE Signal Processing Letters*, 11(2):123–126, 2004.

[6] M. K. Kupinski and E. Clarkson. Method for optimizing channelized quadratic observers for binary classification of large-dimensional image datasets. *J Opt Soc Am A Opt Image Sci Vis*, 32(4):549–565, Apr 2015.

[7] Richard Bellman. Dynamic programming. Princeton, New Jersey: Princeton University Press. XXV, 342 p. (1957)., 1957.

[8] Abhijit Mahalanobis, Robert R Muise, S Robert Stanfill, and ALAN Van Nevel. Design and application of quadratic correlation filters for target detection. *Aerospace and Electronic Systems, IEEE Transactions on*, 40(3):837–850, 2004.

[9] David J Field. Relations between the statistics of natural images and the response properties of cortical cells. *JOSA A*, 4(12):2379–2394, 1987.

[10] Craig K. Abbey, Harrison H. Barrett, and Miguel P. Eckstein. Practical issues and methodology in assessment of image quality using model observers. *Proc. SPIE*, 3032:182–194, 1997.

[11] M.A. Kupinski, E. Clarkson, and J.Y. Hesterman. Bias in Hotelling observer performance computed from finite data. *Proc. SPIE*, 6515:65150S–65150S–7, 2007.

[12] Brandon D. Gallas. Variance of the channelized-hotelling observer from a finite number of trainers and testers. *Proc. SPIE*, 5034:100–111, 2003.

[13] Keinosuke Fukunaga and Warren LG Koontz. Application of the karhunen-loeve expansion to feature selection and ordering. *IEEE Transactions on Computers*, 19(4):311–318, 1970.

[14] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015.

[15] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, USA, 2006.

[16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

Author Biography

Meredith Kupinski received the M.S. and Ph.D. degrees in Optical Science from the University of Arizona, USA, in 2003 and 2008, respectively. She is currently a Research Professor with the College of Optical Sciences, at the University of Arizona. Her research interests include task-relevant metrics for imaging system design, estimation/detection theory, and stochastic systems analysis and information quantitation.

Luca Caucci received a B.S. in computer science from the University of Bologna, Italy in 2002, a B.S/ in mathematics from the University of Pisa, Italy in 2005, a M.S. in electrical and computer engineering from the University of Arizona in 2007, a M.S. in optical sciences from the University of Arizona in 2009, and a Ph.D. in optical sciences from the University of Arizona in 2012. His current research interests are in medical imaging and computing. Applications include task-based assessment of image quality, optimal methods for signal detection and parameter estimation, adaptive systems, and fast computational methods on parallel architectures for list-mode data processing for PET, SPECT, and CT.

Khalid Omer received a B.S. in Optical Sciences and Engineering from the University of Arizona, USA, in 2018, with a minor in Physics. He is currently a graduate student at the College of Optical Sciences, at the University of Arizona. His current research interest are in object/scene recognition utilizing neural networks.

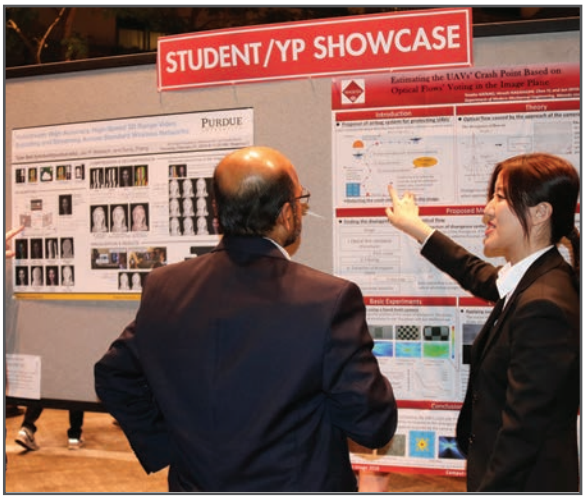
JOIN US AT THE NEXT EI!

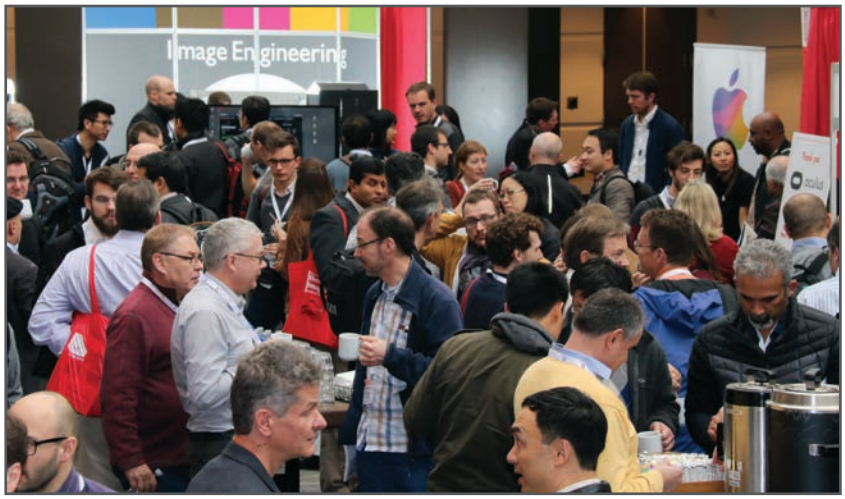
IS&T International Symposium on

Electronic Imaging

SCIENCE AND TECHNOLOGY

Imaging across applications . . . Where industry and academia meet!





• SHORT COURSES • EXHIBITS • DEMONSTRATION SESSION • PLENARY TALKS •
• INTERACTIVE PAPER SESSION • SPECIAL EVENTS • TECHNICAL SESSIONS •

www.electronicimaging.org

IS&T
imaging.org