# Autoencoders and Convolutional Neural Networks Applied to Structural Damage Detection

Flávio Barbosa[1 [0000−0002−7991−8425]], Lucas Resende[1 [0000−0002−5792−3338]], Rafaelle Finotti[1 [0000−0002−8569−0041]], Alexandre Cury[1 [0000−0002−8860−1286]], Cássio Motta[1 [000−0002−6461−869X]], Hernán Garrido[2 [0000−0002−8840−9390]], Martín Domizio[2 [0000−0003−3157−9337]] and Oscar Curadelli[2 [0000−0003−2877−2750]]

[1] Univeridade Federal de Juiz de Fora
Programa de Pós-Graduação em Engenharia Civil, Brazil
flavio.barbosa@ufjf.br
http://www.ufjf.br/pec
[2] Universidad Nacional de Cuyo - Facultad de Ingeniería
CONICET, Argentina
carloshernangarrido@gmail.com
https://www.uncuyo.edu.ar

**Abstract.** Structural Health Monitoring (SHM) is a growing field in civil engineering and is relevant to detect changes in the state of structures, including identifying possible damage conditions. SHM strategies commonly employ Artificial Intelligence (AI) techniques on raw dynamic data measured from structures to perform classifications or extract features from the original data. Among the AI algorithms for SHM, autoencoders and convolutional neural networks have been identified as promising solutions, being the focus of this paper. Both algorithms are applied to identify eight damage scenarios in a beam from the time histories of the tested structure, pointing out the advantages and disadvantages of each algorithm. The authors tested the beam through monitoring based on image processing using a high-speed camera.

**Keywords:** Structural Health Monitoring · Autoencoder · Convolutional Neural Network · Image Processing

## 1 Introduction

Classically, monitoring the state of conservation of structures is done by analysing the evolution of their modal characteristics [1]. This strategy was adopted in the first works that dealt with this theme and is still relevant today. However, environmental and operational factors may negatively affect this strategy, causing changes in the natural frequencies that could be incorrectly associated with structural damage. Therefore, several Structural Health Monitoring (SHM) techniques have been developed, seeking to consider the environmental effects in damage detection algorithms. The most promising seems to be those based on

Artificial Intelligence (AI) [2]. In these techniques, displacement or acceleration time histories collected from structures at different conditions are presented to algorithms so that they can "learn" if an alteration in the mechanical behaviour is due to operating conditions or consequences of structural damage. Among the most promising AI algorithms that have received special attention from researchers, Sparse Autoencoders (SAE) [3, 4], and Convolutional Neural Networks (CNN) [5, 6] stand out. These algorithms have been successfully used in several recent works that address SHM [7, 8]. In this context, this paper compares the performance of autoencoders and CNN applied to identify eight damage scenarios in a beam tested by the authors, pointing out the strengths and weaknesses verified in each AI technique. The paper's experimental program includes a methodology that provided good results for the displacements measured simultaneously at 60 points on the beam, of which 40 monitored points were used to evaluate the tested AI algorithms.

## 2 Theoretical Background

### 2.1 Sparse Autoencoders - SAE

An autoencoding neural network [9], or just autoencoder, is a type of artificial neural network designed to reproduce its inputs as closely as possible after unsupervised training. An autoencoder is made up of two parts: an encoder, which receives the input data $\mathbf{x}$ and transforms it into a lower dimensional code $\mathbf{h}$; and a decoder, responsible for decoding $\mathbf{h}$ into an output $\mathbf{x}^{'}$ that closely matches $\mathbf{x}$. In other words, autoencoders attempt to reconstruct their inputs as outputs while generating a vector $\mathbf{h}$ that holds much of the information contained within the training data in fewer dimensions. Therefore, autoencoders are useful tools for extracting features that represent the original data with lower dimensionality, which facilitates classification through other methods. Figure 1 shows the structure of the autoencoder applied in the present work: The input layer is composed of vectors $\mathbf{x}$ containing 2500 time steps of displacements; the hidden layer is formed by vectors $\mathbf{h}$ with 15 or 25 elements ($n$=15 or 25), depending on the analyzed case, used in the classification process (damage scenario identification).

A Sparse Autoencoder (SAE) is a type of autoencoding neural network that includes a sparsity penalty term in its training function. It is particularly useful to represent large datasets with a small number of $\mathbf{h}$ components. The function that calculates the error between the input and the output is minimized in the training process (see Goodfellow *et al.* [9]). In the present work, the chosen cost function for the SAE was the mean squared error with sparsity regularizers, and the training algorithm was the scaled conjugate gradient backpropagation. There are a few hyperparameters that must be optimized for SAE models to be able to represent the training data while still being able to generalize. In this work, those hyperparameters are the Sparsity Proportion, the L2 Weight Regularization, and the Sparsity Regularization. The optimization was performed empirically through a grid search process, as described in [8]. Table 1 presents general characteristics of the SAE applied in this work. In this work, the vec-

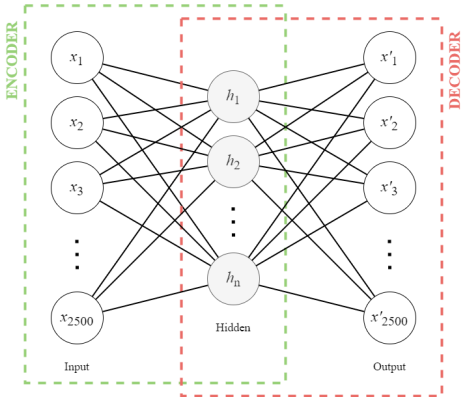



**Fig. 1.** Basic structure of the autoencoder neural network used in this work.

**Table 1.** SAE's characteristics

| | |
|---|---|
| activation function - encoder | logarithmic sigmoid |
| activation function - decoder | logarithmic linear |
| training algorithm | scaled conjugate gradient |
| sparsity ratio | 0.4 |
| regularization term | 0.001 |
| sparsity control term | 4 |
| maximum number of epochs | 1000 |

tors $\mathbf{h}$ with dimension $n = 15$ or $n = 25$, resulting from the application of SAE to the data coming from the structural instrumentation, are used as input data of a Multi-Layer Perceptron (MLP) network for supervised classification of each tested structural damage scenario. From this point in the text, the MLP mentioned above network will be referred to as the shallow network.

### 2.2 Shallow network

The shallow network architecture used in this work to classify the monitoring data is shown in figure 2. For each monitored structure signal ($\mathbf{x}$), the output of the shallow network provides a vector ($\mathbf{y}$) that is associated with each of the 8 experimentally tested structural damage scenarios. For example, for $\mathbf{y} = \{1, 0, 0, 0, 0, 0, 0, 0\}$, it means that the first damage scenario has been identified. Table 2 presents general characteristics of the shallow network used in this work.

### 2.3 Convolutional Neural Network - CNN

CNNs [10, 11] are widely used biologically inspired neural networks that have become the standard in many systems for recognizing objects and events in an

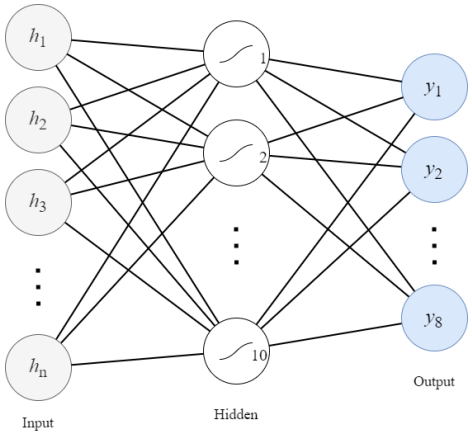



**Fig. 2.** MPL architecture of the shallow network.

**Table 2.** shallow network's characteristics

| | |
|---|---|
| activation function for the hidden layer | logarithmic sigmoid |
| activation function for the output layer | softmax |
| training algorithm | scaled conjugate gradient |
| number of neurons in the hidden layer | 10 |
| maximum number of epochs | 1000 |

image or video. Although almost 30 years have passed since the first CNN was proposed, its modern architectures still share common features with the first, such as convolutional and clustering layers. Furthermore, the most used training method is still the backpropagation technique [12].

The popularity and wide range of application domains of deep CNN can be attributed to the following advantages:

- CNN can merge the processes of parameter extraction and feature classification into a single body of learning;
- Since CNN neurons are sparsely connected, they can process large inputs with high computational efficiency compared to conventional fully connected Multi-Layer Perceptron (MLP) networks;
- CNNs are invariant to minor transformations on the input data, including translation, scaling, skewing, and skew;
- CNN can adapt to different input sizes.

Classically applied to image data (two-dimensional), the use of CNN in problems where the inputs have only one dimension (time series) can be interpreted as applying the technique to an image with one pixel in one dimension.

In this paper, the architectures of the CNN used are shown in figure 3. The input layer has 2500 neurons with the respective 2500 time steps of each monitored response. The next layer is the convolution layer, where 10 convolution

filters with 50 positions are used. Then, batch normalization and ReLU activation function are applied to the convolution layer. The padding technique is used here to keep the ReLU's outputs with the same size as the inputs. The next layer is a pooling layer: clusters with 100 and 125 positions were evaluated, and the highest values (maxpool) were adopted as the result of each grouping. It leads to 25 or 20 (respectively associated to p=100 or p=250 positions of each cluster) in the pooling layer. The softmax activation function is finally applied to the outputs of the pooling layer (fully connected), leading to a classification vector ($\mathbf{y}$) associated with damage scenarios (identical to the output of the shallow network).

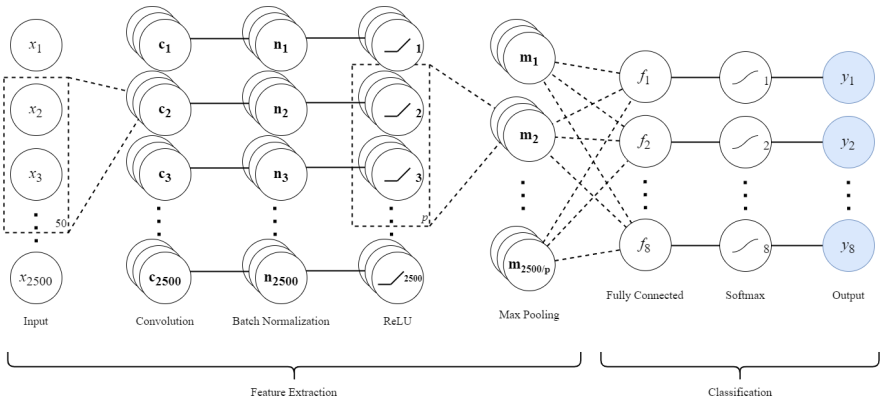



**Fig. 3.** Architecture of the CNN adopted in this paper.

The CNNs are trained in a supervised way by the backpropagation algorithm [12]. During each iteration, the gradient magnitude (or sensitivity) of each network parameter, such as the weights of convolution layers and fully connected layers, is calculated. The parameter sensitivities are then used to iteratively update the CNN parameters until a given stopping criterion is reached. Several optimization methods can be used to calculate the CNN parameters, similar to what happens with the SAE. In this work, the Adam Stochastic Optimization algorithm was applied [13].

## 3 Experimental program

The methodology is validated through an experimental program which includes both laboratory and field conditions. Fig. 4 displays a photograph of the experimental setup. The healthy specimen under test was new and the damage was artificially induced. This ensures that the damage scenarios can be accurately identified to provide a good reference for assessing the performance of the

methodology. Natural daylight was used to illuminate the scene and the background had no special preparation. The idea is to reproduce imperfect situations that are common in field test programs.

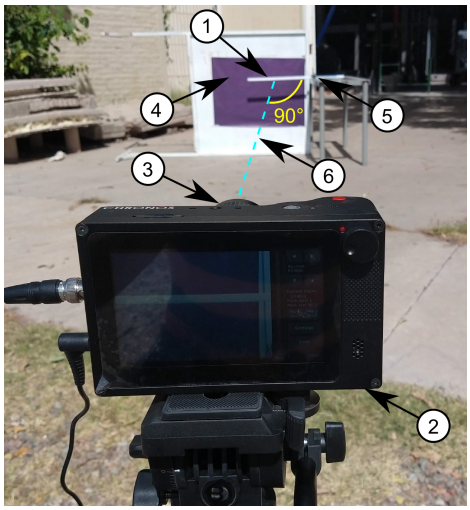



**Fig. 4.** Experimental setup. References: (1) beam (specimen), (2) high-speed camera, (3) lens, (4) background, (5) fixed support, (6) line of sight.

The beam specimen under test, (1) in Fig. 4, consisted of a 1 m long straight beam made of polypropylene with a 25.4x25.4 mm cross-section. The beam was free at the left end and fixed at the right end. The resulting free span was 0.8 m long and the first natural frequency in the healthy state was 7.69 Hz.

The beam was artificially damaged by producing vertical and rectangular Mode I (opening) cracks on its upper face using a steel saw. The actual widths of the cracks were between 0.5 and 1 mm. As a combination of crack locations and depths, eight successive damage scenarios were defined, summarized in Table 3. Cracks were produced at 100, 200 and 300 mm from the fixed end of the beam.

Two videos were recorded for each damage scenario (16 runs in total) with a Chronos 2.1-HD high-speed camera, (2) in Fig. 4 at a frame rate of 1000 fps, with an exposure time of 100% (1/1000 s), and a resolution of 1280 columns by 1024 rows.

In this work, only 1200 columns were in the region of interest for measurement. Columns were grouped into 60 horizontal windows (of 20 columns each), one for each space sample point. The optical setup resulted in a pixel scale of $\approx 0.6$ mm/px.

The video records were taken during free vibration after imposing the following initial conditions: a displacement of 0.05 m at the beam free end, with a velocity of 0 m/s in the whole length.

**Table 3.** Damage scenarios.

| Scenario name | depth (mm) at | | |
|---|---|---|---|
| | 100 mm | 200 mm | 300 mm |
| scenario#0 (without damage) | 0 | 0 | 0 |
| scenario#1 | 5 | 0 | 0 |
| scenario#2 | 10 | 0 | 0 |
| scenario#3 | 13 | 0 | 0 |
| scenario#4 | 13 | 5 | 0 |
| scenario#5 | 13 | 5 | 5 |
| scenario#6 | 13 | 10 | 5 |
| scenario#7 | 13 | 10 | 11 |

## 4 Signal processing

To extract displacements from the video captures, in the particular case of a straight beam, an axial edge is a good natural feature if the background is uniform and of a different colour/intensity than the beam [14]. An additional requirement is that the face corresponding to the selected edge is always hidden from the camera perspective. In this work, the method described in [14] for a single image, which uses the beam edge as a natural feature, was applied to each video frame to obtain displacement records. The main steps of the method are detailed below.

For each video frame:

1. The image is converted to a grey-scale by combining the three 8-bit colour channels into a single double-precision matrix of intensity values.
2. For each column: The horizontal edge is detected using the smooth Gaussian gradient calculated along the vertical direction with a 15 px long vertical Gaussian window of $\pm 3$ standard deviations.
3. The image gradient is binarised using a threshold, and then, the largest connected component is identified using the flood-fill algorithm. This component is a region of the image that will be used as a mask.
4. For each horizontal window:
   (a) For each column:
      i. Where the mask is active: Sub-pixel resolution gradient peaks are estimated and located using cubic splines.
   (b) A linear function is fitted using the peak locations as sample points and their amplitudes as weights in a weighted least squares linear fitting. [1]
   (c) The vertical positions are estimated as the y-intercepts of each fitted straight line (assuming the y-axis is at the middle of each horizontal window).

To illustrate the data obtained with this signal processing method, Fig. 5 shows typical time histories of displacements obtained from the video records.

[1] Weighted fitting is advantageous because image columns where, accidentally, the beam and the background are of similar intensities are penalised. On the contrary, zones of high contrast are emphasised.

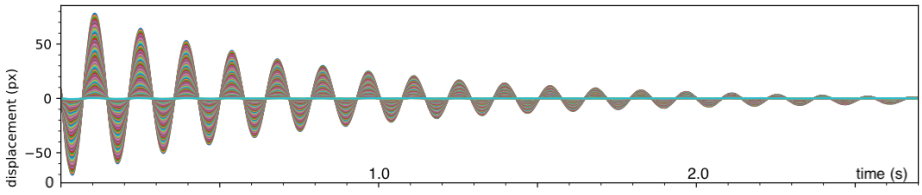



**Fig. 5.** Typical time histories of displacements obtained from the video records.

## 5 Results

This paper tested two configurations of SAE (15 and 25 neurons in the hidden layer) and two configurations of CNN (p=250 and p=100 elements to construct the pooling layer). The ability of each neural network to distinguish data from each structural damage scenario was evaluated. Considering the 8 damage scenarios, with displacements measured at 40 points in 2 test series, there are 640 signals (80 signals for each damage scenario). The respective confusion matrix obtained for a 5-fold cross-validation was calculated in each analysis. The results are presented in Figures 6 and 7, for SAE and CNN, respectively.




**Fig. 6.** Confusion matrices obtained by SAE. On the left for 15 hidden layer neurons. On the right for 25 hidden layer neurons

Both algorithms obtained 100 % of correct answers for the analyses considering 25 neurons in the hidden layer and over 70 % with 15 neurons. Considering the processing time, for the parameters adopted, the CNN converges to 100 % of the efficiency of hits in approximately half of the processing time achieved by




**Fig. 7.** Confusion matrices obtained by CNN. On the left for 250 elements to construct the pooling layer. On the right 100 elements to construct the pooling layer

the SAE. Obviously, the choice of the various parameters of each algorithm directly influences the results. However, it can be seen that, once calibrated, both techniques are very efficient in problems involving SHM.

## 6 Conclusions

This paper presented a comparison between the performance of two AI algorithms - SAE and CNN - for identifying 8 controlled damage scenarios. The tested structure was a beam monitored through image processing. The results showed that SAE required a longer processing time than CNN for the tests carried out. It was also observed that both techniques presented exceptional performance, as they correctly classified the signals from all evaluated damage scenarios. The optimization of these parameters was not the target of this paper, although it represents a pivotal step for such algorithms.

**Acknowledgements:** This study was financed by Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq (Brazil) - grants CNPq/ FNDCT/ MCTI 407256/2022-9, 303982/2022-5, 308008/2021-9 and 304329/2019-3; Fundação de Amparo à Pesquisa do Estado de Minas Gerais - FAPEMIG - grants TEC PPM 00106-17 and TEC PPM-00001-18; Consejo Nacional de Investigaciones Científicas y Técnicas de Argentina (CONICET); Universidad Nacional de Cuyo; and Universidade Federal de Juiz de Fora.

## References


1. Cardoso, R., Cury, A., Barbosa, F.: A clustering-based strategy for automated structural modal identification. Structural Health Monitoring **17**(2), 201–217 (2018) https://doi.org/10.1177/1475921716689239
2. Nunes, A., Amaral, R., Barbosa, F. Cury, A.: A hybrid learning strategy for structural damage detection. Structural Health Monitoring **20**(4), 2143–2160 (2021) https://doi.org/10.1177/1475921720966943
3. Finotti, R., Barbosa, F., Cury, A., Pimentel, R.: Numerical and Experimental Evaluation of Structural Changes Using Sparse Auto-Encoders and SVM Applied to Dynamic Responses. Applied Sciences **11**(24), 11965 (2021) https://doi.org/10.3390/app112411965
4. Resende, L., Finotti, R., Barbosa, F., Cury, A.: Structural damage detection with autoencoding neural networks Health monitoring and numerical modeling of structures. In: Ibero-Latin-American Congress on Computational Methods in Engineering. XLIII CILAMCE Proceedings, Foz do Iguau (2022)
5. Teng, Z., Teng, S., Zhang, J., Chen, G., Cui, F.: Structural Damage Detection Based on Real-Time Vibration Signal and Convolutional Neural Network. Applied Sciences **10**(14), 4720 (2020) https://doi.org/10.3390/app10144720
6. Chen, Z., Wang, Y., Wu, J., Deng, C., Hu, K.: Sensor data-driven structural damage detection based on deep convolutional neural networks and continuous wavelet transform. Applied Intelligence **51**, 5598–5609 (2021) https://doi.org/10.1007/s10489-020-02092-6
7. Avci, O., Abdeljaber, O., Kiranyaz, S., Hussein, M., Gabbouj, M., Inman, D.: A review of vibration-based damage detection in civil structures: From traditional methods to Machine Learning and Deep Learning applications. Mechanical Systems and Signal Processing **147**, 107077 (2021) https://doi.org/10.1016/j.ymssp.2020.107077
8. Finotti, R., Gentile, C., Barbosa, F., Cury, A.: Structural novelty detection based on sparse autoencoders and control charts. Structural Engineering and Mechanics **81**(5), 647–664 (2022) https://doi.org/10.12989/sem.2022.81.5.647
9. GoodfellowW, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge (2016)
10. Abdeljaber, O., Avci, O., Kiranyaz, S., Gabbouj, M., Inman, D.: Real-time vibration-based structural damage detection using one-dimensional convolutional neural networks. Journal of Sound and Vibration **388**, 154–170 (2017) https://doi.org/10.1016/j.jsv.2016.10.043
11. Kiranyaz, S., Avci, O., Abdeljaber, O., Ince, T., Gabbouj, M., Inman, D.: 1D Convolutional neural networks and applications: A survey. Mechanical Systems and Signal Processing **151**, 107398 (2021) https://doi.org/10.1016/j.ymssp.2020.107398
12. Bishop, M.: Pattern Recignition and Machine Learning. Springer-Verlag (2006)
13. Kingma, D., Ba, J.: Adam: A Method for Stochastic Optimization. 3rd International Conference for Learning Representations, San Diego (2015)
14. Garrido, H., Domizio, M., Curadelli, O., Ambrosini, D.: Synthesis of healthy-structure model responses for damage quantification. Structural Health Monitoring **22**(1), 689–713 (2023) https://doi.org/10.1177/14759217221088493