Visible fingerprint of X-ray images of epoxy resins using singular value decomposition of deep learning features

Edgar Avalos, Kazuto Akagi and Yasumasa Nishiura[1]

Mathematical Science Group, WPI-Advanced Institute for Materials Research (AIMR), Tohoku University, Japan

[1] *Research Institute for Electronic Sciences, Hokkaido University and MathAM-OIL, Tohoku University and AIST, Japan*

Abstract

Although the process variables of epoxy resins alter their mechanical properties, recently it was found that the total variation of the X-ray images of these resins is one of the key features that affect the toughness of these materials. However it is still not clear how to visualize such a difference in a clear way. To facilitate the visualization, we use a robust approximation of the gradient of the intensity field of the X-ray images of different kinds of epoxy resins and then we use deep learning to discover the most representative features of the transformed images. In this solution of the inverse problem to find characteristic features to discriminate samples of heterogeneous materials, we use the eigenvectors obtained from the singular value decomposition of all the channels of the response maps of the early layers in a convolutional neural network. While the strongest activated channel gives a visual representation of the characteristic features, often these are not robust enough in some practical settings. On the other hand, the left singular vectors of the matrix decomposition of the response maps barely change when variables such as the capacity of the network or the network architecture change. High classification accuracy and robustness of characteristic features are presented in this work.

Keywords: Epoxy resin, X-ray CT scan, Deep learning, Convolutional neural network, Computer vision, Structure-property mapping

1. Introduction

X-ray CT imaging of polymer composites enables the non-destructive visualization of three-dimensional samples as well as two-dimensional slices of the material [1]. Despite the significative improvements in the spatial resolution, the resulting images exhibit highly fluctuating electron density patterns that are extremely difficult to discriminate by simple observation. These density patterns describe the structural heterogeneity of the epoxy resins, and the heterogeneity at the micro/meso-level has a profound relation with the macroscopic performance [2, 3, 4]. Therefore the visual identification of characteristic features in the X-ray images is desirable to understand the mechanical behavior of these materials.

The problem of grouping images of samples of materials with similar properties was recently addressed in Ref. [5], where the total variation (TV) of an X-ray image defined as $\int \sqrt{f_x^2 + f_y^2}\, dxdy$, is used to order the images into groups with similar mechanical performance. In the expression for the TV, f_x and f_y are the gradients of the intensity field along the x and y directions, respectively. Remarkably, the aforementioned work presents the TV as a property that describes the performance of materials, which is a significant step towards the solution of the inverse problem. However, while it is possible to use the total gradient to categorize X-ray images with similar properties, what is still lacking in the description of the amorphous materials is a visual fingerprint of the most representative features that can be used both for discrimination and for describing the performance of materials. A notable aspect in this work is that we succeed in providing with visual qualitative representation of key features with the aid of a neural network and singular value decomposition. In this paper we propose a solution to the inverse problem based on deep learning.

Machine learning tools are widely used to identify and categorize different samples of materials and to predict their properties [6, 7, 8, 9, 10]. Some examples of these tools include deep convolutional neural networks (CNN) [11, 12, 13, 14], which are highly accurate to classify images by minimizing a suit-

able cost function [15]. By visualizing the intermediate responses in a CNN one can have a better understanding of the features that the net uses for classification [16, 17]. Among some few applications that take advantage of this approach, we can mention Ref. [18], in which the authors use a CNN that is able not only to classify crystals with astonishing accuracy, but remarkably the activated filters of the early layers produce visual fingerprints that distinguish among different crystal symmetries. Similarly in Ref. [19], the authors use a CNN to analyze thermal images by looking at the strongest activation channel of the intermediate layers to construct a visual representation of the early signs of failure in power transformers. In another impressive application, Lakhani et al. [18] are able to visualize features on the intermediate layers in a CNN to detect pulmonary tuberculosis on chest radiographs. An additional example illustrates the use of the activations of the early layers to highlight driver behaviours [20].

The singular value decomposition (SVD) of a matrix A representing a collection of images, provides with an orthonormal basis that can be used to describe correlations between individual images in A. We use SVD to discover the features in X-ray images that are relevant for classification. Although SVD has been used to generate the input to the neural network [21, 22] and to improve the training of the networks [23, 24, 25], to the best of our knowledge, the analysis of the statistical correlations of the feature maps in the early layers of a CNN has not been utilized to discriminate X-ray images of samples of epoxy resins.

This paper is organized as follows. In Section 2 we describe the experimental data and the properties of the materials. We provide with a description of the CNN employed in this work and we highlight the importance of a robust approximation to the gradient field of the X-ray images as a preprocessing step before training the network. In Section 3 we present two approaches to visualize characteristic features in the images that are relevant for classification. One method consists in finding the response map with the strongest activation and the other method leverages the hierarchical ordering of the eigenvectors of a

whole set of response maps in a CNN to produce a visual representation of the features that are relevant for classification. We close with a brief argument to support the use of a single eigenvector to describe a library of response maps and a discussion of the advantages of the proposed approach.

2. Methods

2.1. Materials and image acquisition

In this section we briefly describe the materials used in this study. Four different samples of thermosetting resins were prepared and characterized by NIPPON STEEL Chemical & Material CO., LTD. These samples have the same chemical composition of bisphenol A type epoxy molecule and hardener molecule (primary diamine) with a ratio 3:1. Different conditions of heating temperature and heating time endow the samples with different polymerization rates, densities, and fracture toughnesses, as it is shown in Table 1. The samples were prepared as a plate with a thickness of 1mm and their mechanical properties were measured by the standard means of evaluation. The stress-strain (s-s) curve for the samples in different settings shown in Fig. 1, indicates that the sample No. 3 possesses the best performance in terms of the absorbed energy up to fracture, that is, the largest area under the s-s curve.

In addition to the mechanical tests, X-ray CT images of all samples were obtained using a commercial apparatus with the resolution of 2 μm/pixel. Two squared sections of size 266×266 pixels were cut out from left and right sides of the plate, which were then sampled into X-ray images every 2 μm (Figure 2). A total of 266 slices of each sample were extracted and thus our dataset consists of 1064 X-ray images. Some representative X-ray CT images of samples 1-4 are shown in figure 3. In these images, the bright areas correspond to regions of high electron density (≒ high atomic density). Since the three-dimensional network of covalent bonds obstructs the packing of molecules, we assume that the highly polymerized regions correspond to low density (dark) domains. On the other hand, the highly polymerized region has a larger Young's modulus, therefore,

the dark region is considered to have a larger Young's modulus than the bright region. As it was mentioned above, variations of the patterns of the intensity of the X-ray images have a significant impact on the mechanical properties of the samples, which is revealed by the finding in Ref. [5], that shows the TV (or equivalently the total gradient content of the images) is an appropriate parameter to describe the samples in terms of their mechanical performance. However, the differences are not evident by simple observation. Therefore our goal is to identify the visual fingerprints in the samples that are relevant for their classification.

Table 1: Properties of samples 1-4. Data courtesy of NIPPON STEEL Chemical & Material CO., LTD.

	Sample 1	Sample 2	Sample 3	Sample 4
Polymerization Rate [%]	13.7	45.9	60.3	90.0
Density $[g/cm^3]$	1.151	1.157	1.145	1.143
Fracture Toughness [MPa·m$^{1/2}$]	0.16	1.03	0.99	0.83

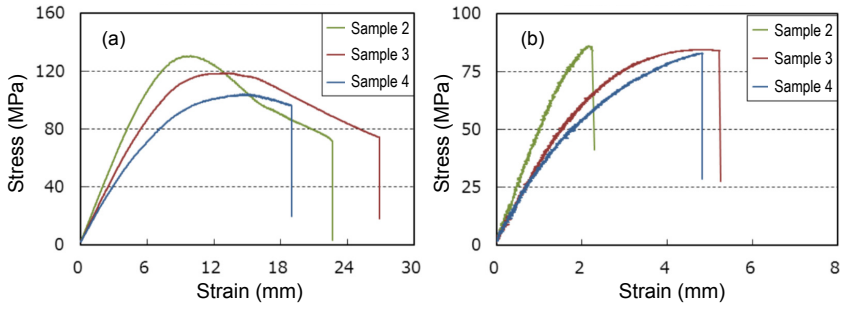


Figure 1: Stress-Strain diagram determined by (a) bending test and (b) tensile test. In both cases sample 3 has the highest total absorbed energy, this is, the area under the s-s curve. The sample No. 1 is rather fragile and could not withstand the mechanical test. Data courtesy of NIPPON STEEL Chemical & Material CO., LTD.

In section 2.3 we use a neural network to classify images like those shown in

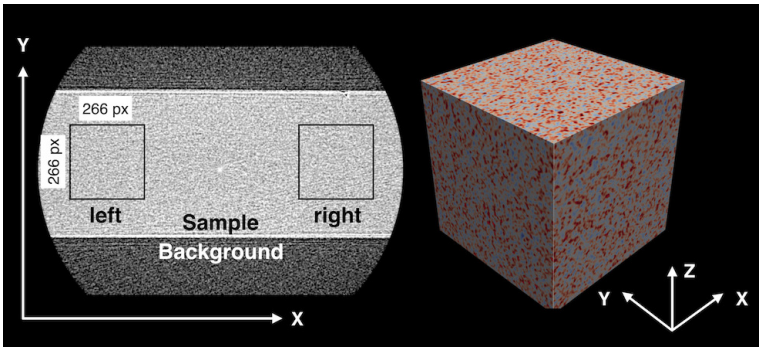


Figure 2: (Color online)(Left) An example of X-ray CT slice image. The middle bright area corresponds to the sample. The upper and lower dark background area shows a noisy image originated from the fluctuation of X-ray intensity. (Right) 3D reconstruction using 266 slices. The brighter (darker) parts are colored red (blue).

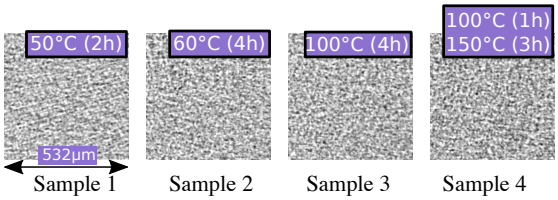


Figure 3: X-ray CT images of four samples of epoxy resins. Variations of the process variables, such as heating and heating time, produce different patterns of density in the X-ray images.

figure 3. Our goal is to discover what are the most distinctive features in the X-ray images that are used to classify these images into four classes. However, the direct use of the original X-ray images is delicate because the images contain a noisy background resulting from the measurement process. Although it is possible to use a neural network to extract the features directly from the original images of the intensity field, the results are not satisfactory, as it is shown in the first column of Figure 4, which exhibits large areas of activation with unclear indication of the features. In order to render a more faithful description of the features, it is desirable to preprocess the X-ray images to highlight more clearly the variations of the intensity field.

A first task is to determine what variable is adequate to produce a good visualization of the features of the images. It was previously found [5] that, for a given image with intensity field Y, the magnitude of the gradient, $|\nabla Y|$, seems adequate to preprocess the X-ray images because the average value of the gradient field, $\overline{|\nabla Y|}$, describes the samples 1-4 according to their mechanical performance. For example, the figure 5b shows that the quantity $\overline{|\nabla Y|}$ is larger in the sample No. 3, which also possesses the largest area under the s-s curve, as it is shown in Fig. 1. On the other hand, although $\overline{|\nabla Y|}$ allows to describe the samples according to the mechanical performance, it would be desirable to develop a method to visualize the most important features on the images for classification. The present work is an attempt to develop such method of visualization.

The gradient field is adequate to preprocess the X-ray images for several reasons. Firstly, the gradient is used frequently in industrial inspection, either to aid humans in the detection of defects or, what is more common, as a preprocessing step in automated inspection [26]. Additionally the ability to enhance small discontinuities in an otherwise flat gray field is another important feature of the gradient [27]. Secondly, although a convolutional neural network can use the original X-ray images to extract characteristic features of the images, these appear blurry and not well defined, as it is shown on the first column of Figure 4. An intuitive explanation is that while some convolutions taking place at the first layers of the neural network are approximately similar to the gradient calculation, the individual loadings responsible for the convolution are not specifically designed to compute the gradient. The loadings at the layers are meant to distinguish different kinds of features on the images and are optimized during the network training, but they are not intended to function as a robust discretization of the gradient operator. In contrast, a good gradient operator is designed to produce a robust description of the variations of the intensity field. In the next section we employ a gradient operator that is robust under noisy conditions. Thirdly, the evidence shows that the intensity field is not the appropriate variable to describe the mechanical performance. Previously

it has been show that the intensity does not provide the correct sequence of the projections of the X-ray images onto a eigenspace spanned by the first two principal components. However, the projections of the gradient images, $|\nabla Y|$, have the correct sequence, in which sample No. 3 has the best mechanical performance [5]. Computing the gradient images allows us to correctly describe the material performance of samples 1-4.

2.2. Preprocessing

2.2.1. Basic statistical quantities.

We look into some basic statistical parameters in the set of the X-ray images. To analyze the distribution of the pixel values in the X-ray images, we proceed as follows. For each grayscale X-ray image with pixels taking values between 0 and 255, we compute the mean value in a fixed squared domain (266×266). The total intensity of the images divided by the pixel count corresponds to the mean value of the images. Figure 5a shows the mean value of the slices of samples 1 to 4, and Figure 5b shows the mean of the module of the gradient of each slice computed as $\frac{1}{N}\int (f_x^2 + f_y^2)^{1/2} dxdy$, with N being the pixel count. Notice that sample No. 3 has the largest average value of the module of the gradient.

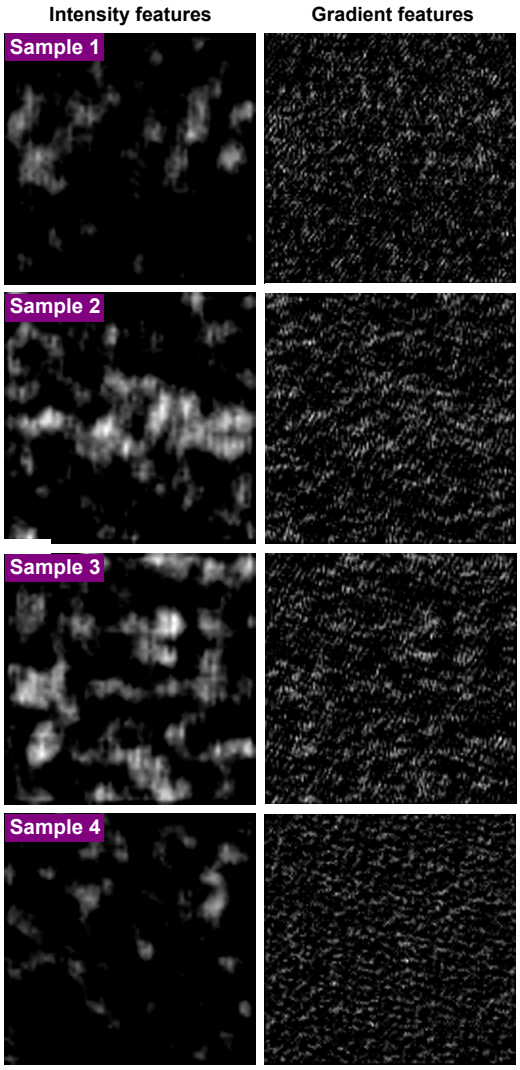
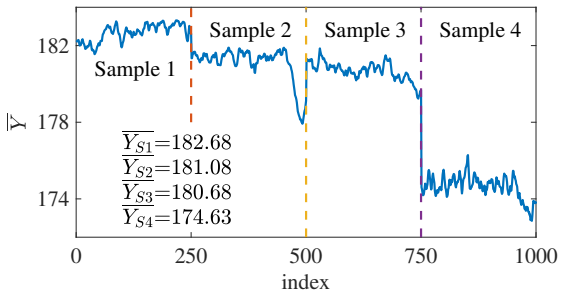
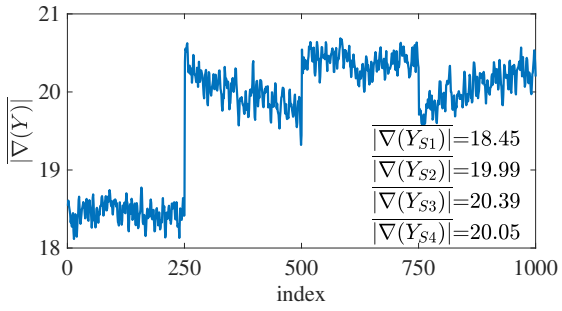


Figure 4: Feature maps of samples 1-4 (top to bottom) obtained by training the CNN described in section 2.3 and then the features of a test image of samples 1-4 are extracted. The first column shows results obtained directly using the original X-ray images of the intensity field, Y. Notice that the features lack of well-defined boundaries and occupy large areas of the domain. The preprocessing presented in section 2.2.2 significatively improve the visualization of the features rendering more accurate responses. The second column shows results obtained from transformed X-ray images, ∇Y, using the strongest channel, α, as presented in Eq. (13) in section 3.2. The features vary considerably when the initial conditions of the training are changed. In section 3.3 we develop a more robust method to extract features. Compare these feature maps with the richer and more accurate details shown in Figure 15.



(a) Mean of each slice.



(b) Module of the gradient

Figure 5: (a) Mean value of the intensity field of each X-ray image shown as $\overline{Y_{Si}}$. The index number represents each image and runs from 1 to 1000. From left to right, the vertical lines separate samples 1 to 4. (b) Mean of the module of the gradient of each slice, $\frac{1}{N}\int(f_x^2 + f_y^2)^{1/2}dxdy$, with N being the pixel count. Notice that sample No. 3 has the largest average value of the module of the gradient.

2.2.2. Robust gradient approximation

The original X-ray images describing the intensity field contain large fluctuations of intensity and it is required to remove all unnecessary information from the images. Transforming the intensity field into the module of the gradient highlights the features of the X-ray images. More importantly, the summation of all the local contributions of the module of the gradient –the total variation– is a quantity related to the mechanical performance of the samples [5]. The role of this transformation is additionally strengthened by the fact that the module of the gradient produces the correct classification of the images, which places the sample No. 3 as the one with the best mechanical performance [5].

To compute the module of the gradient we process as follows. For each image Y defined as a $m \times n$ array, we need to compute the derivatives in the vertical and horizontal directions, $\frac{\partial Y}{\partial x}$ and $\frac{\partial Y}{\partial y}$, respectively. The module of the gradient is then defined as

$$\nabla Y = \begin{bmatrix} Y_x \\ Y_y \end{bmatrix} = \begin{bmatrix} \frac{\partial Y}{\partial x} \\ \frac{\partial Y}{\partial y} \end{bmatrix} \tag{1}$$

$$|\nabla Y| = \sqrt{Y_x^2 + Y_y^2}, \tag{2}$$

where Y_x and Y_y represent simple discrete approximations of the forward difference for interior data points in the the vertical and horizontal direction, respectively. For example, for a matrix with unit-spaced data, Y, that has vertical gradient, Y_x, the interior gradient values are computed as $(f_{i,j+1} - f_{i,j})$. The horizontal gradient is computed similarly.

The calculation of the module of the gradient of each image Y shows that the transformed image $|\nabla Y|$ reveals previously unseen intricate variations of the intensity field as it is shown in figure 7. Additionally, it is well-know that noise suppression is an important issue when dealing with derivatives to compute the gradient [27]. Therefore the discretization of the gradient function requires special attention. Although simple algorithms of differentiation such as central difference and forward difference produce good results of clustering [5], these methods are not longer satisfactory to visualize the gradient of an image

with noisy background. A more comprehensive computation of the gradient is necessary to capture a richer content of visual details in the image of $|\nabla Y|$.

In addition to forward differences, there are other methods to approximate the gradient. More in general, the gradient of a given image Y is computed through a 2D convolution with a 3×3 mask Z, as shown in Eqn. 3.

$$C(x, y) = \sum_{t=-1}^{1} \sum_{s=-1}^{1} Z(s, t) Y(x - s, y - t) \tag{3}$$

Figure 6 shows several masks to perform the approximation of the derivatives needed for the gradient operator. While the forward difference approximation shown in Figure 6b preserves clusters in an eigenspace [5], the contribution of additional neighbouring locations improves the gradient approximation. The Prewitt mask shown in Figure 6c, produces a more comprehensive value of the gradient that includes neighbouring contributions of a given location. Additionally, the Sobel mask shown in Figure 6d gives more weight to the central pixel and also has better noise-suppresion (smoothing) characteristics [27]. For a given image Y, the derivatives Y_x and Y_y can be computed using Eqn. 3 with the Sobel masks shown in Figure 6d, and then $|\nabla Y|$ can be found from Eqn. 2, which is the absolute gradient field of the image Y.

As a visual example, Figure 7 shows $|\nabla Y|$ for a typical image of sample 1 computed using forward difference and a Sobel mask. Notice that additional details are captured by the Sobel mask. In what follows, we use the Sobel operator to transform the intensity field of the whole set of the X-ray images into the corresponding absolute gradient field. The goal is to discover the most notable features that are relevant to classify the images.

2.3. Convolutional neural network

2.3.1. Architecture description

A CNN is a type of neural network [28, 29] that uses convolution (Eqn. 4) to process 2D numerical arrays such as images. Figure 9 shows a schematic representation of the CNN used in this work. The CNN is composed of several convolutional blocks including convolution, batch normalization, rectified linear

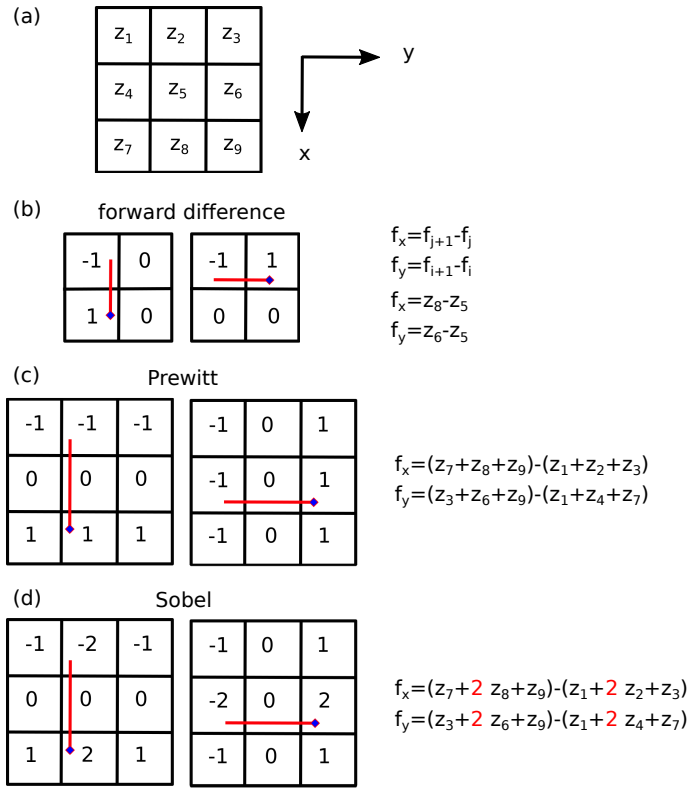
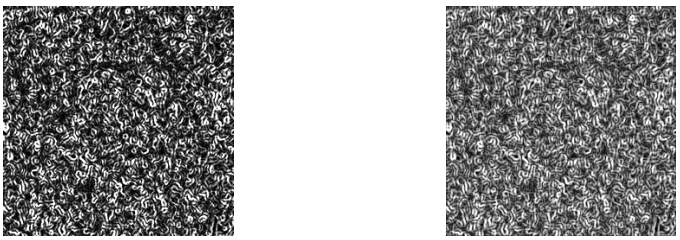


Figure 6: Filter masks to compute the derivatives needed for the gradient operator. (a) Generic 3×3 mask with the ordering of its elements shown as z_p, with $p = 1, .., 9$. (b) Simple forward approximation, (c) Prewitt mask and (d) Sobel mask. Notice this mask gives twice the weight to the central pixel compared to Prewitt.

unit (ReLU) and maxpooling. The final end of the CNN consists of one fully-connected layer and one Softmax layer. The two-dimensional convolution of the input Y with a kernel W, denoted as $C(m,n) = Y(m,n) * W(m,n)$, is defined as

$$C\left(m,n\right) = \sum_{t=\infty}^{\infty} \sum_{s=-\infty}^{\infty} W\left(s,t\right) Y\left(x-s, y-t\right) \tag{4}$$

Where m and n are indices along the horizontal and vertical directions, respectively. The resulting function $C(m,n)$ is the output or feature map of the convolution. For a given image y_i at the l-th layer, the convolution produces



(a) $|\nabla Y|$ computed using forward difference. (b) $|\nabla Y|$ computed using a Sobel mask.

Figure 7: Image of module of gradient of a typical sample using two different discretization methods. The intricate texture of the images is the result of the local variations of the intensity field captured by the transformation.

the output

$$s_j^l = f\left(y_i^{l-1} * w_{ij}^l + b_j^l\right) \tag{5}$$

Where w_{ij}^l are learnable parameters, b_j^l is a bias and $f(\cdot)$ is the output of the activation function [30]. We implement batch normalization [31, 32, 33] right after each convolution and before activation by calculating the mean and standard deviation of each input variable to a layer per mini-batch, which consists of a subset of the training dataset. We employ a mini-batch mean μ_B and standard deviation σ_B^2 as:

$$\hat{x_i} = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \tag{6}$$

A small value of ϵ is needed in case the mini-bach variance is very small. The rectified linear unit (ReLU) defined in Eqn. 7 [34, 35], is employed as nonlinear activation function because it is faster than other alternatives such as the sigmoid or $\tanh(\cdot)$ functions. What the ReLU function does is to clip the negative part out of the output of the activations of the convolution.

$$g(z) = max\left(0, z\right) \tag{7}$$

Maxpooling is employed on the activations after the ReLU function to downsample the spatial size of the input arrays by taking the maximum values of a

subset of the input array [36].

Five convolutional blocks are employed in sequence and then the fully connected layer executes a linear combination of the features of the output of the previous layer as:

$$h_j^l = f\left(\sum_i \left(x_i^{l-1} w_{ij}^l\right) + b_j^l\right) \tag{8}$$

In this case l denotes the l-th layer, b_j is a given bias, w_{ij} is the ij-th weight between the input x_i and the j-th output unit h_j. In our case j =1,2,3, and 4. To normalize the output of the fully connected layer in the range $[0, 1]$, we use the softmax function (Eqn. 9) [37]. The values of softmax$(z)_i$ represent probabilities of the classes i =1,2,3, and 4. The total sum of the probabilities of the four classes is 1.

$$\text{softmax}(z)_i = \frac{e^{z_i}}{\sum_j e^{z_j}} \tag{9}$$

We initialize the training with random values of the weights and then run the entire data set of images for several epochs. The classification error for multiple samples and multiple classes is computed using cross-entropy [15] as loss function, J, as

$$J = -\sum_{i=1}^{N}\sum_{j=1}^{K} t_{ij} \ln y_{ij} \tag{10}$$

Where t_{ji} indicates the i-th image belongs to the j-th class and y_{ij} is the i-th prediction obtained for class j from the softmax activation (Eqn. 9). We want to minimize the cost function with respect to all the parameters w_{ij}^l in the model as

$$\arg\min_{w_{ij}^l} \quad J \tag{11}$$

The architecture of the convolutional neural network used in this work is detailed in Table 1.

Training was performed using stochastic gradient descent [32] to minimize the loss function (Eqn. 10) in such a way that the weights of both the convolutional and fully-connected layers are updated to reduce the error. To evaluate the gradient of the loss function and update the weights, we use batches of 64 images and a learning rate is set equal to 10^{-2}. The validation frequency

is equal to five iterations. Using these values, the classification error steadily decreases to it smallest value. Recent results shows that mini-batches of small size improved training performance and allow a significantly smaller memory footprint [38, 39]. We confirmed that using a mini-batch of size 32 or 64 produces similar feature maps. To prevent overfitting, we have employed a dropout layer [40] at the fully connected layer to randomly set input elements to zero with a given probability of 0.5. Adding a ℓ_2-regularization term for the weights to the loss function also helps to reduce overfitting [37, 41]. The loss function with the regularization term takes the form

$$J_R = J + \lambda\Omega(w) \tag{12}$$

Where w is the weight vector, λ is the regularization factor and $\Omega(w) = \frac{1}{2}\|w\|^2 = \frac{1}{2}w^T w$. For $\lambda > 0$, we minimize J_R as indicated in Eqn. 11. Dropout combined with ℓ_2-regularization gives a lower classification error [40].

Image preprocessing and training procedure. We use grayscale 8-bit images with pixel intensities taking values from 0 to 255. The set of input images consist of 1064 image files stored in a TIF format with the resolution of 266×266 pixels. To have the data dimensions of approximately the same scale, we normalize the images by dividing each image by its standard deviation once it has been zero-centered. Zero-centering means subtracting the mean from each image. Data augmentation is a powerful method to reduce both the validation and training errors by artificially in enlarge the training dataset size by data warping. The augmented data represents a more comprehensive set of possible data points, thus minimizing the distance between the training and the validation set [42]. We employ mirror and upside-down transformations to augment our data base. The images were grouped in four sets, each with 266 images and labels of four classes were assigned to all the images. For each epoch the training set was randomly divided into 2 groups, one data set with 70% of the images for training and the reminding 30% for validation. The validation data is shuffled before each network validation. Over 98% cross-validation accuracy was achieved. Figure 8 shows the training progress. The network was trained

using a CUDA enabled Nvidia Quadro GP100 GPU.

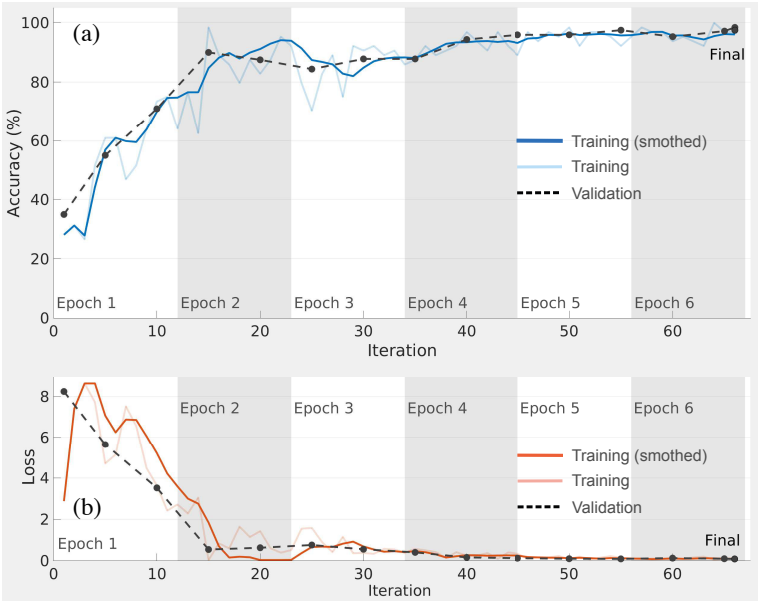


Figure 8: Training progress of the CNN using the hyperparameters described in the text. Using a mini-batch size of 64, it takes 11 iterations to all of the training samples pass through the learning algorithm. (a) Training and validation accuracy reaches more or less the same high values after 60 iterations. (b) The steady decrease of the training and validation loss functions suggests that there is not significative overfitting.

3. Results

3.1. Inverse problem

The inverse problem consists in finding features in the X-ray images that are useful to distinguish among samples 1-4. To solve this problem we choose to employ a CNN because these computing systems are able to classify images with high accuracy [17, 43]. We want to analyze the most representative features that a CNN uses to classify X-ray images. Although CNN's typically achieve a high accuracy in classification, a deep learning solution to the inverse problem is not easy to interpret because the number of parameters involved in the classification

Table 2:

Arquitecture of the convolutional neural network	
Layer type	**Specification**
Convolutional layer	(Kernel: 3 × 3; 128 filters)
ReLU	
Max pooling layer	(Pool size: 2 × 2; stride: 2 × 2)
Convolutional layer	(Kernel: 5 × 5; 128 filters)
ReLU	
Max pooling layer	(Pool size: 2 × 2; stride: 2 × 2)
Convolutional layer	(Kernel: 7 × 7; 64 filters)
ReLU	
Max pooling layer	(Pool size: 2 × 2; stride: 2 × 2)
Convolutional layer	(Kernel: 9 × 9; 64 filters)
ReLU	
Max pooling layer	(Pool size: 2 × 2; stride: 2 × 2)
Convolutional layer	(Kernel: 11 ×11; 64 filters)
ReLU	
Max pooling layer	(Pool size: 2 × 2; stride: 2 × 2)
Fully connected layer	(size: 4)
Softmax	

process is of the order of millions. However a simple approach is to look at the early layers in the network. The filters of the first few layers of a CNN are relatively easy to understand as their primary purpose is to detect simple details in the images such as edges. These edges and other low level features at the early layers of a CNN describe regions in the images that are important for classification. In order to achieve a correct classification, the loadings of the filters of the CNN are optimized during the training process. In this section, firstly we train a CNN to classify the transformed images and secondly we extract the learned features at the early layers of the net. The schematics of the net used in this work is shown in figure 9 and the details of each layer are shown in Table 2.

A simple idea to address the inverse problem is to feed a gradient image $|\nabla Y|$ into a trained CNN and then look at the responses at the early layers [17]. The hope is that such responses reveal the presence of variations of the gradient images across the domain. These responses at the early layers of a CNN show

lumps or spots where the gradient is large and these regions are important for classification.

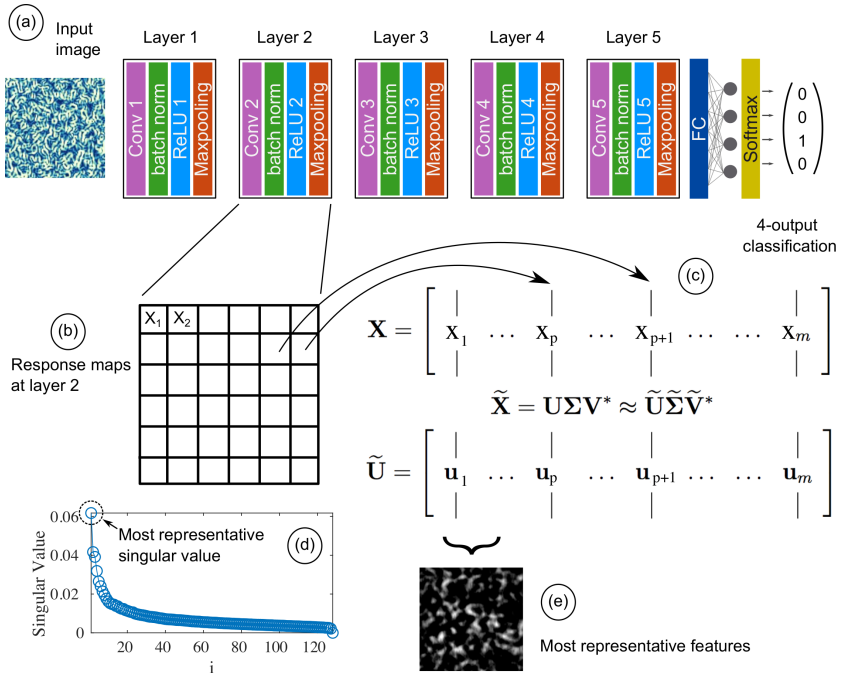


Figure 9: Schematics of the CNN used in this work. (a) The gradient of a test image $|\nabla Y|$ is the input of a trained CNN. Each convolutional block consists of convolution, max-pooling, and ReLU. A fully-connected layer with four outputs at the end. (b) The second convolutional layer contains 128 channels of feature maps. Each feature map X_i arranged as a column vector goes into a matrix $\mathbf{X}$. (c) SVD of $\mathbf{X}$ produces a left singular eigenvector U whose first column u_1 contains the most representative features of $\mathbf{X}$. (d) The fast decay of the singular values indicates that $\mathbf{X}$ has its larger correlation along the u_1 eigendirection. Singular values are obtained from the singular value decomposition of the matrix $\mathbf{X}$. (e) The first eigenvector u_1 reshaped into an image with size equal to a feature map, X_i, represents the eigenfeatures of the input test image.

3.2. Feature maps of a convolutional neural network

We examine the responses of different layers of the network summarized in figure 9 and discover which features the network learns by comparing areas of activation with the original image. Channels in early layers learn simple

features such as edges and small hallmarks, while channels in the deeper layers learn complex features. We focus only on the early layers of the network.

We consider the first two layers in a CNN, namely, Conv 1 and Conv 2. Each layer consists of 3×3 and 5×5 filters, respectively, and each of these two layers contain 128 channels of feature maps. Activation functions placed after each convolutional layer, are responsible for transforming the summed weighted input from the node into the activation of the node. We use the Rectified Linear Unit (ReLU) as activation function. Convolutional layers Conv 1 and Conv 2 are followed by activation functions ReLU 1 and ReLU 2, respectively. Each channel in the convolutional layers contains a feature map that encodes different responses. For instance, they may encode diagonal or horizontal edges. Figure 10 shows all the features maps at ReLU 2. It is necessary to select a channel that illustrates the learned features at this layer and a common choice is the channel with the strongest activation [19, 20, 44]. This figure contains 128 features maps arranged in a 12×11 array. The last four squares are empty. To understand the size of one feature map, we recall that the maxpooling operation halves the features map from 266×266px down to 133×133px. Then at the second convolutional block the activations at ReLU 2 will have a size equal to $W_2 \times W_2$ with $W_2 = (W_1 - F + 2P)/S + 1$. Where $W_1 = 133$ is size of the input, $F = 5$ is the spatial extent of the filter, $P = 1$ is amount of zero padding and $S = 1$ is the stride. Then each feature map at ReLU 2 have size 131×131px and subsequently is rescaled in the range 0 to 1 and resized to match the size of the test image, 266×266 px. To improve the contrast of each image in the collage, we have stretched the range of intensity values of each image to span a desired range. We saturate the upper 1% and the lower 1% of all pixel values. This enhancement is only for visualization purposes. For the calculations we employ the original unsaturated image of each feature map.

To select the channel with the strongest activation we iteratively search for the strongest activation among all the features maps in the responses at a particular layer. For two-dimensional feature maps of size $m \times n$ and k channels,

the strongest channel is identified with index α is obtained as follows:

$$\alpha = max(f(x, y, l) : x \in 1..m, y \in 1..n, l = 1..k) \tag{13}$$

The figure 11, on the right hand side, shows form top to bottom the feature map with the strongest activation, α, at Conv 1, Conv 2 and ReLU 2, respectively. The left hand side in all cases is the gradient image, $|\nabla Y|$. To simplify the notation we will refer to this quantity as $X = |\nabla Y|$. Notice that the responses at Conv 1 look quite similar to the original image of X, which means that Conv 1 is fundamentally acquiring the basic shape of the features of the images.

Conv 2 learns features that are slightly larger in size than those in Conv 1, because the filters in Conv 2 are slightly larger than those in Conv 1 as well. Notice that Conv 2 reveals zones where the concentration of X is large. These zones contains lumps of intensity that can be more easily seen at the activation function of Conv 2, which is called ReLU 2. The ReLU function simply clips any negative activation from the output of the second convolutional layer.

The bottom row in Figure 11 shows a comparison of the original image of X and the α channel at ReLU 2. The bright lumps highlight zones of large concentration of X. One can repeat this process for the whole set of the X-ray images and then compute the mean value of the activations. The result is shown in Figure 12. This figure shows the mean value of the α channel at ReLU 1 and ReLU 2. Notice that the ordering of these responses agrees with what was found in Ref. [5], as expected. In other words, the mechanical performance is high in materials with large concentration of the magnitude of the gradient.

Figure 13 is a solution to the inverse problem in which the features in samples 1-4 can be easily distinguished by eye. Notice that sample No. 3 contains a large concentration of activations. In this figure we employed an augmented data base of images, which is constructed by mirror and upside-down reflections on the horizontal and vertical directions, respectively. Using an augmented data base artificially enlarges the training set producing a clearer distinction of the learned features.

One of the downsides of using the channel with the strongest activation as criterion to select the most representative features of each sample is that the channel is selected by employing a single activation of the image domain. While this criterium is satisfactory for images of well-localized objects such as faces, for instance, it turns to be not necessary the best alternative available for images with features distributed on the entire domain. Specifically, a single strong activation on a feature map can wrongly identify a channel as possessing the most representative features. To assess whether the selected channel is actually the most representative one, we performed several realizations of the feature extraction process. Our results suggest that the learned features at individual layers can be sensitive to the initial values of the weights, as it shown on the second column of Figure 4. This figure shows features maps that vary considerably when using different initializations of the weights. Compare this figure with the feature maps in Figure 13. This lack of statistical robustness prompts us to we propose a more robust approach in section 3.3.

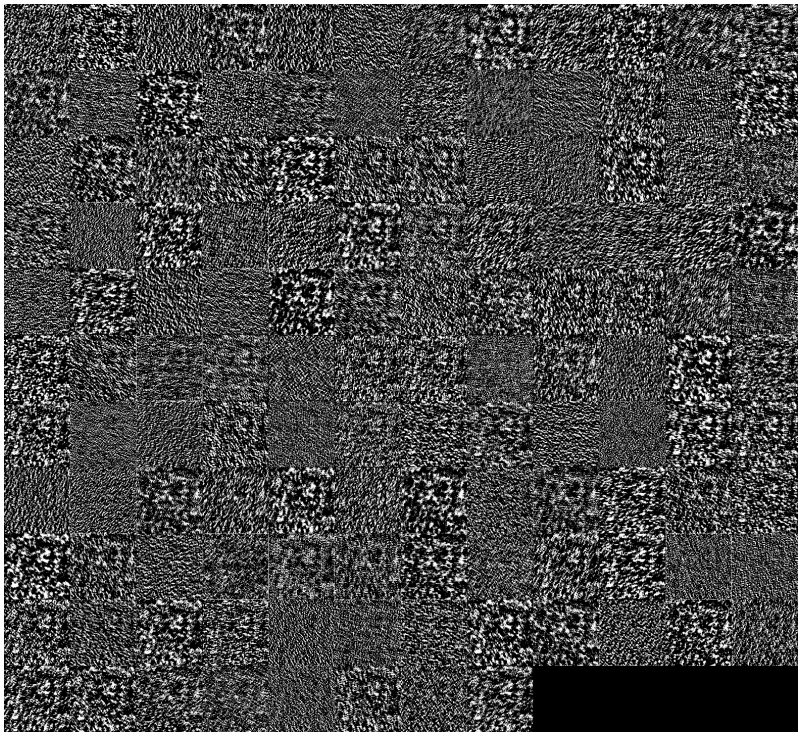


Figure 10: Feature maps of the 128 channels in the second convolutional layer. The channels are the response to the classification process of a test image of sample No. 3. The responses at ReLU 2 are shown. For visualization purposes, the 2% of all pixel values have been saturated. For the calculations, we employ the original intensity values of each feature map. The features maps are arranged in a 12×11 array. The last four squares at the bottom are empty.

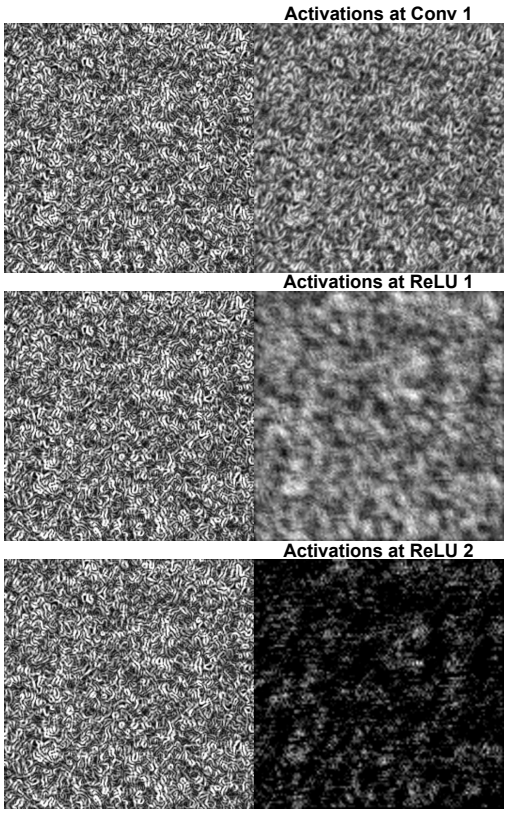
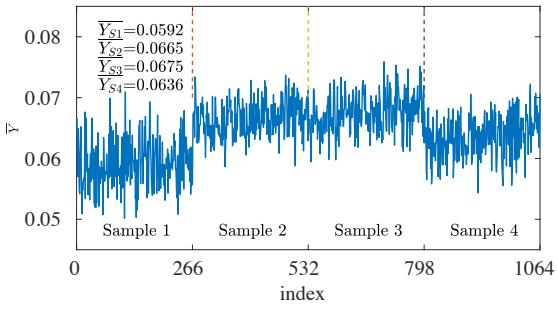
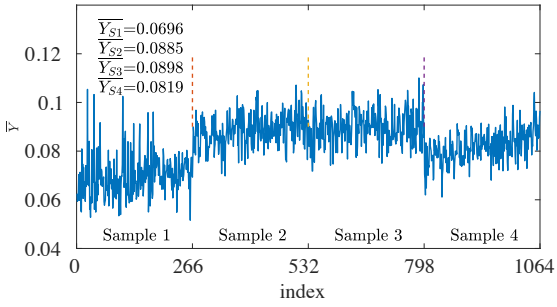


Figure 11: Comparison of the test gradient image, $X = |\nabla Y|$, of sample No. 3 (left hand side) and the channel with the strongest activation (right hand side) at different layers of the network. From top to bottom, the responses at layers Conv 1, Conv 2 and ReLU 2, respectively.



(a) Mean value of activations at ReLU 1



(b) Mean value of activations at ReLU 2

Figure 12: Average values of the positive activations at the first (a) and second (b) convolutional layers. Notice that the average values have the same ordering in (a) and (b) and that sample No. 3 has the largest average activation value in both cases.

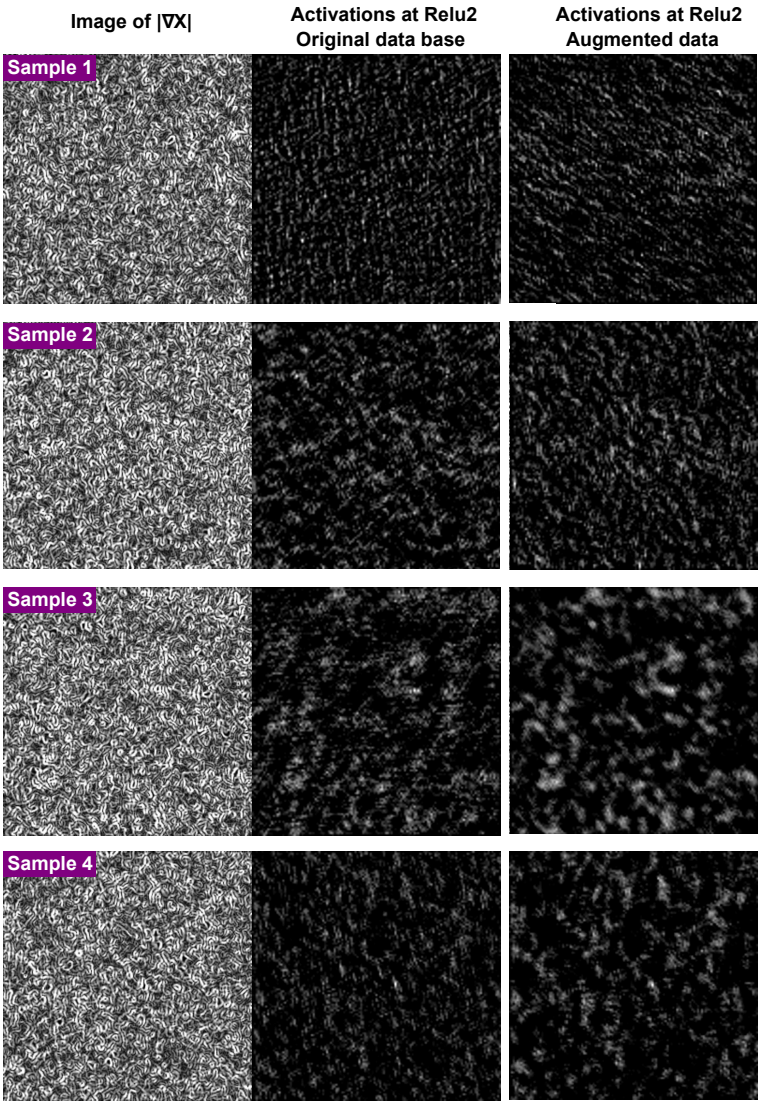


Figure 13: Features of the test images of samples 1-4 (top to bottom) obtained using the channel with the strongest activation, α, at ReLU 2. The α channel shows important features in the images for classification. First column shows test gradient images, $X = |\nabla Y|$. Second and third column show features of the images of the first column obtained using (i) original data base and (ii) the augmented data base, respectively. Augmentation is based on mirror and upside-down transformations. Notice that the responses of sample No. 3 are notably more prominent than those in the other samples, as expected.

3.3. Deep learning eigenfeatures

In this section, instead of using the α channel, we consider simultaneously all feature maps in a convolutional layer. To discover the most representative features hidden in the feature maps, one idea is to assemble a matrix containing all the feature maps and then extract the most representative features of this matrix. SVD provides a convenient way to organize the feature maps into hierarchically ordered contributions. We briefly describe the SVD method in the next section.

3.3.1. Singular value decomposition

SVD is a machine learning tool that has extraordinary applications [45, 46, 47, 48, 49]. We are interested in analzying a data set $\mathbf{X} \in \mathbb{R}^{n \times m}$.

$$\mathbf{X} = \begin{bmatrix} | & | & & | \\ X_1 & X_2 & \dots & X_m \\ | & | & & | \end{bmatrix} \tag{14}$$

In our case, the columns $X_k \in \mathbb{R}^n$ are individual feature maps at a layer of the CNN. We consider the activations at ReLU 2. The size of the response of this layer is resized to match the input image or 266×266 and thus we arrange each of these responses into column vectors with $n = 70756$ elements. Each feature map has index $i = 1, 2, ..., m$, with m being the total number of feature maps. In our case $m = 128$ because the second convolutional layer of the CNN has 128 channels.

The SVD is a unique matrix decomposition that exist for every matrix $\mathbf{X}$:

$$\mathbf{X} = U\Sigma\mathcal{V}^\top \tag{15}$$

with U and $\mathcal{V}$ being the left and right singular vectors, respectively. The symbol $\top$ indicates transpose. The details of SVD can be found in the literature [50]. For the purpose of this work, suffice it to say that SVD is a matrix factorization into unitary matrices U and $\mathcal{V}$ with orthonormal columns that are ordered hierarchically according to their importance, and Σ is a diagonal matrix containing the singular values.

A convenient statistical interpretation of the SVD involves the correlation matrix $\mathbf{X}^\top\mathbf{X}$ defined as

$$\mathbf{X}^\top\mathbf{X} = \begin{bmatrix} X_1^\top X_1 & X_1^\top X_2 & \dots & X_1^\top X_m \\ X_2^\top X_1 & X_2^\top X_2 & \dots & X_2^\top X_m \\ \dots & \dots & \dots & \dots \\ X_m^\top X_1 & X_m^\top X_2 & \dots & X_m^\top X_m \end{bmatrix} \tag{16}$$

where each entry $X_i^\top X_j = \langle X_i, X_j \rangle$ represents the inner product between columns i and j. In other words, $X_i^\top X_j$ accounts for the overlapping between all pairs of columns. This matrix accounts for the correlation between all of the feature maps in $\mathbf{X}$.

To bring the SVD into play, notice that eqn. 15 allows us to write $\mathbf{X}^\top\mathbf{X} = \mathcal{V}\Sigma^2\mathcal{V}^\top$. Similarly, $\mathbf{X}\mathbf{X}^\top = U\Sigma^2U^\top$. These expressions can be written as the following eigenvalue problems:

$$\begin{aligned} \mathbf{X}^\top\mathbf{X}\mathcal{V} &= \mathcal{V}\Sigma^2 \\ \mathbf{X}\mathbf{X}^\top U &= U\Sigma^2 \end{aligned} \tag{17}$$

It is clear from eqn. 17 that the columns of $\mathcal{V}$ and U are eigenvectors of the correlation matrices $\mathbf{X}^\top\mathbf{X}$ and $\mathbf{X}\mathbf{X}^\top$, respectively. Loosely speaking, since the columns in U are ordered according to their importance, then the data in $\mathbf{X}$ has its larger correlation along the u_1 eigendirection. Where u_1 is the first column of the left singular eigenvector U. These eigenvectors define directions along which all the feature maps in $\mathbf{X}$ have the largest variance. In this context, the eigenvectors are the principal component directions PC_i.

3.3.2. Eigenvectors of the feature maps

The process of feature discovery of a test image is summarized in figure 9. To obtain the most representative features of a test image of the absolute gradient of a sample of material, firstly we feed the test image into an already trained CNN with the architecture defined in Table 2. Secondly, we collect all the channels containing the feature maps of the test image and use each feature map as a column vector X_i for the matrix $\mathbf{X}$ in eqn. 14. We then apply the

SVD method to the whole library of feature maps obtained at ReLU 2, which follows the second convolutional layer. The feature maps can be projected onto the subspace spanned by the eigenvectors to obtain the weights of the linear combination of eigenvectors needed to reconstruct them. The fast decay of the singular values shown in Figure 9d suggests that the first eigenvector, u_1, can be used to describe the most relevant features at the second convolutional layer. The eigenvector u_1 represents the direction in which $\mathbf{X}$ has the largest variance. In terms of visual representation, the bright regions of u_1 exhibit the greatest variance and, therefore, contain the most representative characteristics of the entire library of feature maps at a given layer. The eigenvector u_1 is chosen because it represents the most significant components of a test image. Notice that since u_1 is computed from the features map, this quantity represents qualitatively the content of TV on the images.

We apply the above process to extract the deep learning eigenvectors of typical images of samples 1-4 and the result is shown in the second column of Figure 15. Notice that the eigenvectors are similar in appearance to the α channel shown in Figure 13 from the previous section. The deep learning eigenvectors posses the notable difference of being statistically robust in the sense that they seem to be less sensitive to the initial values of the weights, which is a desired property.

It is crucial to verify that the eigenvectors associated to the samples are correct and robust. This is specially necessary because of the limited size of our data base, although we have employed augmentation to enlarge the data base. To this end, we have employed several approaches, namely weight regularization and addition of dropout to the fully-connected layer. Another technique to make sure that overfitting is not significative consists in reducing the capacity of the network [17]. One way to implement this in practice is to compare models having different numbers of hidden units [15]. We assessed this idea in our model by removing the last convolutional block and additionally we substantially decreased the number of learnable parameters by halving the number of channels in each one of the first two layers, thereby leaving the first two convolutional

layers with 64 channels each. Figure 14 shows the training progress of the simpler model with reduced capacity, in which we have used a mini-batch of size 32. The steady decrease of the training and validation loss, suggests that there is no significative overfitting and the final accuracy is similar to that of the original model. We observed no significative reduction in classification accuracy and more importantly, the eigenvectors in samples 1-4 remain more or less unchanged, as show in the third column of Figure 15. A strong correlation coefficient close to $r = 0.9$ between the features obtained using the full model and the model with reduced capacity indicates that the simpler network model captures very well the most representative features for classification.

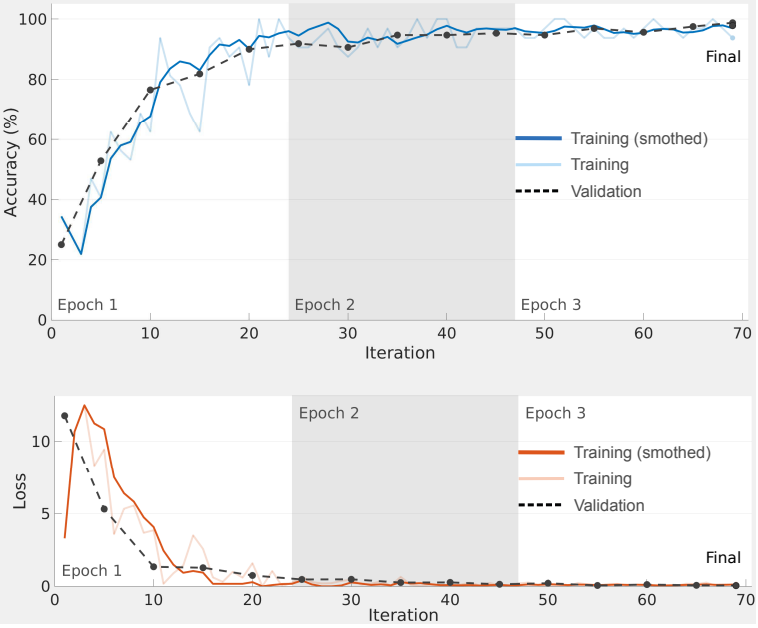


Figure 14: Training progress of the CNN with reduced capacity using the hyperparameters described in the text. Using a mini-batch size of 32, it takes 23 iterations to all of the training samples pass through the learning algorithm. (a) Training and validation accuracy reaches more or less the same high values after 70 iterations. (b) The steady decrease of the training and validation loss functions suggests that there is not significative overfitting.

4. Discussion and conclusions

A highly accurate CNN classifies images of the magnitude of the gradient of X-ray of samples of epoxy resins. The feature maps of the intermediate layers contain the most representative low-level features in a test image. The strongest activated channel produces an image of these features providing us with a simple visual summary of a given sample. However different realizations of the classification process result in slightly different features. For images with well-defined segments, such as faces or objects, this method can be considered adequate. In the case of X-ray images of resins, the descriptive features are distributed throughout the whole domain and therefore a better visualization of the representative features is desirable.

SVD provides a means to expand a matrix of feature maps, $\mathbf{X}$, in terms ordered hierarchically as:

$$\mathbf{X} = \sigma_1 u_1 v_1^\top + \sigma_2 u_2 v_2^\top + \ldots + \sigma_m u_m v_m^\top \tag{18}$$

Truncating the sum in eqn. 18 to include only the fist r terms results in the matrix $\widetilde{\mathbf{X}} = \widetilde{U}\widetilde{\Sigma}\widetilde{\mathcal{V}}^\top$. The Eckard-Young theorem [51] guarantees that the best approximation to $\mathbf{X}$ of rank-r is $\widetilde{\mathbf{X}}$, according to

$$\underset{\widetilde{\mathbf{X}} \text{ s.t. } \operatorname{rank}(\mathbf{X})=r}{\arg\min} \|\mathbf{X} - \widetilde{\mathbf{X}}\|_F = \widetilde{U}\widetilde{\Sigma}\widetilde{\mathcal{V}}^\top \tag{19}$$

Where $\|\cdot\|_F$ is the Frobenius norm. Then the best approximation to $\mathbf{X}$ that has rank $r = 1$ is given by

$$\widetilde{\mathbf{X}} = \sigma_1 u_1 v_1^\top \tag{20}$$

Where u_1 is the first column in the left singular eigenvector which has the same size as the feature maps X_i. Therefore, the eigenvector u_1 is an optimal representation of the rank-1 truncation of the whole library of feature maps and contains the most representative features of the original test image of a given sample. The one-term approximation to any feature map in $\mathbf{X}$ is written as $X_k = \sigma_1 u_1 v_1^\top i_k$, where i_k is a label vector for the k-st feature map.

Some advantages of proposed approach are the following: (i) The eigenvectors of the feature maps are in agreement with the results obtained using the strongest activated channel. This means that both methods highlight approximately the same region in the domain of the test image, although the eigenvectors appear to have a much clear appearance than the features in the strongest activativated channel. (ii) The eigenvectors are statistically robust in the sense that they remain unchanged when retraining the CNN with different initial values of the weights. (iii) More importantly, the eigenvectors seem to be statistically robust across different network architectures. To demonstrate this aspect, we retrained the AlexNet [12] to classify 4 samples of materials. We then used the SVD-based approach to summarize the feature maps and observed that the resulting eigenvectors have similar appearance to what is shown in figure Figure 15.

The SVD-based method to extract the most representative features of an X-ray image can be appropriate to a large variety of applications, including polymers, metal alloys, among others. Since opposite signs of u_1 represent the same eigenvector, a future work should focus on developing an unsupervised selection of the appropriate sign of the eigenvectors for feature identification.

Acknowledgment. This work was partially supported by Cross-ministerial Strategic Innovation Promotion Program (SIP), 'Structural Materials for Innovation' and 'Materials Integration' for Revolutionary Design System of Structural Materials, and the support of KAKENHI Grants-in-Aid no.18H05482.

References

[1] S. Garcea, Y. Wang, P. Withers, X-ray computed tomography of polymer composites, Composites Science and Technology 156 (2018) 305 – 319. doi:https://doi.org/10.1016/j.compscitech.2017.10.023.
URL http://www.sciencedirect.com/science/article/pii/S0266353817312460

[2] S. Torquato, Random heterogeneous materials: microstructure and macroscopic properties, Vol. 16, Springer Science & Business Media, 2013.

[3] N. Tian, R. Ning, J. Kong, Self-toughening of epoxy resin through controlling topology of cross-linked networks, Polymer 99 (2016) 376–385.

[4] Z. J. Thompson, M. A. Hillmyer, J. Liu, H.-J. Sue, M. Dettloff, F. S. Bates, Block copolymer toughened epoxy: role of cross-link density, Macromolecules 42 (7) (2009) 2333–2335.

[5] E. Avalos, S. Xie, K. Akagi, Y. Nishiura, Bridging a mesoscopic inhomogeneity to macroscopic performance of amorphous materials in the framework of the phase field modeling, Physica D: Nonlinear Phenomena 409 (2020) 132470. doi:https://doi.org/10.1016/j.physd.2020.132470.
URL http://www.sciencedirect.com/science/article/pii/S0167278920300610

[6] L. Petrich, D. Westhoff, J. Feinauer, D. P. Finegan, S. R. Daemi, P. R. Shearing, V. Schmidt, Crack detection in lithium-ion cells using machine learning, Computational Materials Science 136 (2017) 297 – 305. doi:https://doi.org/10.1016/j.commatsci.2017.05.012.
URL http://www.sciencedirect.com/science/article/pii/S0927025617302422

[7] A. Frankel, R. Jones, C. Alleman, J. Templeton, Predicting the mechanical response of oligocrystals with deep learning, Computational Materials Science 169 (2019) 109099. doi:https://doi.org/10.1016/j.commatsci.2019.109099.
URL http://www.sciencedirect.com/science/article/pii/S0927025619303908

[8] H. Hwang, J. Oh, K.-H. Lee, J.-H. Cha, E. Choi, Y. Yoon, J.-H. Hwang, Synergistic approach to quantifying information on a crack-based network in loess/water material composites using deep learning and

network science, Computational Materials Science 166 (2019) 240 – 250. doi:https://doi.org/10.1016/j.commatsci.2019.04.014.
URL http://www.sciencedirect.com/science/article/pii/S0927025619302241

[9] J. Schmidt, M. R. G. Marques, S. Botti, M. A. L. Marques, Recent advances and applications of machine learning in solid-state materials science, npj Computational Materials 5 (1) (2019) 83. doi:10.1038/s41524-019-0221-0.
URL https://doi.org/10.1038/s41524-019-0221-0

[10] M. Schwarzer, B. Rogan, Y. Ruan, Z. Song, D. Y. Lee, A. G. Percus, V. T. Chau, B. A. Moore, E. Rougier, H. S. Viswanathan, G. Srinivasan, Learning to fail: Predicting fracture evolution in brittle material models using recurrent graph convolutional neural networks, Computational Materials Science 162 (2019) 322 – 332. doi:https://doi.org/10.1016/j.commatsci.2019.02.046.
URL http://www.sciencedirect.com/science/article/pii/S0927025619301223

[11] Y. LeCun, Generalization and network design strategies, in: R. Pfeifer, Z. Schreter, F. Fogelman, L. Steels (Eds.), Connectionism in Perspective, Elsevier, Zurich, Switzerland, 1989, an extended version was published as a technical report of the University of Toronto.

[12] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1, NIPS'12, Curran Associates Inc., USA, 2012, pp. 1097–1105.
URL https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-n
pdf

[13] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (2015) 436 EP

–.
URL http://dx.doi.org/10.1038/nature14539

[14] W. Song, G. Jia, H. Zhu, D. Jia, L. Gao, Automated pavement crack damage detection using deep multiscale convolutional features, Journal of Advanced Transportation 2020 (2020) 6412562. doi:10.1155/2020/6412562.
URL https://doi.org/10.1155/2020/6412562

[15] C. M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, Inc., New York, NY, USA, 1995.

[16] M. D. Zeiler, R. Fergus, Visualizing and understanding convolutional networks, in: D. Fleet, T. Pajdla, B. Schiele, T. Tuytelaars (Eds.), Computer Vision – ECCV 2014, Springer International Publishing, Cham, 2014, pp. 818–833.

[17] F. Chollet, Deep learning with Python, Manning Publications Co., 2018.

[18] A. Ziletti, D. Kumar, M. Scheffler, L. M. Ghiringhelli, Insightful classification of crystal structures using deep learning, Nature Communications 9 (1) (2018) 2775. doi:10.1038/s41467-018-05169-6.
URL https://doi.org/10.1038/s41467-018-05169-6

[19] D. Mlakić, S. Nikolovski, L. Majdandžić, Deep learning method and infrared imaging as a tool for transformer faults detection, J. of Electrical Engineering 6 (2). doi:10.17265/2328-2223/2018.02.006.
URL https://doi.org/10.17265/2328-2223/2018.02.006

[20] Y. Xing, C. Lv, H. Wang, D. Cao, E. Velenis, F. Wang, Driver activity recognition for intelligent vehicles: A deep learning approach, IEEE Transactions on Vehicular Technology 68 (6) (2019) 5379–5390. doi:10.1109/TVT.2019.2908425.

[21] S. Lawrence, C. L. Giles, Ah Chung Tsoi, A. D. Back, Face recognition: a convolutional neural-network approach, IEEE Transactions on Neural Networks 8 (1) (1997) 98–113.

[22] K. Rama Linga Reddy, G. Babu, L. Kishore, Face recognition based on eigen features of multi scaled face components and an artificial neural network, Procedia Computer Science 2 (2010) 62 – 74, proceedings of the International Conference and Exhibition on Biometrics Technology. doi:https://doi.org/10.1016/j.procs.2010.11.009.
URL http://www.sciencedirect.com/science/article/pii/S187705091000339X

[23] X. Zhang, J. Zou, K. He, J. Sun, Accelerating very deep convolutional networks for classification and detection, CoRR abs/1505.06798. arXiv:1505.06798.
URL http://arxiv.org/abs/1505.06798

[24] M. Astrid, S.-I. Lee, Deep compression of convolutional neural networks with low-rank approximation, ETRI Journal 40 (4) (2018) 421–434. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.4218/etrij.2018-0065, doi:10.4218/etrij.2018-0065.
URL https://onlinelibrary.wiley.com/doi/abs/10.4218/etrij.2018-0065

[25] Y. Wang, S. Huang, J. Dai, J. Tang, A novel bearing fault diagnosis methodology based on SVD and one-dimensional convolutional neural network, Shock and Vibration 2020 (2020) 1–17. doi:10.1155/2020/1850286.
URL https://doi.org/10.1155/2020/1850286

[26] O. Essid, H. Laga, C. Samir, Automatic detection and classification of manufacturing defects in metal boxes using deep neural networks, PLOS ONE 13 (11) (2018) 1–17. doi:10.1371/journal.pone.0203192.
URL https://doi.org/10.1371/journal.pone.0203192

[27] R. C. Gonzalez, R. E. Woods, Digital Image Processing (3rd Edition), Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2006.

[28] F. Fogelman-Soulié, P. Gallinari, Y. LeCun, S. Thiria, Automata networks

and artificial intelligence, in: Automata networks in computer science, theory and applications, Princeton University Press, 1987, pp. 133–186.

[29] S. S. Haykin, Neural networks and learning machines, 3rd Edition, Pearson Education, Upper Saddle River, NJ, 2009.

[30] J. Bouvrie, Notes on convolutional neural networks, Tech. rep., Center for Biological and Computational Learning, Department of Brain and Cognitive Sciences, Massachusetts Institute of Technology, technical report (2006).

[31] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift (2015). arXiv:1502.03167.
URL https://arxiv.org/abs/1502.03167

[32] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016, http://www.deeplearningbook.org.

[33] S. Santurkar, D. Tsipras, A. Ilyas, A. Madry, How does batch normalization help optimization? (2018). arXiv:1805.11604.
URL https://arxiv.org/abs/1805.11604

[34] K. Jarrett, K. Kavukcuoglu, M. Ranzato, Y. LeCun, What is the best multi-stage architecture for object recognition?, in: Proc. International Conference on Computer Vision (ICCV'09), IEEE, 2009.

[35] V. Nair, G. E. Hinton, Rectified linear units improve restricted boltzmann machines, in: Proceedings of the 27th International Conference on International Conference on Machine Learning, ICML10, Omnipress, Madison, WI, USA, 2010, p. 807814.

[36] J. Nagi, F. Ducatelle, G. A. Di Caro, D. Cirean, U. Meier, A. Giusti, F. Nagi, J. Schmidhuber, L. M. Gambardella, Max-pooling convolutional neural networks for vision-based hand gesture recognition, in: 2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA), 2011, pp. 342–347.

[37] C. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.
URL https://www.microsoft.com/en-us/research/publication/pattern-recognition-machine-learning/

[38] Y. Bengio, Practical Recommendations for Gradient-Based Training of Deep Architectures, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 437–478. doi:10.1007/978-3-642-35289-8_26.
URL https://doi.org/10.1007/978-3-642-35289-8_26

[39] D. Masters, C. Luschi, Revisiting small batch training for deep neural networks (2018). arXiv:1804.07612.
URL https://arxiv.org/abs/1804.07612

[40] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, Journal of Machine Learning Research 15 (2014) 1929–1958, cited By 9824.
URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-84904163933&partnerID=40&md5=b865fd654b3befc5d829dbe5d42b80c3

[41] K. P. Murphy, Machine learning : a probabilistic perspective, MIT Press, Cambridge, Mass. [u.a.], 2013.
URL https://www.amazon.com/Machine-Learning-Probabilistic-Perspective-Computation/dp/0262018020/ref=sr_1_2?ie=UTF8&qid=1336857747&sr=8-2

[42] C. Shorten, T. M. Khoshgoftaar, A survey on image data augmentation for deep learning, Journal of Big Data 6 (1) (2019) 60. doi:10.1186/s40537-019-0197-0.
URL https://doi.org/10.1186/s40537-019-0197-0

[43] M. M. Jadoon, Q. Zhang, I. U. Haq, S. Butt, A. Jadoon, Three-class mammogram classification based on descriptive cnn features, BioMed Research International 2017 (2017) 3640901. doi:10.1155/2017/3640901.
URL https://doi.org/10.1155/2017/3640901

[44] F. Hohman, H. Park, C. Robinson, D. H. Chau, Summit: Scaling deep learning interpretability by visualizing activation and attribution summarizations (2019). arXiv:1904.02323.
URL https://arxiv.org/abs/1904.02323

[45] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning, Springer Series in Statistics, Springer New York Inc., New York, NY, USA, 2001.
URL https://link.springer.com/book/10.1007/978-0-387-84858-7

[46] J. N. Kutz, Data-Driven Modeling & Scientific Computation: Methods for Complex Systems & Big Data, Oxford University Press, Inc., New York, NY, USA, 2013.

[47] O. Alter, P. O. Brown, D. Botstein, Singular value decomposition for genome-wide expression data processing and modeling, Proceedings of the National Academy of Sciences of the United States of America 97 (18) (2000) 10101–10106, 10963673[pmid]. doi:10.1073/pnas.97.18.10101.
URL https://www.ncbi.nlm.nih.gov/pubmed/10963673

[48] M. Kang, J.-M. Kim, Singular value decomposition based feature extraction approaches for classifying faults of induction motors, Mechanical Systems and Signal Processing 41 (1) (2013) 348 – 356. doi:https://doi.org/10.1016/j.ymssp.2013.08.002.
URL http://www.sciencedirect.com/science/article/pii/S0888327013003762

[49] F. Fioranelli, M. Ritchie, H. Griffiths, Classification of unarmed/armed personnel using the netrad multistatic radar for micro-doppler and singular value decomposition features, IEEE Geoscience and Remote Sensing Letters 12 (9) (2015) 1933–1937. doi:10.1109/LGRS.2015.2439393.

[50] G. Strang, Linear Algebra and Its Applications, Academic Press, 1980.

[51] C. Eckart, G. Young, The approximation of one matrix by another of lower rank, Psychometrika 1 (3) (1936) 211–218. doi:10.1007/BF02288367. URL https://doi.org/10.1007/BF02288367

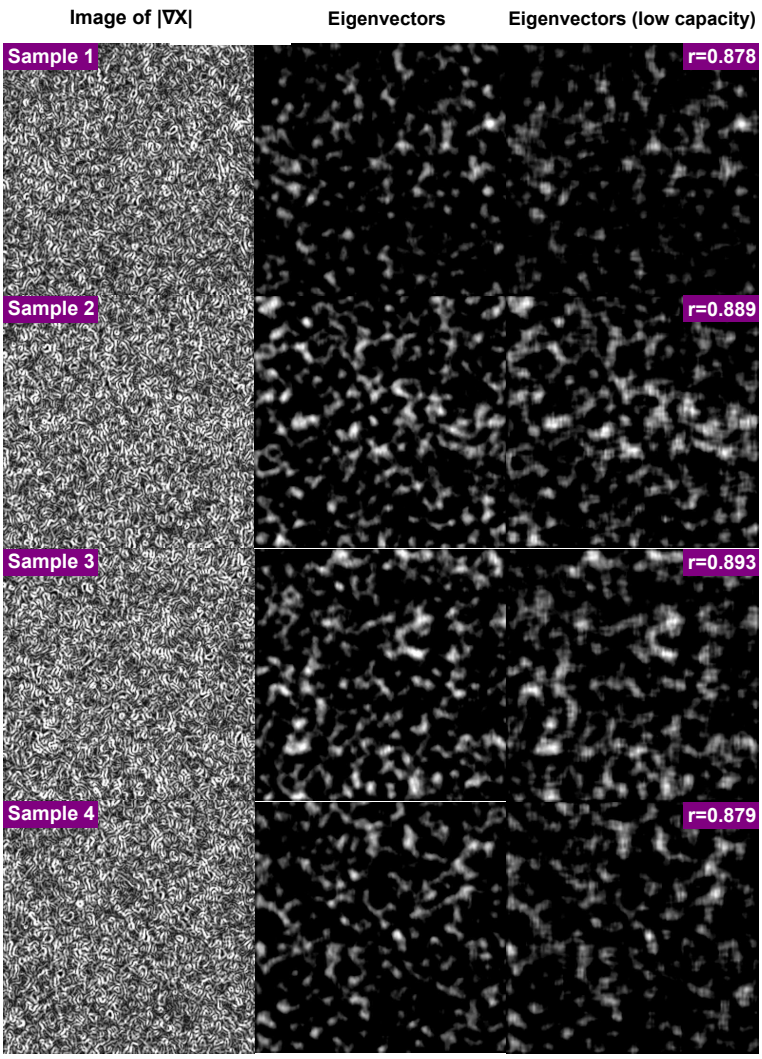


Figure 15: Features of the test images of samples 1-4 (top to bottom) obtained using the eigenvectors of the library of response maps at ReLU 2 as described by Eq. (14). The eigenvectors highlight differences among samples. First column shows test gradient images, $X = |\nabla Y|$. Second and third column show features of the images of the first column obtained using eigenvectors of responses of (i) the CNN described in Table 1 and (ii) the CNN of reduced capacity as described in the text, respectively. The correlation coefficient, r, between the second and third columns is presented for comparison. Notice that the features in sample No. 3 are notably more prominent than those in the other samples, as expected.