Convolutional Neural Network Approach for Robust Structural Damage Detection and Localization

Nur Sila Gulgec, S.M.ASCE[1]; Martin Takáč[2]; and Shamim N. Pakzad, A.M.ASCE[3]

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 brings 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. The CNN technique has the ability to discover abstract features and complex classifier boundaries that are able to distinguish various attributes of the problem. In this paper, a CNN topology was designed to classify simulated damaged and healthy cases and localize the damage when it exists. The performance of the proposed technique was evaluated through the finite-element simulations of undamaged and damaged structural connections. Samples were trained by using strain distributions as a consequence of various loads with several different crack scenarios. Completely new damage setups were introduced to the model during the testing process. Based on the findings of the proposed study, the damage diagnosis and localization were achieved with high accuracy, robustness, and computational efficiency. **DOI: 10.1061/(ASCE)CP.1943-5487.0000820.**

Introduction

Structural systems are subjected to damage and deterioration during their service life due to 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 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, 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 to a mechanical model of the behavior of the structural systems and components, which constrains these methods in two aspects: (1) the methods are limited in their scope, depending on the feature that they use for damage identification; and (2) they are often overwhelmed by big data when damage features are computationally complex. For example, 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 shows the difficulty in dealing with large data sets and the limitation of the identification methods based on selected features.

The main challenge of the damage identification originates 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, was proposed in the 1940s (Flood and Kartam 1994) and is designed such that it can learn from data without a need of feature design process. Since then, it has been practiced in many disciplines including SHM to diagnose damage 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

[1]Graduate Student, Dept. of Civil and Environmental Engineering, Lehigh Univ., 117 ATLSS Dr., Imbt Labs, Bethlehem, PA 18015 (corresponding author). Email: sgulgec@gmail.com

[2]Assistant Professor, Dept. of Civil and Environmental Engineering, Lehigh Univ., 117 ATLSS Dr., Imbt Labs, Bethlehem, PA 18015.

[3]Associate Professor, Dept. of Industrial and Systems Engineering, Lehigh Univ., 200 West Packer Ave., Harold S. Mohler Laboratory, Bethlehem, PA 18015.

Note. This manuscript was submitted on March 11, 2018; approved on September 13, 2018; published online on January 30, 2019. Discussion period open until June 30, 2019; separate discussions must be submitted for individual papers. This paper is part of the ***Journal of Computing in Civil Engineering***, © ASCE, ISSN 0887-3801.

with the improvements in computing power and the introduction of large representative training data sets (Gu et al. 2015). Exploiting the opportunities hidden in big data, deep neural networks (DNNs) (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 the SHM field. Abdeljaber et al. (2017) used a one-dimensional convolutional neural network (CNN) to extract damage features from raw acceleration data and Cha et al. (2017) used raw images taken from structure to perform deep learning–based detection of visible cracks only.

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 robustness 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 Gulgec et al. (2017b), damage diagnosis was performed by using CNN fed through the strain distributions of a structural connection. In this paper, this idea is expanded by performing a CNN-based methodology for both damage identification and localization with comprehensive noise sensitivity analysis. The proposed methodology shares the front-end layers of a deep convolutional network for both identification and localization tasks. Then, customized back-end layers are constructed that are specialized for both tasks. Automatically extracted features in the front-end 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. Finally, this approach makes real-time damage identification possible, thanks to (1) front-end layer sharing, (2) CNN's shared parameterization, and (3) parallel architecture of graphics processing units (GPUs).

The rest of the paper is organized as follows. First, a review of relevant studies and a brief overview of CNNs is provided in the "Introduction" and "Background on Deep Learning"; then the proposed methodology is described in "Proposed Methodology." The performance and robustness of the proposed approach are evaluated by numerical validation in "Numerical Validation" and "Results and Discussion." Conclusions and future directions are given in the "Conclusion."

Background on Deep Learning

Deep Neural Networks

Machine learning (ML) is gradually evolved from pattern recognition and learning theory in artificial intelligence (Alpaydin 2014).

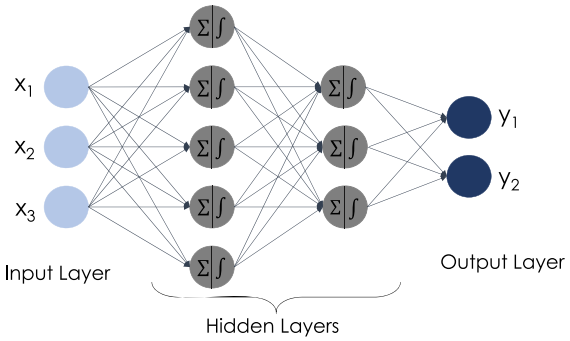

Fig. 1. DNN with two hidden layers.

In 1959, Arthur Samuel (1959) 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 that is then used to make data-driven predictions or decisions.

DNNs 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 nonlinear transformations (LeCun et al. 2015). Fig. 1 shows an example four-layer DNN that 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 Fig. 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 that 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. Softmax 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 follows (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

CNNs are one of the most widely used types of deep neural networks. The framework of CNN was first proposed by 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 subsampling (LeCun et al. 1998). Passing the same set of units all over the input allows extracting multiple feature maps. In this case, the feature map shifts the same amount the 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. Finally, spatial subsampling helps reduce the resolution of the feature maps and prevent the sensitivity of the outputs under shifts and rotations.

CNNs receive the input as three-dimensional (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 (CONV) layer parameters are learnable filters in which each filter (weights or kernels) has spatially small width and height shared in the full depth of the input. While sliding these weights, the 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. Fig. 2 shows a convolution operation for an input size (I) of $7 \times 11 \times 2$. In the figure, a kernel size (K) of $2 \times 2 \times 2$ 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 (POOL) layer performs a downsampling operation in the feature maps using maximum, average, or sum operations. The 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 (FC) layers operate on the stacked convolutional or pooling layer outputs and compute the weighted sum of inputs with a nonlinear mapping as described in the overview of DNNs.

Proposed Methodology

Overview

This section gives a general map of the proposed technique. As shown in Fig. 3, the methodology consists of training and testing phases. The 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 that improves the performance of the 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. The detection task determines the existence of damage where it is treated as a classification problem (i.e., 0 for undamaged and 1 for damaged). The localization task treats the case as a 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 front-end layers avoid having two separate networks, provide more efficient learning, and have shorter training time and lower computation cost.

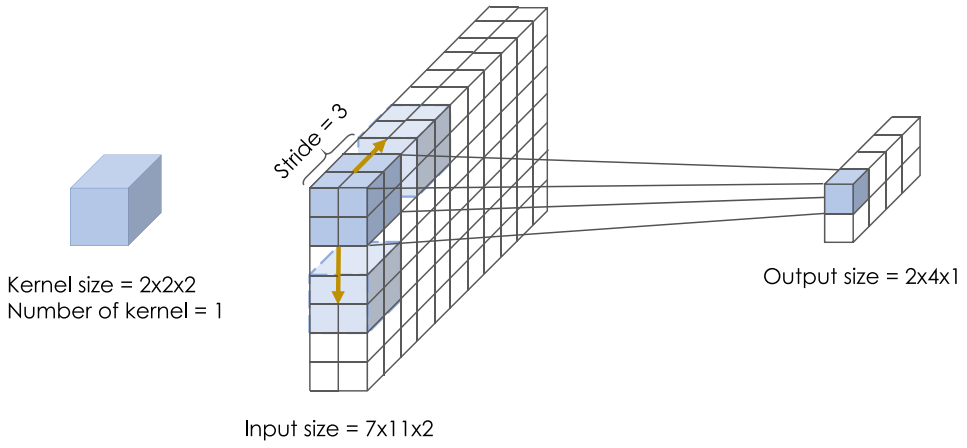

Fig. 2. Example of convolution operation.

J. Comput. Civ. Eng., 2019, 33(3): 04019005

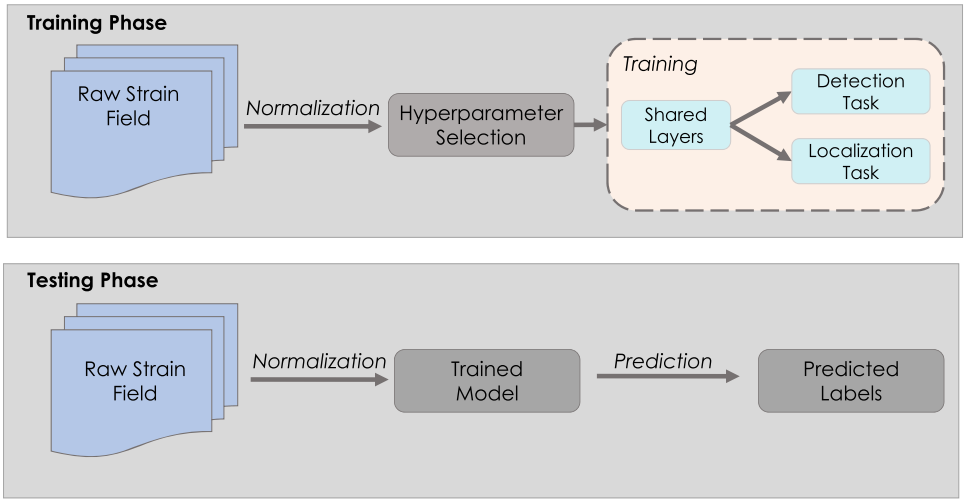

Fig. 3. Overview of the proposed methodology.

The trained model parameters are stored to be used in the 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 CONV, POOL, and 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 was implemented for both damage detection and localization tasks (Li et al. 2016).

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

Training

The training process was comprised of two phases: feedforward and back-propagation (Rojas 2013). The feedforward process evaluated the prediction function for given input instances. Then the back-propagation step adjusted 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 were calculated with SGD, the detection and localization parameters were updated with the learning rates η_{det} and η_{loc}, respectively. Overfitting was prevented by monitoring the validation data set performance in every complete forward and backward pass (epoch). When architecture performance was improved sufficiently on the validation data set, the training process was stopped.

Weight Initialization

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

Prediction Functions

The feedforward step evaluated different prediction functions for detection and localization tasks. This study employed the softmax 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 was estimated by selecting the maximum probability of the softmax function defined as follows:

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

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

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

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

Loss Functions

The proposed model adopted two separate loss functions for the detection and localization tasks. The diagnosis part employed the negative log-likelihood function, where optimal architecture parameters θ^* were learned by maximizing the likelihood of the data set. On the other hand, the localization task calculated the loss between the predicted and true bounding box with the ℓ_2 loss function

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

where z = predicted bounding box; and N = batch size (i.e., number of training samples in one feedforward pass). During the regressor training, the bounds of the boxes were updated based on the conditions of Eqs. (7)–(10). If these criteria were satisfied with the predefined threshold values, the localization was marked as correct localization

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

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

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

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

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

Numerical Validation

Data Preparation

The 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 were simulated by using Abaqus shell elements. The modeled connection consisted of two C8 × 11.5 channels welded to a steel plate with dimensions of 71 × 36 × 0.6 cm (28 × 14 × 1/4 in.) as visualized in Fig. 4. Each channel member was 51 cm long and had 20-cm overlap with the main gusset plate. The finite-element model adopted the mesh size of 1.3 cm (0.5 in.). The behavior of the steel was introduced as elastic–perfectly plastic material with a yield strength of 250 MPa. As can be seen from Fig. 4, stress gradients occurred at the crack tips as well as the central part of the plate when it becomes a plastic region. Strain distribution in the direction of loading is represented by 28 × 56 × 1 tensors and was used to feed the CNN architecture after being normalized by its absolute maximum value.

Training, validation, and test data sets were formed by modeling different loading cases, damage scenarios, and noise levels. The load was selected from uniformly distributed load ~U[−445 kN (compression), 534 kN (tension)] and applied to the end of the channel members. The damage in the gusset plate was simulated as 2.5-cm-long cracks, which are the smallest crack size given the mesh size. The crack locations were 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 Fig. 5. In order to assess the approach with completely unseen damaged samples, none of the coordinates of the training set was used in the testing samples.

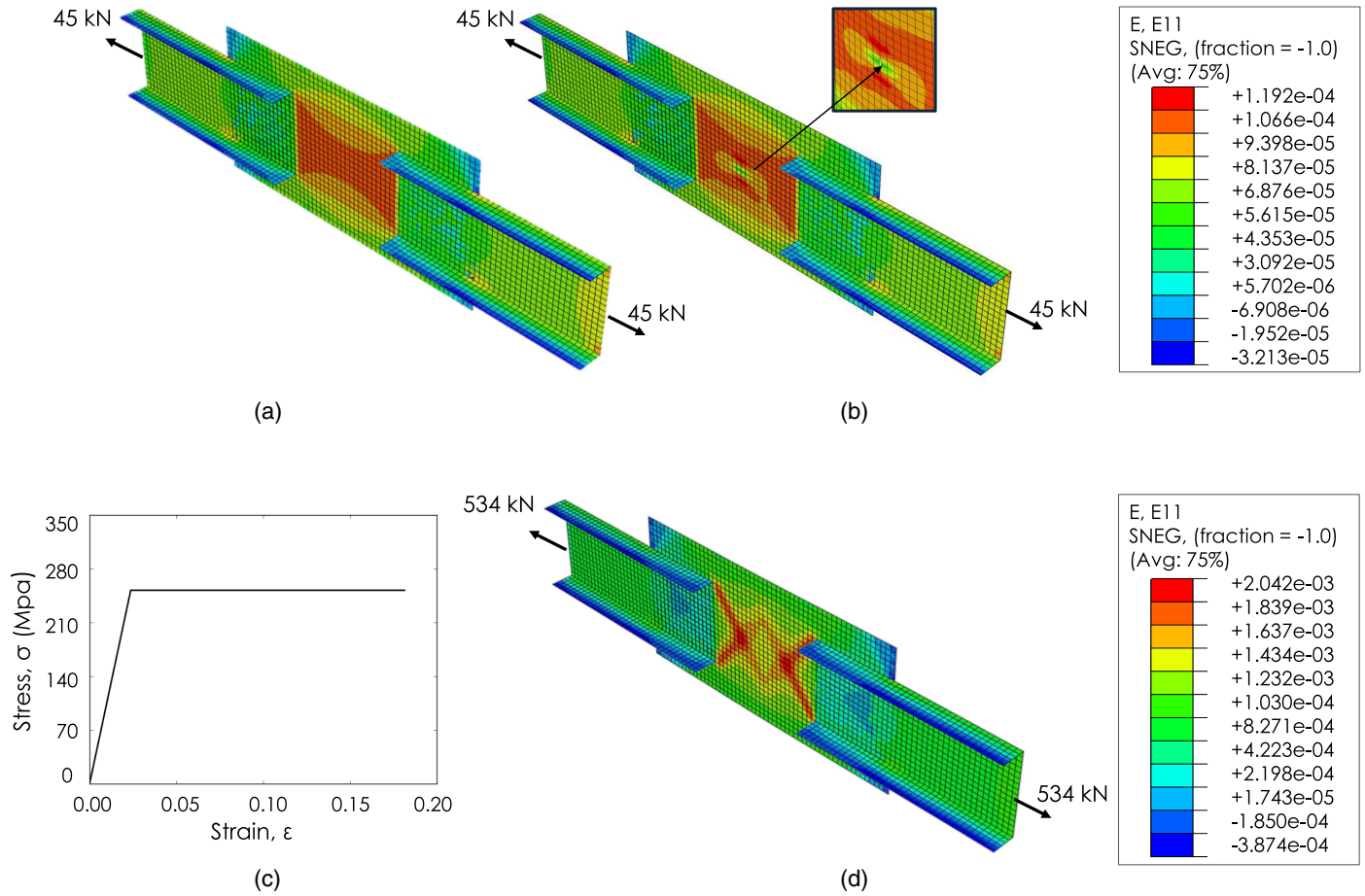

Fig. 4. Setup of the (a) healthy, and (b) single-damaged gusset plate; and (c) material behavior, and (d) inelastic behavior of the plate.

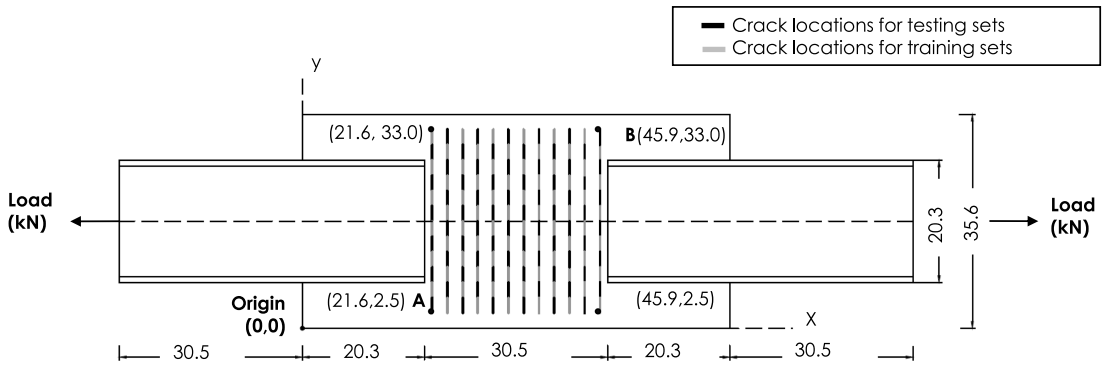

Fig. 5. Damage locations for training and test data sets.

The uncertainty in the measurement process was 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) were generated to compute the influence of the noise on CNN architecture performance.

The crack coordinates of the single-damaged samples were also collected for the localization task. Crack location was 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 was 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) and (21.6, 5.0), the bounding box is defined as [20.3, 2.5, 22.9, 5.0]. For the healthy samples, bounding box was set to [0, 0, 0, 0].

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

Hyperparameters

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

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

Training and Proposed Architecture

Training was implemented by using a Python library called Theano to optimize the mathematical expressions consisting of multidimensional arrays (Theano Development Team 2016). Higher performance was achieved by using NVIDIA (Holmdel, New Jersey) Tesla K80 GPUs, which enabled parallelism for data-intensive calculations.

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

Fig. 7 shows the proposed architecture as a result of the hyperparameter search mechanism. The network consisted of three convolutional layers followed by two separate fully connected layers for detection and localization tasks. The detection part classified $28 \times 56 \times 1$ inputs as healthy or damaged, whereas the localization part predicted the bounding box of the crack area. The convolutional layers received the input layer and passed them through a filter size of (3×3). As a result of these CONV layers, the network formed 8, 16, and 32 feature maps. The max-pooling operation was implemented right after the first and second convolution layer. A max-pool size of (2×2) with a stride of 2 was

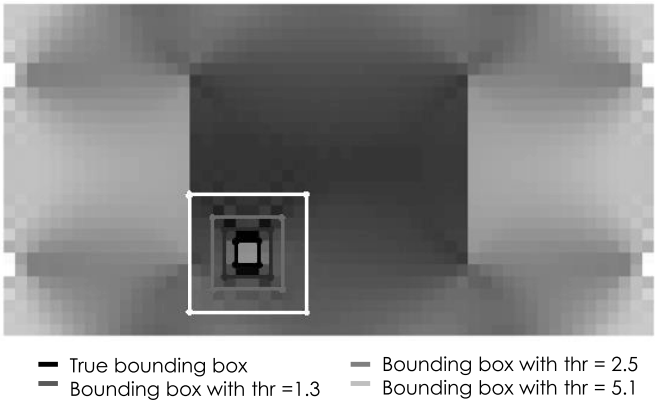

Fig. 6. Threshold values adopted for the localization task.

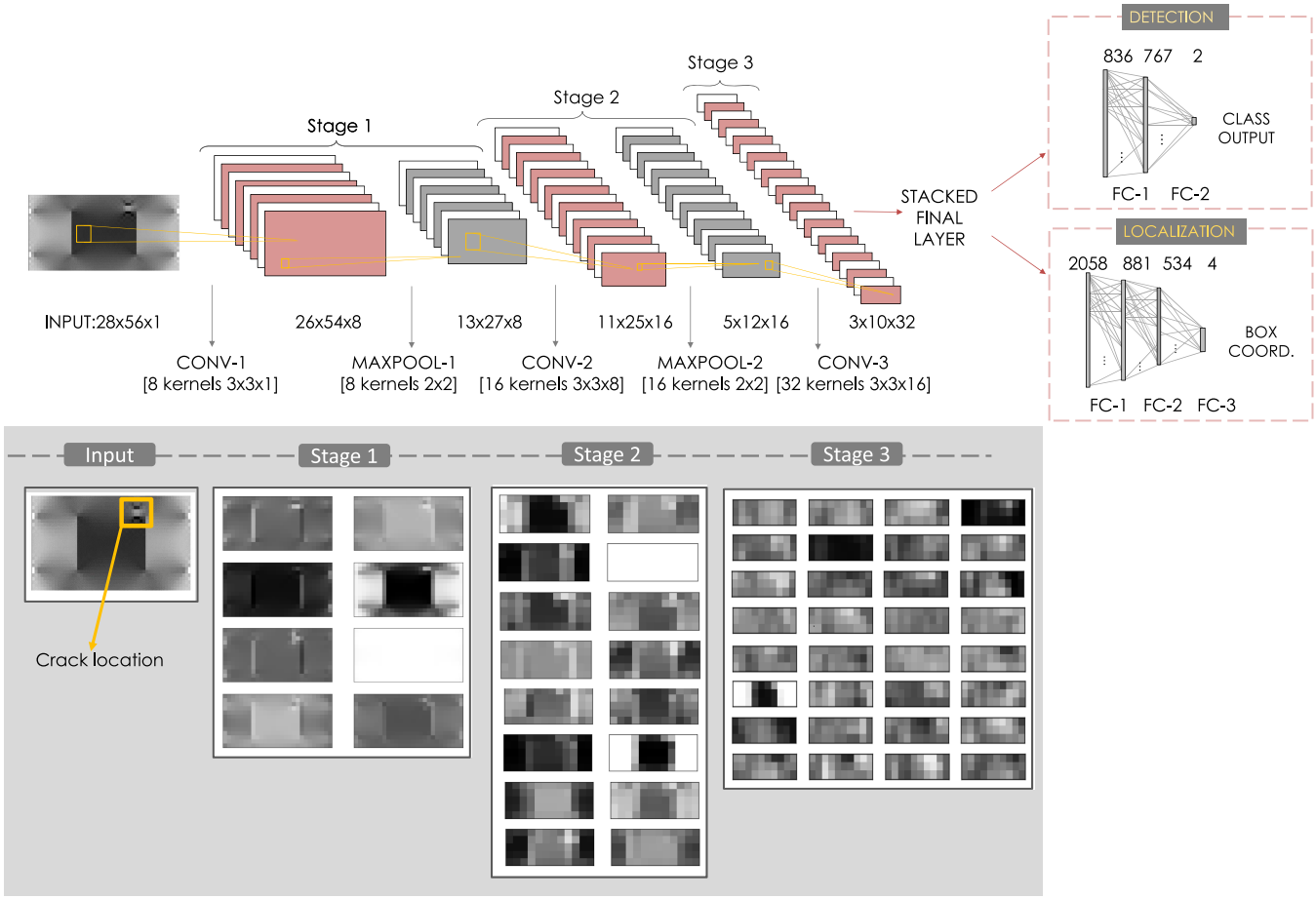

Fig. 7. Proposed CNN architecture.

used for POOL layers. The feature maps of the last convolutional layer were 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 [2058, 881, 534] for the localization task. The learning rate of $\eta_{det} = 0.0451$ and $\eta_{loc} = 0.0026$ were used for the detection and localization parts, respectively.

As mentioned previously, 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 Fig. 7. The activations were normalized to have the scale between 0 and 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 Stages 1 and 2. After the 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 data sets were prepared with both consisting of 6,000 undamaged and 6,000 damaged samples. Data Set 2 was formed by only noise-free samples and Data Set 3 was selected from a subset of Data Set 1. Hyperparameter search was performed for these two data sets for fair comparison. Trained models were then tested with samples including a variation of different noise levels (0% or noise-free, 2%, 4%, 6%, 8%, 10%, 12%, 14%, and 16%) 100 times. Proposed network topologies for two additional training processes are listed as follows:

- Training of Data Set 2: The network is trained with Data Set 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 (3×3) with kernel numbers of 2 and 4. Max-pool size of (2×2) 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 Data Set 3: The selected network for Data Set 3 includes two CONV layers with the filter size of (3×3) with kernel numbers of 8 and 32. Similar to the first case, a max-pool size of (2×2) with a stride of 2 is used for POOL layers. The network has the two FC layers size of [2,477, 804] after shared layers. The learning rate of $\eta_{det} = 0.069$ is adopted.

The sensitivity analysis of three training cases is visualized in Fig. 8. Fig. 8(b) presents the testing performance of Data Set 2, which had the worst testing performance among the three cases. Although the testing error was 1.19% for lower noise levels, it reached around 12% under the noise level of 16%. It is noticeable

J. Comput. Civ. Eng., 2019, 33(3): 04019005

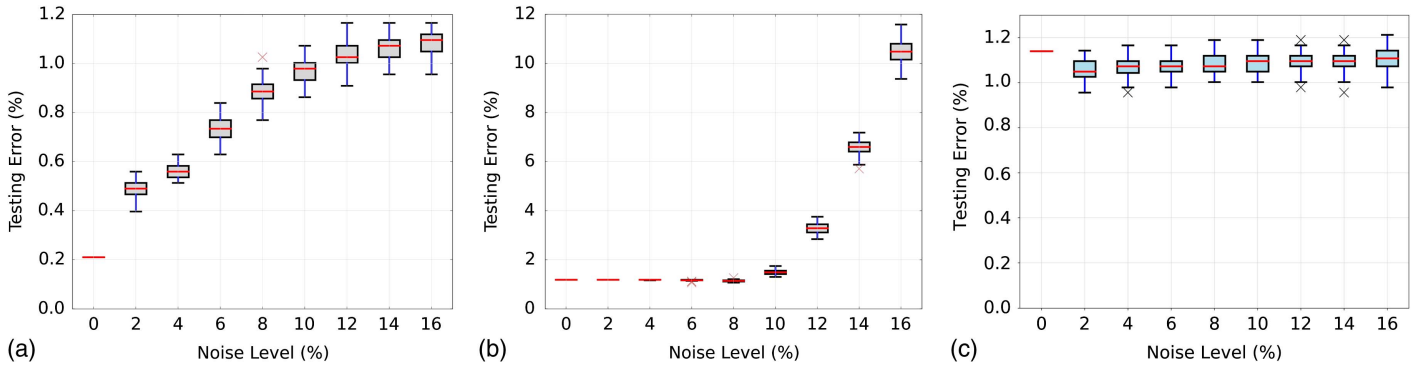

Fig. 8. Sensitivity analysis of detection task trained with (a) Data Set 1; (b) Data Set 2; and (c) Data Set 3.

that the error rate exponentially increases with the increase in the noise levels.

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

Fig. 8(a) illustrates the best testing performance from the given training cases. According to the figure, the proposed architecture identified 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, the 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 designed carefully. In order to evaluate this characteristic, the performance of the proposed method was assessed with a larger crack size. A total of 3,000 samples with a crack size of 5.1 cm were tested for the detection task. As shown in Fig. 9, although samples with crack size of 5.1 cm were not included in the training data set, the testing accuracy was almost perfect. The filters used in the architecture managed to highlight the cracked region.

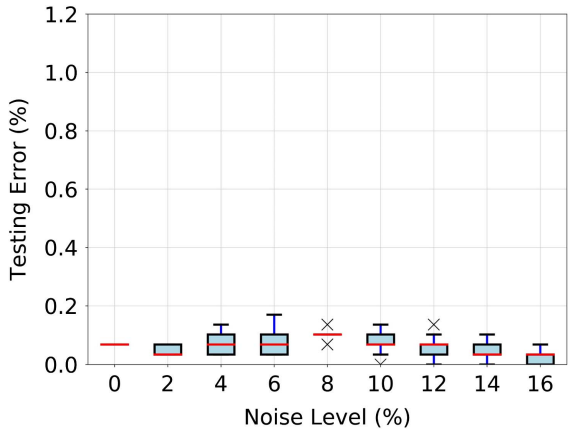

Fig. 9. Sensitivity analysis of the detection task for the crack size 5.1 cm.

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

Localization Task

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

Fig. 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 Fig. 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).

Fig. 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 was almost 1% under all levels of noise for both threshold values 2.5 and 5.1 cm as shown in Figs. 10(b and c).

Computational Performance

The computational performance of the case study was evaluated on an Intel (Hillsboro, Oregon) 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 was considered. The computation times for shared layers and detection task, shared layers and localization task, and only localization task are compared in Table 1.

As illustrated in Table 1, the testing time for all tasks was less than 20 ms for both hardwares. A video stream input with 25 frames per second would give a 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 the real-time

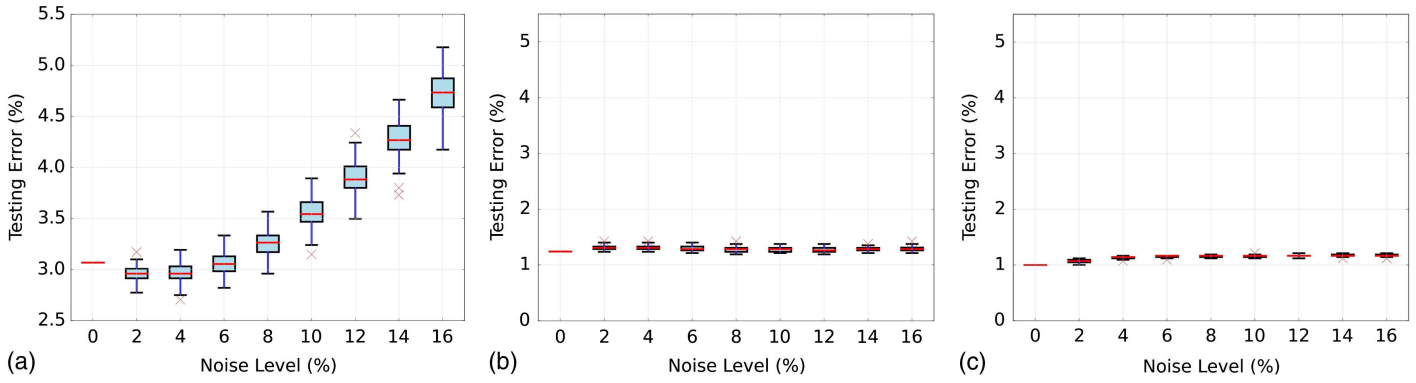

Fig. 10. Sensitivity analysis of localization task for thresholds: (a) thr = 1.3 cm; (b) thr = 2.5 cm; and (c) thr = 5.1 cm.

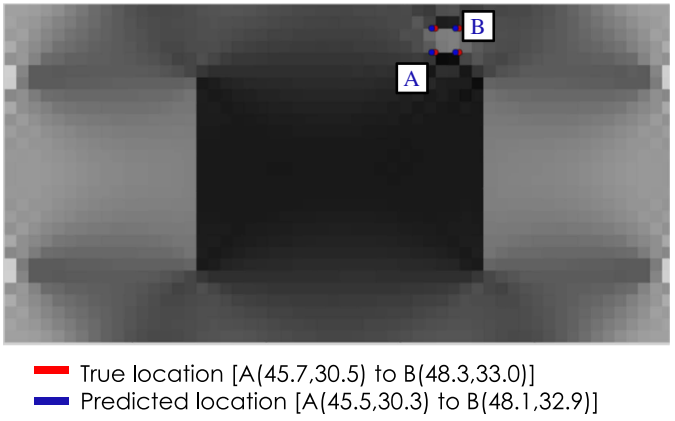

Fig. 11. Example of correct bounding box estimation.

requirement for the testing phase. In addition, as mentioned previously, the proposed architecture reduces the computational needs by exploiting the shared feature extractor for the detection and localization tasks. The computation time for only the localization task was almost half of the computation time for the shared layers and 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 platelike structures in the literature. Finding the 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 include the modal strain energy index where the index is a ratio of summations of fractional energies of elements before and after damage (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 a 60×40 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 the noise-free case and 8.5% for the 3% noise-to-signal (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 in which damage was characterized by the probabilistic behavior. As an example, the research presented in Hasni et al. (2017) extracted the probability density function of strain time histories of a gusset plate and identified cracks by using the 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 principal component analysis were used to generate damage features, which are then utilized to feed the ANN structure. The study was evaluated on a simulated lap-joint structure and validated by

Table 1. Computation performance of the proposed methodology

		K80 GPUs		CPU	
Phase	Task	Batch of samples time (ms)	One sample time (ms)	Batch of samples time (ms)	One sample time (ms)
Training	Shared layers + detection	6	4	30	5
	Shared layers + localization	9	5	45	14
	Localization	4	2	20	12
Testing	Shared layers + detection	3	2	8	2
	Shared layers + localization	4	2	13	6
	Localization	2	1	7	5

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 the damage index. The associated damage index map was validated by using the simulation of a 60×50 cm panel with rivets. The detection accuracies were obtained as the average of the output values from 50 ANNs. For the noise-free case, detection accuracies were reported as 92.5% and 95% for 6- and 8-cm cracks, respectively. The accuracies dropped with the increase in noise level: for example, detection accuracies for 12% additive Gaussian noise case were 25% for a 6-cm crack and 90% for a 8-cm crack, respectively.

The majority of the 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 previously. First, traditional approaches need measurements from both the baseline and unknown state of structures. Second, such approaches consist of 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 data sets.

Conclusion

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

In our study, the abstract feature maps were discovered with CNN technique to classify damaged and healthy cases modeled through analytic simulations. The computational needs of the methodology were 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 accomplished high accuracy, robustness, and computational efficiency, which holds great potential for real-time damage diagnosis and localization challenge. Also, the performance of deep neural networks highly depends on the training data set. The selection of training data set requires the representation of as many cases as possible to predict test cases accurately. The results show that CNN architecture performs with higher accuracy and robustness when the training data set is formed with noise-free and noisy data. Consequently, training data sets 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 damage for multiple damage cases, and (3) test the designed network on the real experimental setup.

Acknowledgments

Research funding is partially provided by the National Science Foundation through Grant No. CMMI-1351537 by the 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 Grants CCF-1618717 and CMMI-1663256.

References

Abdeljaber, O., O. Avci, S. Kiranyaz, M. Gabbouj, and D. J. Inman. 2017. "Real-time vibration-based structural damage detection using one-dimensional convolutional neural networks." *J. Sound Vib.* 388: 154–170. https://doi.org/10.1016/j.jsv.2016.10.043.

Alavi, A. H., H. Hasni, N. Lajnef, K. Chatti, and F. Faridazar. 2016. "An intelligent structural damage detection approach based on self-powered wireless sensor data." *Autom. Constr.* 62: 24–44. https://doi.org/10.1016/j.autcon.2015.10.001.

Alpaydin, E. 2014. *Introduction to machine learning*. Cambridge, MA: MIT Press.

Bayissa, W., and N. Haritos. 2007. "Structural damage identification in plates using spectral strain energy analysis." *J. Sound Vib.* 307 (1–2): 226–249. https://doi.org/10.1016/j.jsv.2007.06.062.

Bishop, C. M. 2006. *Pattern recognition and machine learning: Information science and statistics*, 209. New York: Springer.

Cha, Y.-J., W. Choi, and O. Büyüköztürk. 2017. "Deep learning-based crack damage detection using convolutional neural networks." *Comput.-Aided Civ. Infrastruct. Eng.* 32 (5): 361–378. https://doi.org/10.1111/mice.12263.

Choi, S., S. Park, S. Yoon, and N. Stubbs. 2005. "Nondestructive damage identification in plate structures using changes in modal compliance." *NDT & E Int.* 38 (7): 529–540. https://doi.org/10.1016/j.ndteint.2005.01.007.

Cornwell, P., S. W. Doebling, and C. R. Farrar. 1999. "Application of the strain energy damage detection method to plate-like structures." *J. Sound Vib.* 224 (2): 359–374. https://doi.org/10.1006/jsvi.1999.2163.

Deng, J., W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. 2009. "Imagenet: A large-scale hierarchical image database." In *Proc., IEEE Conf. on Computer Vision and Pattern Recognition*, 248–255. New York: IEEE.

Elkordy, M., K. Chang, and G. Lee. 1993. "Neural networks trained by analytically simulated damage states." *J. Comput. Civ. Eng.* 7 (2): 130–145. https://doi.org/10.1061/(ASCE)0887-3801(1993)7:2(130).

Fang, X., H. Luo, and J. Tang. 2005. "Structural damage detection using neural network with learning rate improvement." *Comput. Struct.* 83 (25): 2150–2161. https://doi.org/10.1016/j.compstruc.2005.02.029.

Farrar, C. R., and K. Worden. 2007. "An introduction to structural health monitoring." *Philos. Trans. R. Soc. London Ser. A* 365 (1851): 303–315. https://doi.org/10.1098/rsta.2006.1928.

Flood, I. 2008. "Towards the next generation of artificial neural networks for civil engineering." *Adv. Eng. Inf.* 22 (1): 4–14. https://doi.org/10.1016/j.aei.2007.07.001.

Flood, I., and N. Kartam. 1994. "Neural networks in civil engineering. II: Systems and application." *J. Comput. Civ. Eng.* 8 (2): 149–162. https://doi.org/10.1061/(ASCE)0887-3801(1994)8:2(149).

Fujimaki, R., T. Yairi, and K. Machida. 2005. "An approach to spacecraft anomaly detection problem using kernel feature space." In *Proc., 11th ACM SIGKDD Int. Conf. on Knowledge Discovery in Data Mining*, 401–410. New York: Association for Computing Machinery.

Glorot, X., and Y. Bengio. 2010. "Understanding the difficulty of training deep feedforward neural networks." In Vol. 9 of *Proc., 13th Int. Conf. on Artificial Intelligence and Statistics*, 249–256. Sardinia: Proceedings of Machine Learning Research.

Gu, J., Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, and G. Wang. 2015. "Recent advances in convolutional neural networks." Preprint, submitted December 22, 2015. http://arXiv.org/abs/1512.07108.

Gul, M., and F. N. Catbas. 2009. "Statistical pattern recognition for structural health monitoring using time series modeling: Theory and experimental verifications." *Mech. Syst. Sig. Process.* 23 (7): 2192–2204. https://doi.org/10.1016/j.ymssp.2009.02.013.

Gulgec, N. S., G. S. Shahidi, T. J. Matarazzo, and S. N. Pakzad. 2017a. "Current challenges with bigdata analytics in structural health

monitoring." Vol. 7 of *Structural health monitoring and damage detection*, 79–84. New York: Springer.
Gulgec, N. S., S. G. Shahidi, and S. N. Pakzad. 2016. "A comparative study of compressive sensing approaches for a structural damage diagnosis." In *Proc., Geotechnical and Structural Engineering Congress 2016*, 1910–1919. Reston, VA: ASCE.
Gulgec, N. S., M. Takáč, and S. N. Pakzad. 2017b. "Structural damage detection using convolutional neural networks." Vol. 3 of *Model validation and uncertainty quantification*, 331–337. New York: Springer.
Hadzima-Nyarko, M., E. K. Nyarko, and D. Morić. 2011. "A neural network based modelling and sensitivity analysis of damage ratio coefficient." *Expert Syst. Appl.* 38 (10): 13405–13413. https://doi.org/10.1016/j.eswa.2011.04.169.
Hasni, H., A. H. Alavi, P. Jiao, and N. Lajnef. 2017. "Detection of fatigue cracking in steel bridge girders: A support vector machine approach." *Arch. Civ. Mech. Eng.* 17 (3): 609–622. https://doi.org/10.1016/j.acme.2016.11.005.
He, K., X. Zhang, S. Ren, and J. Sun. 2015. "Deep residual learning for image recognition." Preprint, submitted December 10, 2015. http://arXiv.org/abs/1512.03385.
Katsikeros, C. E., and G. Labeas. 2009. "Development and validation of a strain-based structural health monitoring system." *Mech. Syst. Sig. Process.* 23 (2): 372–383. https://doi.org/10.1016/j.ymssp.2008.03.006.
Krizhevsky, A., I. Sutskever, and G. E. Hinton. 2012. "ImageNet classification with deep convolutional neural networks." In *Advances in neural information processing systems*, 1097–1105. Neural Information Processing Systems.
Laflamme, S., L. Cao, E. Chatzi, and F. Ubertini. 2016. "Damage detection and localization from dense network of strain sensors." *Shock Vib.* 2016: 2562949. https://doi.org/10.1155/2016/2562949.
LeCun, Y., and Y. Bengio. 1995. *Convolutional networks for images, speech, and time-series*. Cambridge, MA: MIT Press.
LeCun, Y., Y. Bengio, and G. Hinton. 2015. "Deep learning." *Nature* 521 (7553): 436–444. https://doi.org/10.1038/nature14539.
LeCun, Y., L. Bottou, Y. Bengio, and P. Haffner. 1998. "Gradient-based learning applied to document recognition." *Proc. IEEE* 86 (11): 2278–2324. https://doi.org/10.1109/5.726791.
Lee, J. J., J. W. Lee, J. H. Yi, C. B. Yun, and H. Y. Jung. 2005. "Neural networks-based damage detection for bridges considering errors in baseline finite element models." *J. Sound Vib.* 280 (3): 555–578. https://doi.org/10.1016/j.jsv.2004.01.003.
Li, L., K. G. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar. 2017. "Hyperband: A novel bandit-based approach to hyperparameter optimization." *J. Mach. Learn. Res.* 18 (1): 6765–6816.
Li, Y. 2010. "Hypersensitivity of strain-based indicators for structural damage identification: A review." *Mech. Syst. Sig. Process.* 24 (3): 653–664. https://doi.org/10.1016/j.ymssp.2009.11.002.
Mehrjoo, M., N. Khaji, H. Moharrami, and A. Bahreininejad. 2008. "Damage detection of truss bridge joints using artificial neural networks." *Expert Syst. Appl.* 35 (3): 1122–1131. https://doi.org/10.1016/j.eswa.2007.08.008.
Nair, K. K., A. S. Kiremidjian, and K. H. Law. 2006. "Time series-based damage detection and localization algorithm with application to the ASCE benchmark structure." *J. Sound Vib.* 291 (1): 349–368. https://doi.org/10.1016/j.jsv.2005.06.016.
Pan, B., K. Qian, H. Xie, and A. Asundi. 2009. "Two-dimensional digital image correlation for in-plane displacement and strain measurement: A review." *Meas. Sci. Technol.* 20 (6): 062001. https://doi.org/10.1088/0957-0233/20/6/062001.
Pei, J.-S., A. Smyth, and E. Kosmatopoulos. 2004. "Analysis and modification of Volterra/Wiener neural networks for the adaptive identification of non-linear hysteretic dynamic systems." *J. Sound Vib.* 275 (3–5): 693–718. https://doi.org/10.1016/j.jsv.2003.06.005.
Robbins, H., and S. Monro. 1951. "A stochastic approximation method." In *The annals of mathematical statistics*, 400–407. New York: Springer.
Rojas, R. 2013. *Neural networks: A systematic introduction*. New York: Springer.
Rumelhart, D. E., G. E. Hinton, and R. J. Williams. 1988. "Learning representations by back-propagating errors." *Cognit. Model.* 5 (3): 1.
Samuel, A. L. 1959. "Some studies in machine learning using the game of checkers." *IBM J. Res. Dev.* 3 (3): 210–229. http://doi.org/10.1147/rd.33.0210.
Sbarufatti, C., A. Manes, and M. Giglio. 2013. "Performance optimization of a diagnostic system based upon a simulated strain field for fatigue damage characterization." *Mech. Syst. Sig. Process.* 40 (2): 667–690. https://doi.org/10.1016/j.ymssp.2013.06.003.
Shahidi, S. G., N. S. Gulgec, and S. N. Pakzad. 2016. "Compressive sensing strategies for multiple damage detection and localization." Vol. 2 of *Dynamics of civil structures*, 17–22. New York: Springer.
Shalev-Shwartz, S., and S. Ben-David. 2014. *Understanding machine learning: From theory to algorithms* Cambridge, UK: Cambridge University Press.
Shi, A., and X.-H. Yu. 2012. "Structural damage detection using artificial neural networks and wavelet transform." In *Proc., IEEE Int. Conf. on Computational Intelligence for Measurement Systems and Applications*, 7–11. New York: IEEE.
Shu, J., Z. Zhang, I. Gonzalez, and R. Karoumi. 2013. "The application of a damage detection method using artificial neural network and train-induced vibrations on a simplified railway bridge model." *Eng. Struct.* 52: 408–421. https://doi.org/10.1016/j.engstruct.2013.02.031.
Simon, P. 2013. *Too big to ignore: The business case for big data*. Vol. 72. New York: Wiley.
Simonyan, K., and A. Zisserman. 2014. "Very deep convolutional networks for large-scale image recognition." Preprint, submitted September 4, 2015. http://arXiv.org/abs/1409.1556.
Sohn, H., and C. R. Farrar. 2001. "Damage diagnosis using time series analysis of vibration signals." *Smart Mater. Struct.* 10 (3): 446–451. https://doi.org/10.1088/0964-1726/10/3/304.
Swamidas, A., and Y. Chen. 1995. "Monitoring crack growth through change of modal parameters." *J. Sound Vib.* 186 (2): 325–343. https://doi.org/10.1006/jsvi.1995.0452.
Szegedy, C., W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. 2015. "Going deeper with convolutions." In *Proc., IEEE Conf. on Computer Vision and Pattern Recognition*, 1–9. New York: IEEE.
Theano Development Team. 2016. "Theano: A Python framework for fast computation of mathematical expressions." Preprint, submitted May 9, 2015. http://arXiv.org/abs/1605.02688.
Yam, L., Y. Li, and W. Wong. 2002. "Sensitivity studies of parameters for damage detection of plate-like structures using static and dynamic approaches." *Eng. Struct.* 24 (11): 1465–1475. https://doi.org/10.1016/S0141-0296(02)00094-9.
Yao, R., and S. N. Pakzad. 2012. "Autoregressive statistical pattern recognition algorithms for damage detection in civil structures." *Mech. Syst. Sig. Process.* 31 (Aug): 355–368. https://doi.org/10.1016/j.ymssp.2012.02.014.
Yao, R., S. N. Pakzad, and P. Venkitasubramaniam. 2016. "Compressive sensing based structural damage detection and localization using theoretical and metaheuristic statistics." *Struct. Control Health Monit.* 24 (4): e1881. https://doi.org/10.1002/stc.1881.
Zapico, J., M. Gonzalez, and K. Worden. 2003. "Damage assessment using neural networks." *Mech. Syst. Sig. Process.* 17 (1): 119–125. https://doi.org/10.1006/mssp.2002.1547.
Zeiler, M. D., and R. Fergus. 2014. "Visualizing and understanding convolutional networks." In *Proc., European Conf. on Computer Vision*, 818–833. New York: Springer.
Zhang, C., S. Bengio, M. Hardt, B. Recht, and O. Vinyals. 2016. "Understanding deep learning requires rethinking generalization." Preprint, submitted November 10, 2015. http://arXiv.org/abs/1611.03530.