Convolutional Neural Network Approach for Robust Structural Damage Detection and Localization

Nur Sila Gulgec[1], Martin Takáč[2], and Shamim N. Pakzad[3]

[1]Department of Civil and Environmental Engineering, Lehigh University, 117 ATLSS Drive, Imbt Labs, Bethlehem, PA 18015, USA. Email: sgulgec@gmail.com

[2]Department of Civil and Environmental Engineering, Lehigh University, 117 ATLSS Drive, Imbt Labs, Bethlehem, PA 18015, USA.

[2]Department of Industrial and Systems Engineering, Lehigh University, 200 West Packer Avenue, Harold S. Mohler Laboratory, Bethlehem, PA 18015, USA.

ABSTRACT

Damage diagnosis has been a challenging inverse problem in structural health monitoring. The main difficulty is characterizing the unknown relation between the measurements and damage patterns (i.e. damage indicator selection). Such damage indicators would ideally be able to identify the existence, location, and severity of damage. Therefore, this procedure requires complex data processing algorithms and dense sensor arrays, which bring computational intensity with it. To address this limitation, this paper introduces convolutional neural network (CNN), which is one of the major breakthroughs in image recognition, to the damage detection and localization problem. CNN technique has the ability to discover abstract features and complex classifier boundaries which are able to distinguish various attributes of the problem. In this paper, a CNN topology is designed to classify simulated "damaged" and "healthy" cases and localize the damages when it exists. The performance of the proposed technique is evaluated through the finite element simulations of undamaged and damaged structural connections. Samples are trained by using strain distributions as a consequence of various loads with several different crack scenarios. Completely new damage

setups are introduced to the model during the testing process. Based on the findings of the proposed study, the damage diagnosis and localization are achieved with high accuracy, robustness, and computational efficiency.

INTRODUCTION

Structural systems are subjected to damage and deterioration during their service life due to the environmental and operational factors. Providing timely damage evaluation becomes important to ensure lifetime safety of these structures (Fang et al. 2005). For this reason, significant research has been conducted in structural health monitoring (SHM) which is a process of diagnosing the deficiencies affecting the performance of the structures (Farrar and Worden 2007). Data-driven SHM processes need large quantities of data containing detailed condition information over an extended period of time (Shahidi et al. 2016). As the temporal and spatial resolution of monitoring data is drastically increased by advances in sensing technology and with the adaptation of new data collection techniques, SHM applications reach the thresholds of big data (Gulgec et al. 2017a; Gulgec et al. 2016).

Traditional damage identification methods mostly adopt time series or frequency analysis, in conjunction with pattern classification techniques (Gul and Catbas 2009). Many studies focus on extracting patterns from observations and making decisions based on the obtained patterns (Sohn and Farrar 2001; Nair et al. 2006; Yao and Pakzad 2012; Fujimaki et al. 2005). The pattern recognition technique consists of two processes, feature selection and feature classification which require manual effort and expert knowledge.

Such methods are often efficient in identifying structural damage of a particular type that is closely tied with a mechanical model of the behavior of the structural systems and components, which constrains these methods in two aspects: (i) the methods are limited in their scope, depending on the feature that they use for damage identification, and (ii) they are often overwhelmed by big data when damage features are computationally complex. For example, Yao et al. (Yao et al. 2016) presented a damage identification method using cross-correlation of strain data in a steel gusset plate, which demonstrates the wealth of information in data from dense sensing systems, but at

the same time the difficulty in dealing with large datasets and the limitation of the identification methods based on selected features.

The main challenge of the damage identification is originated from defining the unknown relation between the measurements and damage patterns. In order to solve such poorly defined problems, biologically inspired soft-computing techniques have gained traction (Mehrjoo et al. 2008). The most widely used soft-computing method called neural networks were proposed in the 1940s (Flood and Kartam 1994) which is designed such that it can learn from data without a need of feature design process. Since then, they have been practiced in many disciplines including SHM to diagnose damages from the measurement data or its features (Shi and Yu 2012). These studies employed several different inputs to feed the neural network such as modal analysis of vibration response (Zapico et al. 2003; Hadzima-Nyarko et al. 2011; Lee et al. 2005); statistical parameters of vibration (Shu et al. 2013) and strain data (Alavi et al. 2016); frequency response functions (FRFs) (Fang et al. 2005); and wavelet transform coefficients of the acceleration data (Shi and Yu 2012). Nevertheless, most of the prior work still used *damage indicators* as inputs to the neural networks via preprocessing instead of *learning directly from data*.

Although neural network applications are promising, they showed that more complex network architectures are needed to achieve their full potential (Flood 2008). This idea became practical with the improvements in computing power and the introduction of large representative training datasets (Gu et al. 2015). Exploiting the opportunities hidden in big data, deep neural networks (or deep learning) started to gain popularity and soon reached the state-of-the-art technique for image, speech and video recognition. Yet, there are only a few studies using breakthrough deep learning techniques in SHM field. Abdaljaber et al. (2017) used one-dimensional convolutional neural network (CNN) to extract damage features from raw acceleration data (Abdeljaber et al. 2017) and Cha et al. (2017) used raw images taken from structure to perform deep learning-based detection of visible cracks only (Cha et al. 2017).

The previous studies adopted trial-and-error search for tuning the hyperparameters in neural network architecture and did not consider the noise sensitivity of the measurement data and ro-

bustness of the network architecture which may cause the fundamental problem of overfitting (i.e. a neural network can fit even a random noise when the network is not designed carefully (Zhang et al. 2016)). This paper addresses these limitations by proposing an optimized two-dimensional CNN based approach to detect and localize cracks in a noise-tolerant way. The approach feeds the network by using raw strain field measurements which are a direct indicator of stress, fatigue, and failure and can be obtained by an optic-based technique called digital image correlation (DIC) (Pan et al. 2009).

In a former study by the authors, damage diagnosis was performed by using CNN fed through the strain distributions of a structural connection (Gulgec et al. 2017b). In this paper, this idea is expanded by performing a CNN-based methodology for both damage identification and localization with comprehensive noise sensitivity analysis. Proposed methodology shares the frontend layers of a deep convolutional network for both identification and localization tasks. Then, customized backend layers are constructed which are specialized for both tasks. Automatically extracted features in the frontend layers are meaningful for both tasks, hence sharing these layers eliminates the need for two completely separate networks. This reduces the total training time and computational resources.

This methodology learns sophisticated damage features and complex classifier boundaries without extracting hand-designed damage features as is done in traditional methods. The network architecture accomplishes accurate damage diagnosis even from the unseen damage scenarios since the network is trained with a variety of loading cases, damage scenarios, and measurement noise levels. Additionally, the paper presents a comprehensive sensitivity analysis to better understand the behavior of CNN architecture subjected to uncertainties and calibrate it to achieve robust results. Lastly, this approach makes real-time damage identification possible, thanks to (i) frontend layer sharing, (ii) CNN's shared parameterization, and (iii) parallel architecture of GPUs.

The rest of the paper is organized as follows. First, review of relevant studies and a brief overview of CNNs is provided in Section 2 and 3; then the proposed methodology is described in Sections 4. The performance and robustness of the proposed approach are evaluated by numerical

validation in Section 5 and 6. Conclusions and future directions are given in Section 7.

BACKGROUND ON DEEP LEARNING

Deep Neural Networks

Machine learning (ML) is gradually evolved from pattern recognition and learning theory in artificial intelligence (Alpaydin 2014). In 1959, Arthur Samuel defined machine learning as a "field of study that gives computers the ability to learn without being explicitly programmed"(Simon 2013). ML algorithms are designed such that they can learn from data. During this learning process, they build a model which is then used to make data-driven predictions or decisions.

Deep Neural Networks (DNN) are a subfield of machine learning that are conceptually motivated by the human brain. DNNs aim to build a model using a deep graph formed in multiple linear layers followed by non-linear transformations (LeCun et al. 2015). Figure 1 shows an example four layer DNN which consists of an input layer, two hidden layers, and an output layer. The architecture operates on the input instance $x = (x_1, \ldots, x_p)^T$ to get the output of the network. In Figure 1, each circle represents a neuron and an arrow illustrates a connection from the output of one neuron to the input of another connection. Each arrow has an associated weight parameter which indicates the significance of the respective inputs to the output. The output of the neuron in a hidden layer can be determined by the weighted sum of the inputs activated by a nonlinear mapping (e.g., sigmoid, tanh, or others).

For a given input $x \in \mathbb{R}^p$, ML algorithms try to build a prediction function $\theta(x; w)$ parametrized by weights, w. The simplest case of this function can be considered as linear, i.e., $\theta(x; w) = x^T w$. After the family of prediction functions is set, a loss function is selected to measure the error between a prediction and the true value. The most elementary loss function can be denoted as $\ell(\theta(x; w), y) = \|\theta(x; w) - y\|^2$, where $y \in \mathbb{R}^c$ is the true observed value (i.e. label) of the input query x. Soft-max loss entropy and cross-entropy functions can be the other common examples of loss functions (Bishop 2006).

The learning problem seeks the best possible instance of the prediction function from the selected family; in other words, it boils down to finding the best possible values of the weights w

to minimize the loss function. Mathematically speaking, the optimization problem can be defined as following (Shalev-Shwartz and Ben-David 2014):

$$\min_{w} \mathbb{E}_{(X,Y)}[\ell(\theta(x; w), y)]. \tag{1}$$

where the expectation is taken over the true distribution of inputs and labels (X, Y). Nevertheless, the exact knowledge about the true distribution is almost never available in practice. The common practice is to sample n data points $\{(x_i, y_i)\}_{i=1}^{n}$ (frequently called training data) from the unknown distribution, and minimize the empirical loss instead:

$$\min_{w} \frac{1}{n} \sum_{i=1}^{n} \ell(\theta(x_i; w), y_i). \tag{2}$$

Convolutional Neural Networks

Convolutional neural networks (CNN) are one of the most widely used types of deep neural networks. The framework of CNN was first proposed by LeCun et al. in 1998 (LeCun et al. 1998) to classify handwritten digits. CNN became a breakthrough in visual and speech recognition in the last few years with the introduction of a highly parallel programmable unit called GPUs and large-scale hierarchical image database (Deng et al. 2009). CNN architectures kept evolving (Krizhevsky et al. 2012; Simonyan and Zisserman 2014; Zeiler and Fergus 2014) through the years and the performance improved significantly as the networks become more complex and deeper (Szegedy et al. 2015; He et al. 2015). The reason behind such achievement was the ability to keep temporal features of the input and reduce memory requirements by using fewer parameters (LeCun and Bengio 1995).

Convolutional neural networks are composed of three architectural frameworks: local receptive fields, shared weights, and spatial sub-sampling (LeCun et al. 1998). Passing the same set of units all over the input allows extracting multiple feature maps. In this case, feature map shifts as the same amount that input shifts. This is called local receptive fields which makes CNN robust to the translation and distortion of the input. Furthermore, the weights and biases are shared through the

feature maps. This characteristic reduces the learned parameters as well as the memory demands. Lastly, spatial sub-sampling helps reducing the resolution of the feature maps and preventing the sensitivity of the outputs under shifts and rotations.

CNNs receive the input as 3D volumes (width, height, depth). As an example from image recognition, the depth of a colored image (i.e. having red-green-blue color channels) is three, whereas the depth of a gray image is one. These 3D input volumes feed the CNN architecture which can be constructed by using three types of layers:

1. *Convolutional layer (CONV)* parameters are learnable filters where each filter (weights or kernels) has spatially small width and height shared in the full depth of the input. While slidding these weights, CONV layer computes the dot product between these filters and the small region of the input in any position. Then, the weighted sum of the input and weights is activated by the nonlinear functions to form feature maps. This operation is called "convolution".

   The size of the feature map is associated with a variety of hyperparameters such as: the number of kernels, kernel size, number of strides and zero-padding. The nonlinear activation maps are generated based on the number of kernels used. The number of strides determines the number of instances skipped in each position, whereas zero-padding controls the number of zeros added to the borders of the input. Figure 2 shows a convolution operation for an input size (I) of 7x11x2. In the figure, a kernel size (K) of 2x2x2 passes through the input with the stride (S) of 3 and no zero-padding (P). The output size is found by the equation $(I - K + 2P)/S + 1$.
2. *Pooling layer (POOL)* performs a downsampling operation in the feature maps using maximum, average or sum operations. Pooling layer gets the input and resizes it to reduce the number of parameters and control the overfitting. Similarly, the output size is controlled by different hyperparameters such as pool size and the number of strides.
3. *Fully-connected layers (FC)* operate on the stacked convolutional or pooling layer outputs and compute the weighted sum of inputs with a non-linear mapping as described in the

overview of DNNs.

PROPOSED METHODOLOGY

Overview

This section gives a general map of the proposed technique. As shown in Figure 3, the methodology consists of training and testing phases. Training phase operates on raw strain fields from structures. After normalizing each strain field by its absolute maximum, the search mechanism finds a good set of hyperparameters which improves the performance of network architecture. Then, the selected architecture is trained to minimize the error between predictions and true labels.

The training phase consists of two tasks: detection and localization. Detection task determines the existence of damage where it is treated as a classification problem (i.e. 0 for undamaged and 1 for damaged). Localization task treats the case as a regression problem where the goal as regression problem where the goal is accurate estimation of the boundaries of the damaged area. In the proposed methodology, both of the tasks use shared layers in the early stages of the deep learning pipeline. These layers are specialized to extract local features that are common for both localization and detection. Then, these early layers are fed into task-specific layers. Shared frontend layers avoid having two separate networks, provide more efficient learning, shorter training time and lower computation cost.

The trained model parameters are stored to be used in testing phase. In this phase, raw strain fields are fed into the CNN architecture to predict the labels for detection and localization tasks.

Hyperparameter Selection

The CNN architectures can be built in various ways by using the sequence of convolutional (CONV), pooling (POOL) and fully connected (FC) layers. The performance of the neural networks critically depends on identifying a good set of hyperparameters (Pei et al. 2004). In this study, these hyperparameters include learning rate, the number of CONV and FC layers, the number of kernels, kernel and pool sizes, and the number of hidden layer sizes. In order to find the structure with good configuration, the hyperparameter search mechanism is implemented for both damage

detection and localization tasks (Li et al. 2016).

Different networks are constructed with randomly selected hyperparameters. The 10% of the networks which has the worst validation score is removed after the first run, and the remaining networks are run for another set of an epoch. The runs are repeated until the best 10 networks remain in the pool. After the best network is selected for the damage identification part, the output of the last convolutional layer is stored and used as an input for the hyperparameter search for the localization task. The search for this task is performed on with FC layers only.

Training

The training process is comprised of two phases: feed-forward and back-propagation (Rojas 2013). The feed-forward process evaluates the prediction function for given input instances. Then back-propagation step adjusts the weights in proportion as their contributions to the total error (Rumelhart et al. 1988) by using a stochastic gradient descent (SGD) algorithm (Robbins and Monro 1951). After the gradients are calculated with SGD, the detection and localization parameters are updated with the learning rates η_{det} and η_{loc}, respectively. Overfitting is prevented by monitoring the validation dataset performance in every complete one forward and backward pass (epoch). When architecture performance is improved sufficiently on the validation dataset, the training process is stopped.

Weight Initialization

The first step of training is initializing the weights to control input instances in a reasonable range along the layers. This study adopts Xavier initialization for tanh function (Glorot and Bengio 2010). Weight initialization of i^{th} layer is set to have a uniform distribution in the interval $\left[-\sqrt{\frac{6}{n_{i-1}+n_i}}, \sqrt{\frac{6}{n_{i-1}+n_i}}\right]$ where n_{i-1} and n_i are the number of units in the $(i-1)^{th}$ and i^{th} layer.

Prediction Functions

Feed-forward step evaluates different prediction functions for detection and localization tasks. This study employs soft-max classifier (Bishop 2006) to predict the label of the detection output (y_{pred}) which is either healthy or damaged. The class i of the input x is estimated by selecting the

maximum probability of soft-max function defined as follows:

$$[\mathsf{softmax}(\theta(x;w))]_i = \frac{e^{[\theta(x;w)]_i}}{\sum_j e^{[\theta(x;w)]_j}}, \tag{3}$$

$$y_{pred} = \mathsf{argmax}_i([\mathsf{softmax}((\theta(x;w))]_i). \tag{4}$$

The localization task aims to predict the location of the crack which is defined by a bounding box vector z_{pred}. For this reason, this task uses a regressor instead of a classifier. The bounding box i of the input x is estimated by the following function:

$$[z_{pred}]_i = \sum_j [\theta(x;w)]_j. \tag{5}$$

Loss functions

The proposed model adopts two separate loss functions for the detection and localization tasks. The diagnosis part employs the negative log-likelihood function, where optimal architecture parameters θ^* are learned by maximizing the likelihood of the dataset. On the other hand, the localization task calculates the loss between the predicted and true bounding box with ℓ_2 loss function:

$$\mathcal{L} = \sum_{i=1}^{N} \frac{([z_{pred}]_i - z_i)^2}{N}, \tag{6}$$

where z is the predicted bounding box and N is the batch size (i.e. number of training samples in one fee-forward pass). During the regressor training, the bounds of the boxes are updated based on the conditions of equations 7-10. If these criteria are satisfied with the pre-defined threshold values, the localization marked as correct localization.

$$|\min(a_1, a_2) - \min(\hat{a}_1, \hat{a}_2)| \leq \mathsf{thr}_a \tag{7}$$

$$|\min(b_1, b_2) - \min(\hat{b}_1, \hat{b}_2)| \leq \mathsf{thr}_b \tag{8}$$

$$|\max(\hat{a}_1, \hat{a}_2) - \max(a_1, a_2)| \leq \mathsf{thr}_a \tag{9}$$

$$|\max(\hat{b}_1, \hat{b}_2) - \max(b_1, b_2)| \leq \mathsf{thr}_b \tag{10}$$

where $(\hat{a}_1, \hat{a}_2, \hat{b}_1, \hat{b}_2)$ are predicted box coordinates, (a_1, a_2, b_1, b_2) are true box coordinates, and thr is the user-defined threshold.

NUMERICAL VALIDATION

Data Preparation

Damage identification process requires a large training set of correctly classified damage states (Elkordy et al. 1993). In this study, well-known damage states of a structural connection are simulated by using ABAQUS shell elements. The modeled connection consists of two C8x11.5 channels welded to a steel plate with the dimension of 71x36x0.6 cm (28x14x1/4 inches) as visualized in Figure 4. Each channel member is 51 cm long and has 20 cm overlap with the main gusset plate. The finite element model adopts the mesh size of 1.3 cm (0.5 inches). The behavior of the steel is introduced as elastic-perfectly plastic material with a yield strength of 250 MPa. As can be seen from in Figure 4, stress gradients occur at the crack tips as well as the central part of the plate when it undergoes to a plastic region. Strain distribution in the direction of loading is represented by 28x56x1 tensors and used to feed the CNN architecture after normalized by its absolute maximum value.

Training, validation and test datasets are formed by modeling different loading cases, damage scenarios, and noise levels. The load is selected from uniformly distributed load $\sim U$[-445 kN (compression), 534 kN (tension)] and applied to the end of the channel members. The damages in the gusset plate are simulated as 2.5 cm long cracks which are the smallest crack size given the mesh size. The crack locations are chosen at the beginning of each run with a specified load level. The coordinates of the cracks changing between the two corners of the middle part of the plate [lower left corner point A with coordinates (21.6,2.5) to upper right corner point B with coordinates (45.9,33.0)] are shown in Figure 5. In order to assess the approach with completely unseen damaged samples, none of the coordinates of the training set is used in the testing samples.

The uncertainty in the measurement process is simulated as an additive Gaussian noise $\sim N(0, \sigma^2)$, where σ is the standard deviation of the measurement noise. Different noise levels (i.e.

the ratio between the standard deviation of measurement noise to actual strain values) are generated to compute the influence of the noise on CNN architecture performance.

The crack coordinates of the "single-damaged" samples are also collected for the localization task. Crack location is stored as bounding box (a_1, b_1, a_2, b_2), where b_1 and b_2 indicate the coordinates of the tips of the crack. While defining a_1 and a_2, 1.3 cm is subtracted and added to the x coordinate of the crack to reduce the rounding error in the direction of loading; for example, if a crack is located between (21.6,2.5) to (21.6,5.0), the bounding box is defined as [20.3, 2.5, 22.9, 5.0]. For the "healthy" samples, bounding box is set to $[0, 0, 0, 0]$.

While preparing damaged samples, 72 different crack locations and 3,000 loading scenarios are used. None of the coordinates of training sets is used in the testing samples (i.e. 36 locations for training, 36 locations for testing as shown in Figure 5). Healthy samples are modeled with 6,000 loading scenarios. As a result, a total of 6,000 healthy and 6,000 damaged samples are generated. Then, four different noise levels (2%, 5%, 10% and 15%) are added to noise-free samples to produce a total 30,000 healthy and 30,000 damaged samples. This dataset is called *Dataset 1* and distributed to training, validation and testing samples.

Hyperparameters

A total of 50 networks are constructed with randomly selected hyperparameters in detection task. The hyperparameter range for the detection task has the following characteristics: learning rate [2 to 2^{-8}]; the number of CONV and FC layers $[1, 2 \text{ or } 3]$; the number of kernels [2 to 2^7]; kernel size [(3x3) or (5x5)] with stride of 1 and without zero-padding; max-pool size [(1x1) with stride of 1] or [(2x2) with stride of 2] and randomly selected hidden layer sizes.

The last convolutional layer of the best architecture in the detection task is stored as an input for the hyperparameter search for the localization task. The search for localization task is performed on a total of 70 networks with FC layers only. The networks for localization task are built with hyperparameters using learning rate [2^{-6} to 2^{-18}]; the number of FC layers $[1, 2 \text{ or } 3]$; and randomly selected hidden layer sizes. Activation function tanh() is adopted for the activation of the layers for both detection and localization tasks.

Training and Proposed Architecture

Training is implemented by using a Python library called Theano to optimize the mathematical expressions consisting multi-dimensional arrays (Theano Development Team 2016). Higher performance is achieved by using NVIDIA Tesla K80 GPUs that enables parallelism for data-intensive calculations.

In this study, mini-batch SGD algorithm with a batch size of $N = 64$ is implemented. Identical thresholds are used for thr_a and thr_b described through the equations 7-10. In order to discover the effect of the size of search area on the localization accuracy, the sides of the bounding box are increased in length by different threshold values. The threshold values $1.3, 2.6$ and 5.1 cm are selected to have an increase in length by scale factors of $2, 3$ and 5, respectively. The scale coefficients are selected randomly but not to exceed the quarter of the area of the central part of the plate. Thresholds are illustrated in the Figure 6 with the values of thr = 1.3 cm, thr = 2.5 cm and thr = 5.1 cm.

Figure 7 shows the proposed architecture as a result of the hyperparemeter search mechanism. The network consists of three convolutional layers followed by two separate fully connected layers for detection and localization tasks. The detection part classifies 28x56x1 inputs as healthy or damaged, whereas the localization part predicts the bounding box of the crack area. The convolutional layers receive the input layer and pass them through a filter size of (3x3). As a result of these CONV layers, the network forms 8, 16 and 32 feature maps. Max-pooling operation is implemented right after first and second convolution layer. Max-pool size of (2x2) with a stride of 2 is used for POOL layers. The feature maps of the last convolutional layer are stacked together in an array and given as an input to the fully connected layers with a hidden layer size of $[836 - 767]$ for the detection task and $[1305 - 1191 - 406]$ for the localization task. The learning rate of $\eta_{det} = 0.0451$ and $\eta_{loc} = 0.0026$ are used for the detection and localization parts, respectively.

As mentioned earlier, CNNs have the ability to keep spatial features of inputs. In order to visualize this ability, the activated feature maps after POOL-1, POOL-2 and CONV-3 layers of a correctly identified damaged sample are shown in Figure 7. The activations are normalized to have

the scale between 0-1, where white represents 0 and black represents 1. The figure shows that the damage location (i.e. right top corner) is still visible during the Stage 1 and 2. After CONV-3 layer (Stage 3), the features become abstract where it is almost impossible to design it by hand.

RESULTS AND DISCUSSION

The performance and sensitivity analysis of the proposed methodology is evaluated in this section. The accuracy and robustness of the CNN architecture are discussed for both detection and localization tasks.

Detection Task

This section presents the performance and sensitivity analysis of the detection task. In order to measure the effect of noise, two additional datasets are prepared where both consist of 6,000 undamaged and 6,000 damaged samples. *Dataset 2* is formed by only noise-free samples and *Dataset 3* is selected from a subset of *Dataset 1*. Hyperparameter search is performed for these two datasets for fair comparison. Trained models are then tested with samples including a variation of different noise levels (0% – noise-free, 2%,4%,6%,8%,10%,12%,14% and 16%) for 100 times. Proposed network topologies for two additional training processes are listed as follows:

Training of Dataset 2: The network is trained with *Dataset 2* which consists of only noise-free samples. The proposed network for the second case is composed of two CONV layers followed by POOL layers, and two FC layers. The CONV layer adopts the filter size of (3x3) with kernel numbers of 2 and 4. Max-pool size of (2x2) with a stride of 2 is used for POOL layers. The last POOL layer is connected to the two FC layers size of $[373 - 223]$. The learning rate is chosen as $\eta_{det} = 0.0158$.

Training of Dataset 3: The selected network for *Dataset 3* includes two CONV layers with the filter size of (3x3) with kernel numbers of 8 and 32. Similar to the first case, max-pool size of (2x2) with a stride of 2 is used for POOL layers. The network has the two FC layers size of $[2477 - 804]$ after shared layers. The learning rate of $\eta_{det} = 0.069$ is adopted.

The sensitivity analysis of three training cases is visualized in Figure 8. Figure 8(b) presents the

testing performance of *Dataset 2* which has the worst testing performance among the three cases. Although the testing error is 1.19% for lower noise levels, it reaches around 12% under the noise level of 16%. It is noticeable that the error rate exponentially increases with the increase in the noise levels.

As can be observed from Figure 8(c), the testing accuracy increases significantly compared to the architecture trained with noise-free samples. The performance of trained architecture stays stable with the increase in noise level. Consequently, the introduction of different noise levels during the training process helps network to learn damage features under uncertainty.

The figure 8(a) illustrates the best testing performance from the given training cases. According to the figure, the proposed architecture identifies the previously unseen damages with 0.21% error on noise-free samples. This error rate represents that the CNNs are capable of learning the damage features almost perfectly even with the smallest crack size if enough training cases are provided. Furthermore, test error does not change significantly even under 16% noise which shows that the proposed methodology is robust for various levels of noise.

As discussed previously, deep learning-based approaches can be effective in identifying structural damage more than a particular scenario unlike traditional methods. They have a capability of generalization when it is designed carefully. In order to evaluate this characteristic, the performance of the proposed method is assessed with a larger crack size. A total of 3,000 samples with a crack size of 5.1 cm is tested for detection task. As shown in Figure 9, although samples with crack size of 5.1 cm are not included in the training dataset, the testing accuracy is almost perfect. The filters used in the architecture manage to highlight the cracked region.

In summary, the introduction of uncertainty in measurement noise avoids overfitting which leads to better testing and generalization performance. Such fact emphasizes that training dataset selection is vital in designing CNN architectures. Another point which is worthwhile to mention is adding more samples to the training dataset increases the accuracy and robustness.

Localization Task

This section discusses the main findings of the localization task. The localization part of the network is trained with *Dataset 1* including both noise-free and noisy samples which result in better detection accuracy. In order to eliminate the error coming from the detection task, the localization task is run with both healthy and damaged samples. The CNN architecture is tested under different noise levels and different threshold values.

Figure 10 displays the percent localization error under different noise levels as well as different user-defined threshold values such as; thr = 1.3 cm, thr = 2.5 cm and thr = 5.1 cm. According to the Figure 10(a), the proposed architecture localizes the crack with 96.8% accuracy when the noise level is zero and the threshold value is 1.3 cm. This error rate demonstrates that the proposed CNN architecture successfully localizes the damages. The testing performance of different noise levels does not change significantly which indicates the robustness of the method (i.e. Testing accuracy is 95.3% when the network is tested with 16% noisy samples).

Figure 11 shows an example of correct classification by using the threshold value of 1.3 cm. When the crack location is searched in the larger area by increasing the threshold, the error rate is reduced even further. The error rate is almost 1% under all levels of noise for both threshold values 2.5 and 5.1 cm as shown in Figure 10(b)(c).

Computational Performance

The computational performance of the case study is evaluated on Intel® Xeon® CPU E5-2620 v3 and NVIDIA Tesla K80 GPUs. The time required for training and testing phases for a single strain field and a batch size of 64 strain fields are summarized in Table 1. In the training phase, one forward and backward pass is considered. The computation times for shared layers + detection task, shared layers + localization task and only localization task are compared in Table 1.

As illustrated in Table 1, testing time for all tasks is under 20 ms for both of the hardware. A video stream input with 25 frames per second would give 40 ms time budget to complete testing for a single sample which can be considered as a real-time requirement. Therefore, the proposed methodology achieves real-time requirement for the testing phase. In addition, as mentioned earlier,

proposed architecture reduces the computational needs by exploiting shared feature extractor for the detection and localization tasks. The computation time for only localization task is almost half of the computation time for shared layers + localization task which proves the efficiency of the methodology. As a result, the proposed study reduces the shorter training time and lower computation cost.

RELATED WORK

There have been several strain-based studies on damage detection and localization for plate-like structures in literature. Finding damage index is one of the widely used techniques in crack identification where the existence of cracks is defined by comparing healthy and damaged states, and the potential damage location is estimated by constructing a threshold value (Li 2010). Some examples of these strain-based damage indicators can be listed as modal strain energy index where the index is ratio of summations of fractional energies of elements before and after damages (Cornwell et al. 1999), curvature mode shape index (Yam et al. 2002), cross-correlation of strain data (Yao et al. 2016), strain frequency response function (Swamidas and Chen 1995), spectral strain energy (Bayissa and Haritos 2007) and the strain measured by a sensor against the measurement obtained by neighbor sensors (Laflamme et al. 2016). Another study (Choi et al. 2005) adopted the changes in modal compliance distribution and demonstrated the validation of their approach on 60 cm x 40 cm plate including 48 elements. This study defined the percentage of false positive error as the ratio of the number of false positive predictions over the number of healthy elements in the plate. The percentage of false positives was stated to be 6.4% for noise-free case and 8.5% for 3% signal-to-noise (N/S) ratio where the crack size was 5.2 cm. The percentage of false positives for crack size of 13 cm were 4.3%, 6.5% and 8.7% for 0%, 1% and 3% N/S ratio, respectively. There were also several approaches where damage is characterized by the probabilistic behavior. As an example, the research presented in (Hasni et al. 2017) extracted probability density function of strain time histories of gusset plate and identified cracks by using support vector machine (SVM) classifier. The best performance of the classifier is noted as 82% where the performance is the number of correctly classified data points divided by the total number of data points on the girder.

In addition, several approaches adopt strain-based damage indicators as inputs to artificial neural networks (ANNs). In (Katsikeros and Labeas 2009), Discrete Fourier Transformation and Principle Component Analysis were used to generate damage features which are then utilized to feed ANN structure. The study was evaluated on a simulated lap-joint structure and validated by using mean square error of target and predicted crack parameters. Another study (Sbarufatti et al. 2013) normalized each sensor of a confined region with respect to the average value measured by all the sensors within the same region to obtain damage index. Associated damage index map was validated by using the simulation of 60 cm x 50 cm panel with rivets. The detection accuracies were obtained as the average of the output values from 50 ANNs. For noise-free case, detection accuracies were reported as 92.5% and 95% for 6 cm and 8 cm cracks, respectively. The accuracies dropped as the increase in noise level such as; detection accuracies for 12% additive Gaussian noise case were 25% for 6 cm crack and 90% for 8 cm crack, respectively.

Majority of described damage identification approaches are effective in detecting a particular type and number of crack scenarios. Nevertheless, there are several limitations in these techniques, as mentioned earlier. First, traditional approaches need for measurements from both baseline and unknown state of structures. Second, such approaches consist damage feature design and threshold selection processes that require manual effort and human expertise. Finally, they aim to reduce the number of measurements due to the difficulty in dealing with large datasets.

CONCLUSION

The major challenge of damage diagnosis is characterizing the unknown relation between the measurements and damage patterns. To address this limitation, this paper introduces convolutional neural network (CNN), which is one of the major breakthroughs in image recognition, to this damage detection and localization problem. CNN technique has the ability to discover abstract features and complex classifier boundaries which are able to distinguish various features of the problem.

In our study, the abstract feature maps are discovered with CNN technique to classify “damaged” and “healthy” cases modeled through the analytic simulations. The computational needs of the

methodology are decreased by exploiting CNN's shared parameterization and GPU's massively parallel architecture.

The proposed CNN architecture can process the available data and adjust itself to the control variables such as measurement noise. As a result, this study accomplishes high accuracy, robustness, and computational efficiency which holds a great potential for real-time damage diagnosis and localization challenge. It is also notable to mention that the performance of deep neural networks highly depends on the training dataset. The selection of training dataset requires the representation of as many of cases possible to predict test cases accurately. The results show that CNN architecture performs with higher accuracy and robustness when training dataset is formed with noise-free and noisy data. Consequently, training datasets should be designed considering the existence of uncertainties.

In order to discover more about the abilities of convolutional neural networks, further research is needed. This work should aim to (1) perform damage diagnosis with more complicated loading scenarios and larger structures, (2) determine the severity of the damages for multiple damage cases, (3) testing the designed network on the real experimental setup.

Acknowledgment. Research funding is partially provided by the National Science Foundation through Grant No. CMMI-1351537 by Hazard Mitigation and Structural Engineering program, and by a grant from the Commonwealth of Pennsylvania, Department of Community and Economic Development, through the Pennsylvania Infrastructure Technology Alliance (PITA). Martin Takáč was supported by National Science Foundation grant CCF-1618717 and CMMI-1663256.

REFERENCES

Abdeljaber, O., Avci, O., Kiranyaz, S., Gabbouj, M., and Inman, D. J. (2017). "Real-time vibration-based structural damage detection using one-dimensional convolutional neural networks." *Journal of Sound and Vibration*, 388, 154–170.

Alavi, A. H., Hasni, H., Lajnef, N., Chatti, K., and Faridazar, F. (2016). "An intelligent struc-

tural damage detection approach based on self-powered wireless sensor data." *Automation in Construction*, 62, 24–44.

Alpaydin, E. (2014). *Introduction to machine learning*. MIT press.

Bayissa, W. and Haritos, N. (2007). "Structural damage identification in plates using spectral strain energy analysis." *Journal of Sound and Vibration*, 307(1-2), 226–249.

Bishop, C. M. (2006). "Pattern recognition." *Machine Learning*, 128.

Cha, Y.-J., Choi, W., and Büyüköztürk, O. (2017). "Deep learning-based crack damage detection using convolutional neural networks." *Computer-Aided Civil and Infrastructure Engineering*, 32(5), 361–378.

Choi, S., Park, S., Yoon, S., and Stubbs, N. (2005). "Nondestructive damage identification in plate structures using changes in modal compliance." *Ndt & E International*, 38(7), 529–540.

Cornwell, P., Doebling, S. W., and Farrar, C. R. (1999). "Application of the strain energy damage detection method to plate-like structures." *Journal of sound and vibration*, 224(2), 359–374.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). "Imagenet: A large-scale hierarchical image database." *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, IEEE, 248–255.

Elkordy, M., Chang, K., and Lee, G. (1993). "Neural networks trained by analytically simulated damage states." *Journal of Computing in Civil Engineering*, 7(2), 130–145.

Fang, X., Luo, H., and Tang, J. (2005). "Structural damage detection using neural network with learning rate improvement." *Computers & structures*, 83(25), 2150–2161.

Farrar, C. R. and Worden, K. (2007). "An introduction to structural health monitoring." *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 365(1851), 303–315.

Flood, I. (2008). "Towards the next generation of artificial neural networks for civil engineering." *Advanced Engineering Informatics*, 22(1), 4–14.

Flood, I. and Kartam, N. (1994). "Neural networks in civil engineering. ii: Systems and application." *Journal of Computing in Civil Engineering*, 8(2), 149–162.

Fujimaki, R., Yairi, T., and Machida, K. (2005). "An approach to spacecraft anomaly detection problem using kernel feature space." *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, ACM, 401–410.

Glorot, X. and Bengio, Y. (2010). "Understanding the difficulty of training deep feedforward neural networks.." *Aistats*, Vol. 9, 249–256.

Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., Liu, T., Wang, X., and Wang, G. (2015). "Recent advances in convolutional neural networks." *arXiv preprint arXiv:1512.07108*.

Gul, M. and Catbas, F. N. (2009). "Statistical pattern recognition for structural health monitoring using time series modeling: Theory and experimental verifications." *Mechanical Systems and Signal Processing*, 23(7), 2192–2204.

Gulgec, N. S., Shahidi, G. S., Matarazzo, T. J., and Pakzad, S. N. (2017a). "Current challenges with bigdata analytics in structural health monitoring." *Structural Health Monitoring & Damage Detection, Volume 7*, Springer, 79–84.

Gulgec, N. S., Shahidi, S. G., and Pakzad, S. N. (2016). "A comparative study of compressive sensing approaches for a structural damage diagnosis." *Geotechnical and Structural Engineering Congress 2016*, 1910–1919.

Gulgec, N. S., Takáč, M., and Pakzad, S. N. (2017b). "Structural damage detection using convolutional neural networks." *Model Validation and Uncertainty Quantification, Volume 3*, Springer, 331–337.

Hadzima-Nyarko, M., Nyarko, E. K., and Morić, D. (2011). "A neural network based modelling and sensitivity analysis of damage ratio coefficient." *Expert systems with applications*, 38(10), 13405–13413.

Hasni, H., Alavi, A. H., Jiao, P., and Lajnef, N. (2017). "Detection of fatigue cracking in steel bridge girders: a support vector machine approach." *Archives of Civil and Mechanical Engineering*, 17(3), 609–622.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). "Deep residual learning for image recognition." *arXiv preprint arXiv:1512.03385*.

549 Katsikeros, C. E. and Labeas, G. (2009). "Development and validation of a strain-based structural
550 health monitoring system." *Mechanical Systems and Signal Processing*, 23(2), 372–383.
551 Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). "Imagenet classification with deep convo-
552 lutional neural networks." *Advances in neural information processing systems*, 1097–1105.
553 Laflamme, S., Cao, L., Chatzi, E., and Ubertini, F. (2016). "Damage detection and localization
554 from dense network of strain sensors." *Shock and Vibration*, 2016.
555 LeCun, Y. and Bengio, Y. (1995). "Convolutional networks for images, speech, and time-series."
556 *MIT Press*.
557 LeCun, Y., Bengio, Y., and Hinton, G. (2015). "Deep learning." *Nature*, 521(7553), 436–444.
558 LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). "Gradient-based learning applied to
559 document recognition." *Proceedings of the IEEE*, 86(11), 2278–2324.
560 Lee, J. J., Lee, J. W., Yi, J. H., Yun, C. B., and Jung, H. Y. (2005). "Neural networks-based damage
561 detection for bridges considering errors in baseline finite element models." *Journal of Sound and*
562 *Vibration*, 280(3), 555–578.
563 Li, L., Jamieson, K. G., DeSalvo, G., Rostamizadeh, A., and Talwalkar, A. (2016). "Efficient
564 hyperparameter optimization and infinitely many armed bandits." *CoRR*, abs/1603.06560.
565 Li, Y. (2010). "Hypersensitivity of strain-based indicators for structural damage identification: A
566 review." *Mechanical Systems and Signal Processing*, 24(3), 653–664.
567 Mehrjoo, M., Khaji, N., Moharrami, H., and Bahreininejad, A. (2008). "Damage detection of
568 truss bridge joints using artificial neural networks." *Expert Systems with Applications*, 35(3),
569 1122–1131.
570 Nair, K. K., Kiremidjian, A. S., and Law, K. H. (2006). "Time series-based damage detection and
571 localization algorithm with application to the asce benchmark structure." *Journal of Sound and*
572 *Vibration*, 291(1), 349–368.
573 Pan, B., Qian, K., Xie, H., and Asundi, A. (2009). "Two-dimensional digital image correlation for
574 in-plane displacement and strain measurement: a review." *Measurement science and technology*,
575 20(6), 062001.

Pei, J.-S., Smyth, A., and Kosmatopoulos, E. (2004). "Analysis and modification of volterra/wiener neural networks for the adaptive identification of non-linear hysteretic dynamic systems." *Journal of Sound and Vibration*, 275(3-5), 693–718.

Robbins, H. and Monro, S. (1951). "A stochastic approximation method." *The annals of mathematical statistics*, 400–407.

Rojas, R. (2013). *Neural networks: a systematic introduction*. Springer Science & Business Media.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1988). "Learning representations by back-propagating errors." *Cognitive modeling*, 5(3), 1.

Sbarufatti, C., Manes, A., and Giglio, M. (2013). "Performance optimization of a diagnostic system based upon a simulated strain field for fatigue damage characterization." *Mechanical Systems and Signal Processing*, 40(2), 667–690.

Shahidi, S. G., Gulgec, N. S., and Pakzad, S. N. (2016). "Compressive sensing strategies for multiple damage detection and localization." *Dynamics of Civil Structures, Volume 2*, Springer, 17–22.

Shalev-Shwartz, S. and Ben-David, S. (2014). *Understanding machine learning: From theory to algorithms*. Cambridge University Press.

Shi, A. and Yu, X.-H. (2012). "Structural damage detection using artificial neural networks and wavelet transform." *2012 IEEE International Conference on Computational Intelligence for Measurement Systems and Applications (CIMSA) Proceedings*, IEEE, 7–11.

Shu, J., Zhang, Z., Gonzalez, I., and Karoumi, R. (2013). "The application of a damage detection method using artificial neural network and train-induced vibrations on a simplified railway bridge model." *Engineering structures*, 52, 408–421.

Simon, P. (2013). *Too big to ignore: the business case for big data*, Vol. 72. John Wiley & Sons.

Simonyan, K. and Zisserman, A. (2014). "Very deep convolutional networks for large-scale image recognition." *arXiv preprint arXiv:1409.1556*.

Sohn, H. and Farrar, C. R. (2001). "Damage diagnosis using time series analysis of vibration signals." *Smart materials and structures*, 10(3), 446.

603 Swamidas, A. and Chen, Y. (1995). "Monitoring crack growth through change of modal parameters."
604 *Journal of Sound and Vibration*, 186(2), 325–343.
605 Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and
606 Rabinovich, A. (2015). "Going deeper with convolutions." *Proceedings of the IEEE Conference*
607 *on Computer Vision and Pattern Recognition*, 1–9.
608 Theano Development Team (2016). "Theano: A Python framework for fast computation of mathe-
609 matical expressions." *arXiv e-prints*, abs/1605.02688.
610 Yam, L., Li, Y., and Wong, W. (2002). "Sensitivity studies of parameters for damage detection
611 of plate-like structures using static and dynamic approaches." *Engineering structures*, 24(11),
612 1465–1475.
613 Yao, R. and Pakzad, S. N. (2012). "Autoregressive statistical pattern recognition algorithms for
614 damage detection in civil structures." *Mechanical Systems and Signal Processing*, 31(Supplement
615 C), 355 – 368.
616 Yao, R., Pakzad, S. N., and Venkitasubramaniam, P. (2016). "Compressive sensing based structural
617 damage detection and localization using theoretical and metaheuristic statistics." *Structural*
618 *Control and Health Monitoring*.
619 Zapico, J., Gonzalez, M., and Worden, K. (2003). "Damage assessment using neural networks."
620 *Mechanical Systems and Signal Processing*, 17(1), 119–125.
621 Zeiler, M. D. and Fergus, R. (2014). "Visualizing and understanding convolutional networks."
622 *European Conference on Computer Vision*, Springer, 818–833.
623 Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. (2016). "Understanding deep learning
624 requires rethinking generalization." *arXiv preprint arXiv:1611.03530*.

List of Tables

TABLE 1. Computation performance of the proposed methodology

		K80 GPUs		CPU	
	Task	Batch of Samples	One Sample	Batch of Samples	One Sample
Training	Shared Layers+Detection	6	4	30	5
Time (ms)	Shared Layers+Localization	9	5	45	14
	Localization	4	2	20	12
Testing	Shared Layers+Detection	3	2	8	2
Time (ms)	Shared Layers+Localization	4	2	13	6
	Localization	2	1	7	5

List of Figures

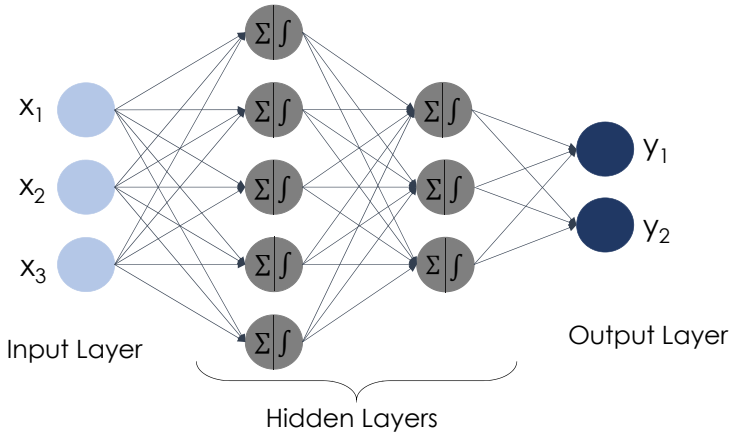

Fig. 1. A DNN with two hidden layers.

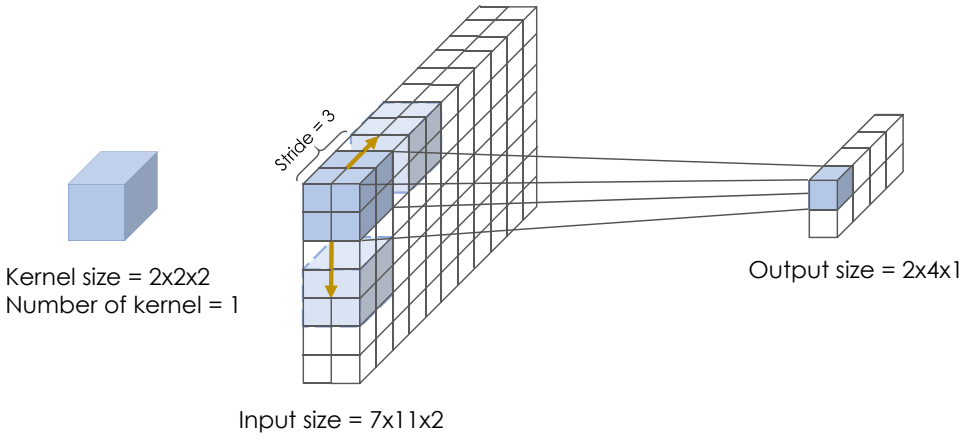

Fig. 2. An example of convolution operation.

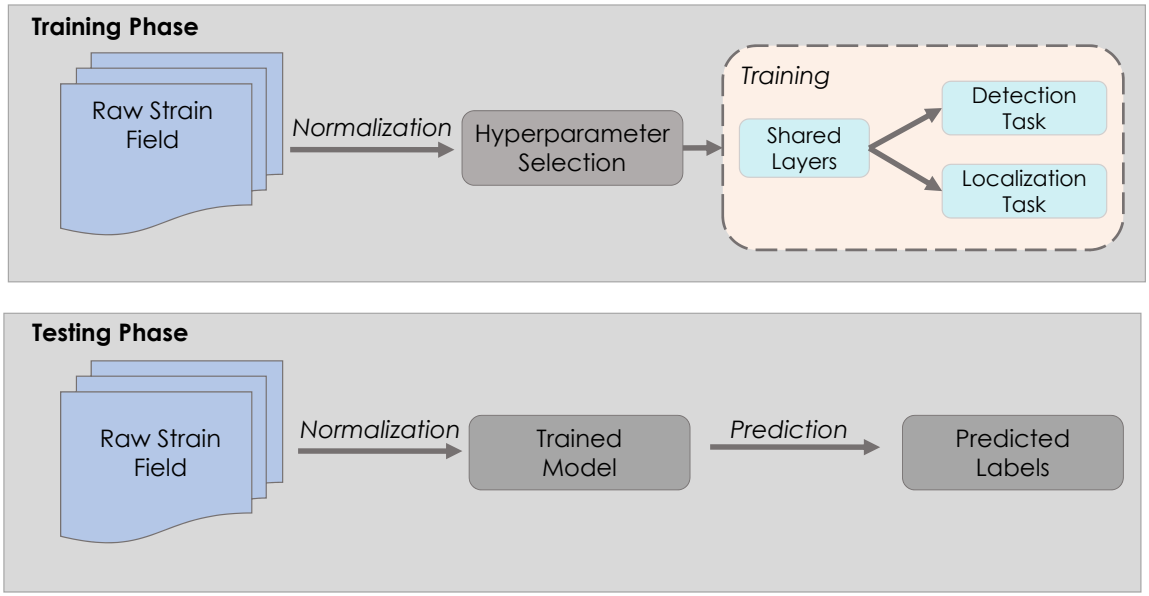

Fig. 3. Overview of the proposed methodology.

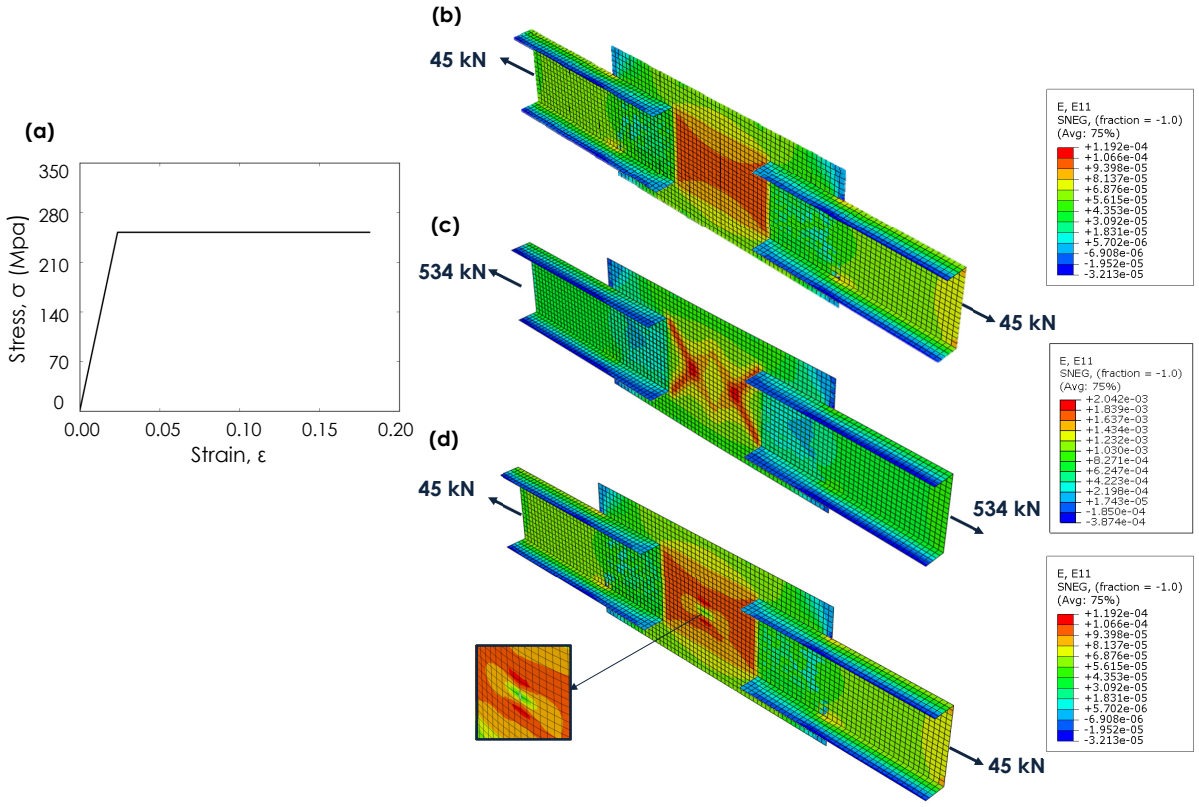

Fig. 4. (a) Material behavior; The setup of the (b) healthy, (c) inelastic and (d) single damaged gusset-plate.

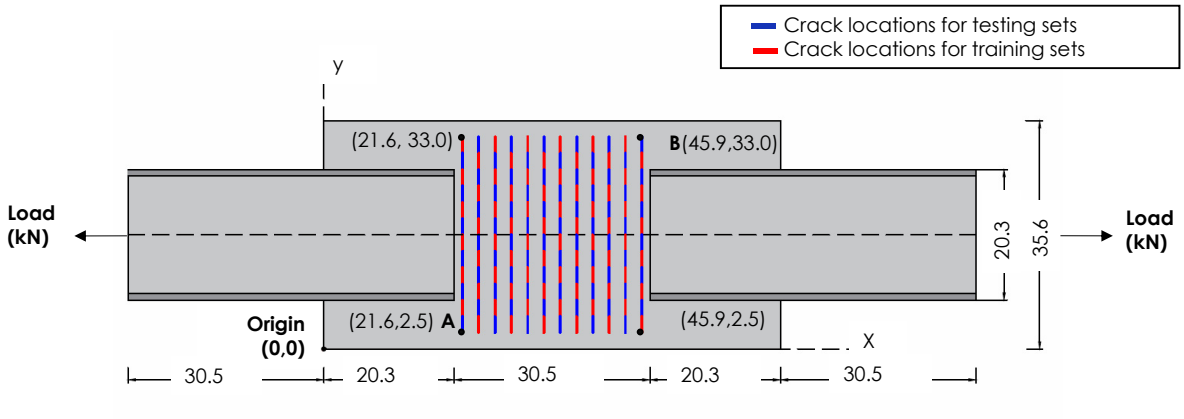

Fig. 5. The damage locations for training and test datasets.

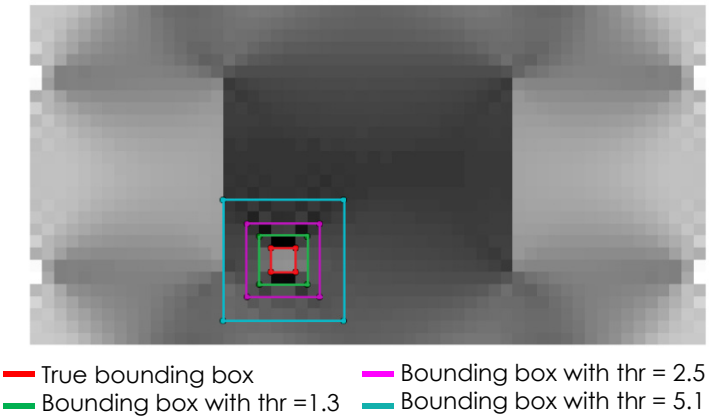

Fig. 6. Threshold values adopted for the localization task.

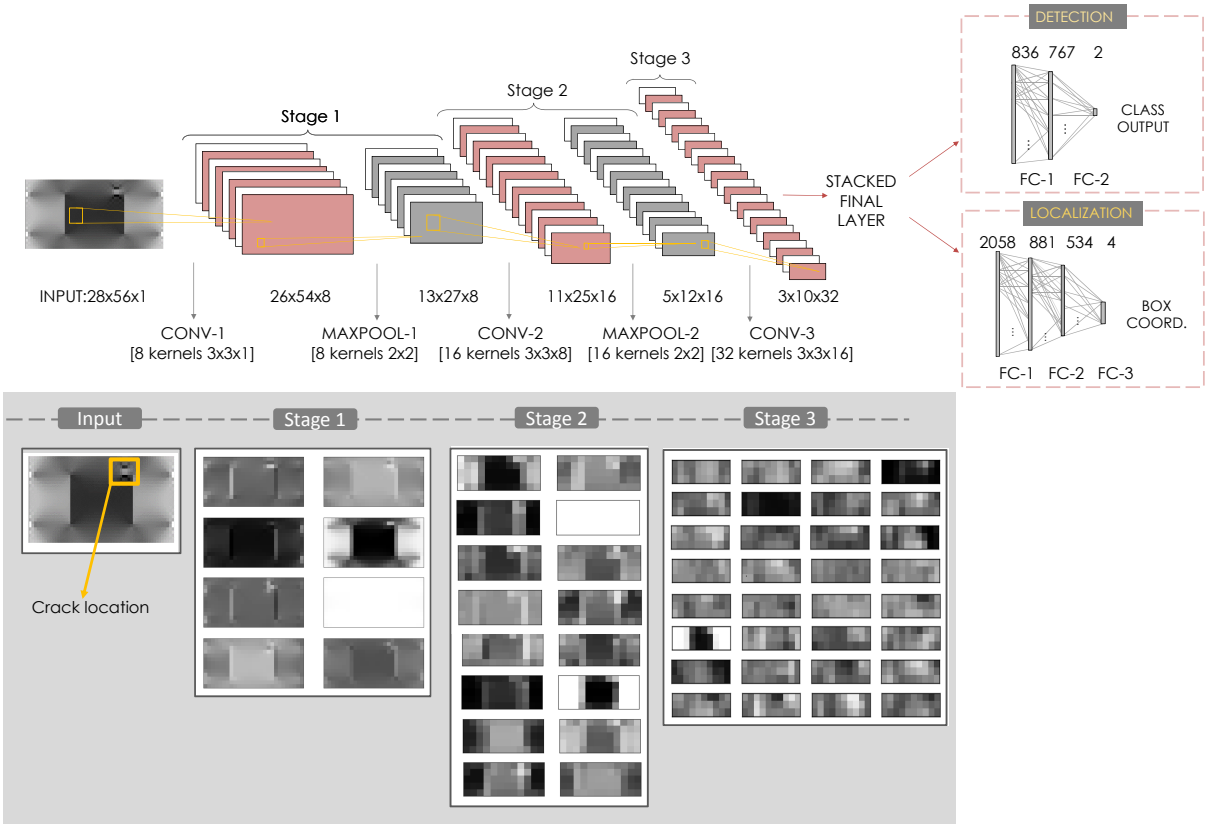

Fig. 7. Proposed CNN Architecture.

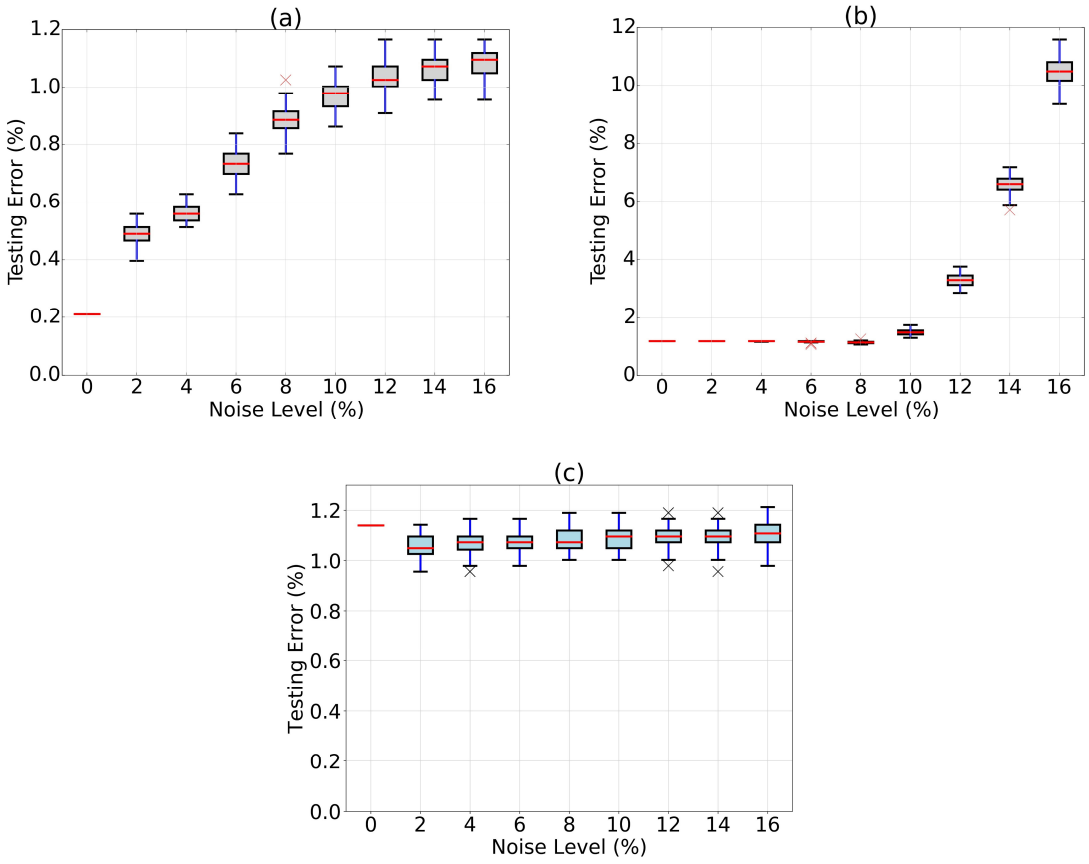

Fig. 8. Sensitivity analysis of detection task trained with (a) Dataset 1, (b) Dataset 2 and (c) Dataset 3.

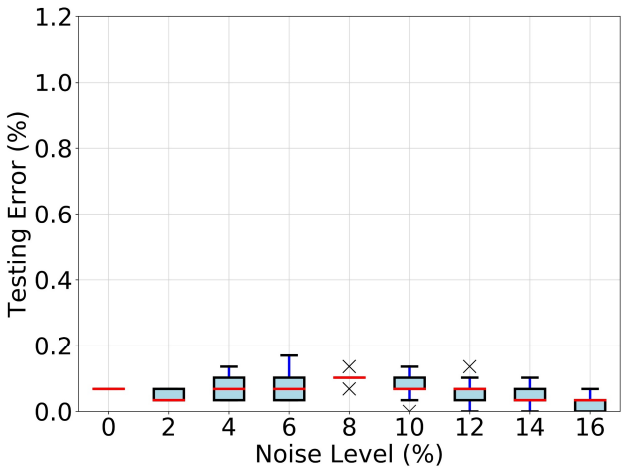

Fig. 9. Sensitivity analysis of the detection task for the crack size 5.1 cm.

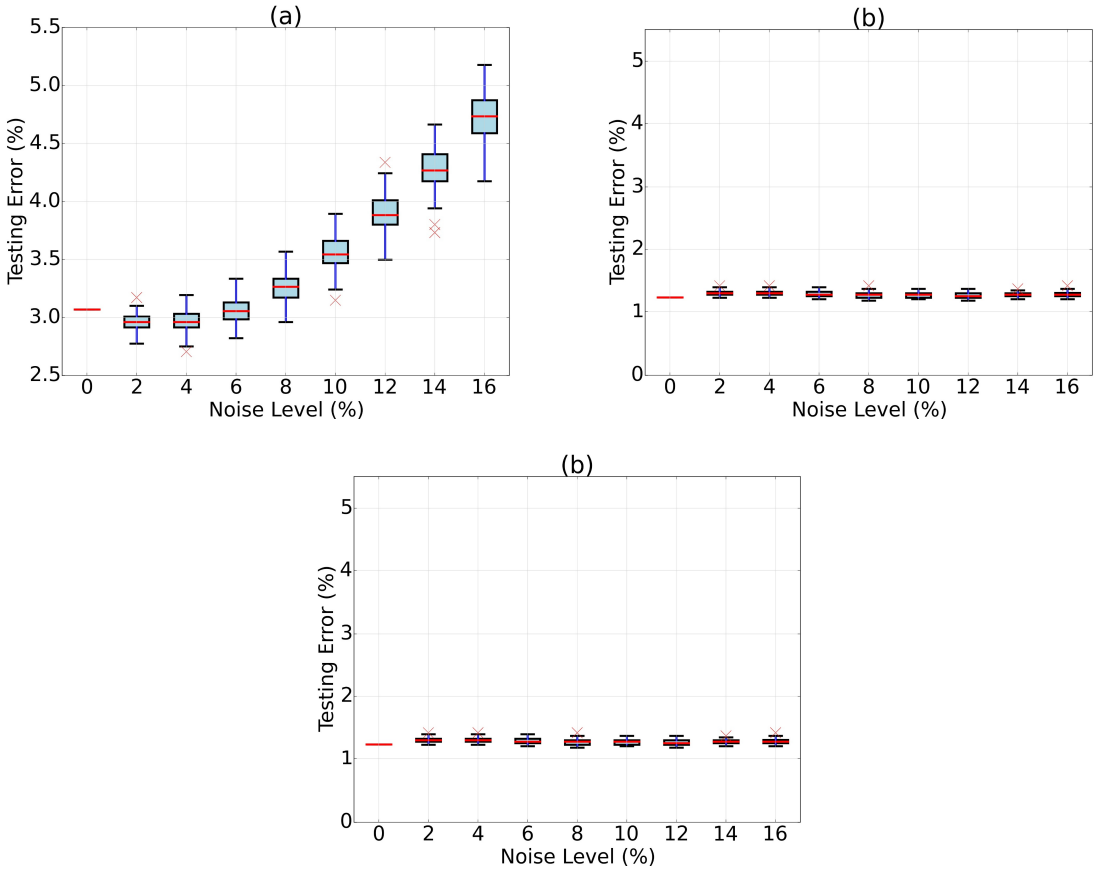

Fig. 10. Sensitivity analysis of localization task for thresholds (a) thr = 1.3 cm, (b) thr = 2.5 cm, (c) thr = 5.1 cm .

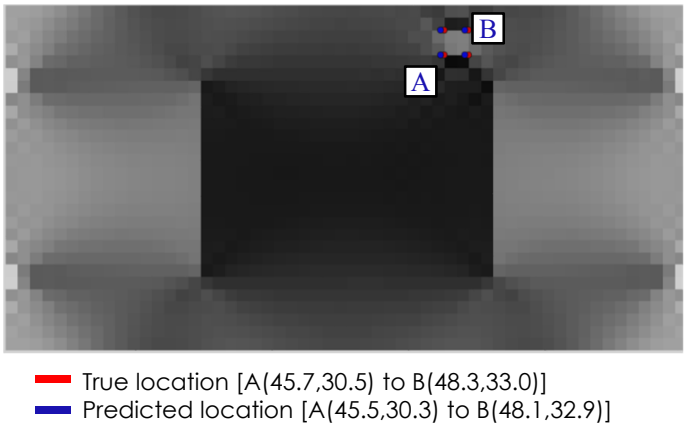

Fig. 11. An example of correct bounding box estimation.