Automated classification of civil structures defects based on Convolutional Neural Network

Automated classification of civil structures defects based on Convolutional Neural Network

Pierclaudio SAVINO[a*], Francesco TONDOLO[a]

[a]*Department of Structural, Geotechnical and building Engineering, Politecnico di Torino, Torino 10129, Italy.*
**Corresponding author. E-mail: pierclaudio.savino@polito.it*

ABSTRACT Today, the most used method for civil infrastructure inspection is based on visual assessment performed by certified inspectors following prescribed protocols. However, the increase in aggressive environmental and load conditions, coupled with the achievement for many structures of the end life-cycle, highlighted the need to automate damage identification to satisfy the number of structures that need to be inspected. To overcome this challenge, the current paper presents a method to automate the concrete damage classification using a deep Convolutional Neural Network (CNN). The CNN is designed after an experimental investigation among a wide number of pretrained networks, all applying the transfer learning technique. Training and Validation are performed using a built database with 1352 images balanced between "undamaged", "cracked", and "delaminated" concrete surface. To increase the network robustness compared to images with real-world situations, different configurations of images has been collected from Internet and on-field bridge inspections. The GoogLeNet model is selected as the most suitable network for the concrete damage classification, having the highest validation accuracy of about 94%. The results confirm that the proposed model can correctly classify images from real concrete surface of bridges, tunnel and pavement, resulting an effective alternative to the current visual inspection.

KEYWORDS Concrete structure, infrastructures, visual inspection, convolutional neural network, artificial intelligence

1 Introduction

The management and efficiency of the infrastructural network has a great relevance for the growth of a country, presenting repercussions from both economic and safety point of view. Preserving the structural integrity and reliability of bridges, tunnels and roads is a difficult task to which countries all over the world are called. If on the one hand the construction of reinforced concrete structures made it possible to exploit the static collaboration of two complementary materials such as concrete and steel, on the other hand it has brought out the problems of durability deriving from their union over the years. As a result, structures still in service with aging problems have been inherited and for which priority interventions should be defined to plan the right allocation of resources and avoid catastrophic events. At the same time, in rapidly developing regions characterized by increasing traffic volume, the detection of defects is required from the early stages of the structures to avoid any further losses in structural capacity, durability and therefore maintenance costs.

Today, the first step to identify the damage state and potential risk conditions takes place through direct visual inspections of certified professionals, combined with decision-making process. Early detection and classification of defects helps to plan an effective repair program, preventing worse conditions and enabling low-cost maintenance. Cracks, spalling and

delamination are key indicators for the general conditions of a concrete structural member. Other than to visually showing the load conditions of a structure, cracks are the first mark of surface degradation caused by corrosion of steel. Based on their shape and location it is possible to discover hidden diseases and potential causes. However, because the inspection tasks are done manually, there are some issues that need to be solved. Most of the structure placements are in dangerous locations, where the low accessibility could lead to inaccurate evaluations or sometimes to ignore the defects. In addition, the subjective nature of such assessment is strictly linked to the expertise of the inspector. Consequently, the subjectivity and variability affect the damage detection which results ineffective and not repeatable during time. Furthermore, the condition assessment can be expensive, with both long logistic time and labor intensive.

Developments in modern sensors and ICT techniques can significantly help to replace current defect inspection practices with a more repeatable, reliable and automated civil infrastructure condition assessment. More powerful Graphical Processing Units (GPU) for the parallel computing, the ability to transfer and collect big data via Internet and innovative learning architectures, are changing the way a wide range of industries works by introducing Artificial Intelligence (AI). One of the machine learning technique that has been spreading in recent years thanks to the high performance achieved is deep learning. Based on neural networks architectures, deep learning technique performs classification tasks analysing directly images, text or sounds, giving a relevant impact in fields such as autonomous driving system, medical imaging, face recognition, etc. Furthermore, Artificial neural networks (ANNs) have been developed to approximate the solution of partial differential equations of mechanical problems. In most of the approach, a collocation method is employed to fit both governing equations and boundary conditions at randomly selected points. Anitescu et al. [1] proposed a collocation method for solving 2^{nd} order boundary value problems such as Poisson and Helmholtz equations, based on adaptive approach. Guo et al. [2] developed collocation method for the governing partial differential questions of Kirchhoff plate bending problems. The implementation of computer vision technique in self-navigating robots such as unmanned aerial vehicle, ground robots or consumer electronics, could represent a turning point also for the automating structural inspection process. Images acquired mechanically contain visual information comparable to that obtained by the human inspector. Furthermore, images provide details from the entire field of view quickly, economically and without contact.

Due to the complex texture of the defects, various image backgrounds, changing lighting, surface roughness and finish, a neural network suitable for the automatic classification of such images is the Convolutional Neural Network (CNN). Unlike traditional network architectures, characterized by simple structures and based on the manual extraction of features with image processing technique, CNN learns features directly from the training data, by using 2D convolutional layer. This is an important advantage because CNNs no longer require the images processing, the design and the manual extraction of features. In recent years, numerous studies applied CNN-based algorithms for solving image-based object detection and classification problems. Fan et al. [3] showed a supervised algorithm using CNN to learn the cracks structure on different pavement conditions. Zhang et al. [4] developed four-layer CNN to detect road crack, reaching an accuracy of about 87%. Pre-trained deep network on a large-scale image dataset is often reused with transfer-learning technique, to improve the network relating to accuracy and time consuming. Kim and Cho [5] applied transfer learning to the AlexNet network dividing the training set into cracks, intact surfaces, patterns of joints and edges, and plants. The training of a deep neural network led to an average precision of

about 92%. Hung et al. [6] built a new dataset from Internet with normal, cracked, honeycomb, blistering and moss labels. The transfer learning approach is used obtaining a maximum accuracy of 93%. Zhu and Song [7] developed a model for detecting surface defects on concrete bridges based on transfer learning and CNN. The VGG-16 was modified to classify surface as normal, cracks, plate fracture, corner break, edge exfoliation, skeleton exposure, and repairs with about 83% of validation accuracy. Feng et al. [8] proposed a deep CNN with transfer learning to the damage detection of hydro-junction infrastructure. For the fast tunnel crack identification, Song et al. [9] built a tunnel crack dataset and proposed a deep learning algorithm to define a complete analysis system for identifying tunnel cracks. Makantasis et al. [10] combined CNN and multi-layer perceptron for the tunnel defects inspection problem. Patterson et al. [11] trained a deep learning algorithm to automate post-earthquake reconnaissance image tagging tasks. Gulgec et al. [12] performed finite-element analysis to simulate cracking of gusset plate connections in steel bridges and classify the strain field as "damaged" and "healthy". To provide detection of multiple types of damages, Cha et al. [13] defined a Faster Region-based CNN modifying the ZF-net architecture. A database of 297 images containing steel delamination and corrosion, bolt corrosion and concrete crack is created, reaching a mean precision of about 88%. Soukup and Huber-Mork [14] trained CNNs to detect rail surface defects, using a database of photometric stereo images of rail surfaces in a dark-field configuration. Li et al. [15] improved the YOLO architecture to detect six type of surface defect on cold-rolled steel strip, improving the manufacturing quality of a whole steel strip line.

The above studies have highlighted the potential and versatility of neural networks in the automatic classification tasks, however there are some aspects to overcome to make this procedure reliable. Most of the research has been mainly limited to detect only the surface cracks excluding other types of damages typical of real structures such as delamination. Many times, the datasets are based only on ideal laboratory conditions, containing only images with high resolution and prefixed camera-object distance, excluding real structures in different environmental conditions. The main contribution of the present paper is a deep learning-based algorithm able to classify various type of civil infrastructure surfaces in "undamaged", "cracked" and "delaminated" class. The robustness of the network is ensured considering a database of raw images collected from Internet, field bridge inspections and Google Street View, containing real environmental conditions, background variety and affected by noise. The CNN architecture is developed with the transfer learning approach by in-depth comparison between the most performing deep neural networks in terms of accuracy and prediction time. A further contribution was therefore to identify the best performing network in identifying damages in civil infrastructures, analysing a wide number of the existing pre-trained networks. The neural network developed is able to classify various types of structures, damage surfaces and images quality with about 94% accuracy. The experimental results on the validation dataset highlight a promising performance of the deep learning model to provide a faster and cheaper overview of existing infrastructures than the current visual inspection. The selected network will be the basis for future developments in automatic defect localization and quantification.

2 Convolutional Neural Network

The CNN is a common algorithm of deep neural networks, a technique of machine learning where computer models learn how to perform classification tasks directly from input data. CNNs learn from image data, eliminating the need for manual features extraction. Similar to

the biological structure of a visual cortex in the human brain, a network can have up to hundreds of layers, each aimed at learning and detecting specific features from images. It can initially be very low-level features or kernels, such as brightness and edges, to gradually take on more complex shapes that uniquely define the object. The layers closer to the output interpret the features extracted related to the context of the classification task. The CNN generally contains convolution, activation, pooling and fully connected layer [16]. Filters are applied to each input image, and the output of each layer is the input for the next layer. After learning the features in the surface layers, the network moves to the classification. The last layer before the output one is a fully connected layer which creates a vector of size equal to the number of classes that the network can predict, containing the probabilities for each class (Fig. 1). Since CNN can also be trained on millions of images and can have very complex network architectures, the processing time required to train a model can be significantly reduced by using GPUs. Depending on the size of the dataset, it can be created a CNN from scratch, or re-train existing networks with an own dataset, for new recognition tasks. This procedure, called transfer-learning, is a cost-effective solution for applying the deep learning technique without a large dataset and long processing and training time.

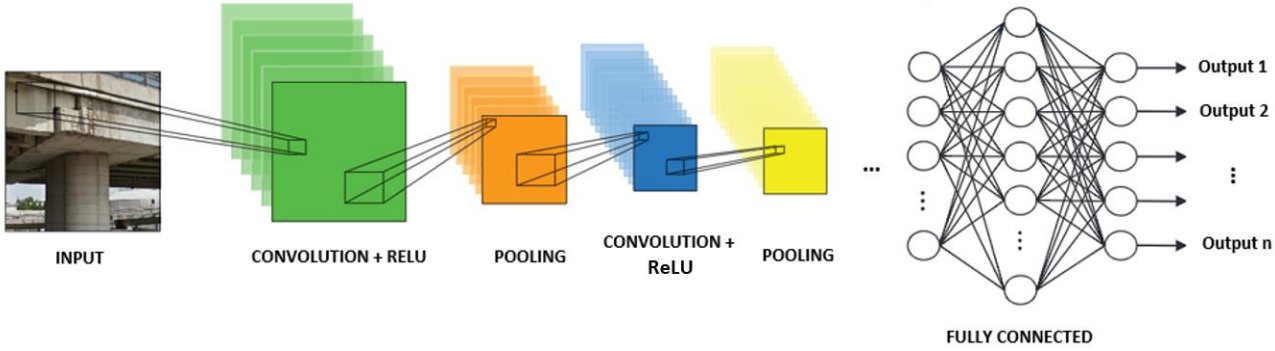

Fig. 1 CNN image classification architecture.

2.1 Convolution layer

The Convolution layers extract and learn features from their input images by means of neurons arranged in feature maps. Each neuron is linked to a series of adjacent neurons in the following layer with a set of trainable weights called convolution kernels. Since within the same convolutional layer there are feature maps with different weights, several features can be extracted at each level [17]. In convolution operations the kernel slides with prefixed stride on each region of the input image, performing a matrix multiplication between the input pixel and the corresponding weights of the convolution kernel. Figure 2 shows an example of convolution operations that involves a 2 x 2 x 1 convolutional kernel of weights 0.5 and stride 1. A lager stride size leads to a smaller output and computational cost but can lose features from the input data.

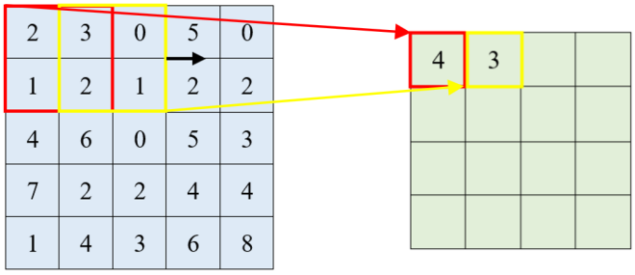

Fig. 2 Convolution operations.

Bias term can be added to the weighted pixels, according to the following equations:

$$Y_i = \sum_i x_i \cdot w_{ij} + b_i \qquad (1)$$

Where x_i denotes the input pixel, w_{ij} and b_i represents the weights and the bias of the convolutional filter and Y_i represents the output. Both parameters are fine-tuned during training process so that the weight increase or decrease the effectiveness of a specific input, bias term make the model more flexible by adjusting the output with the weighted sum of the inputs to the neuron. After the convolutional layer, the convolved results are sent to a non-linear activation function that provides the non-linear behaviour to the neural networks.

2.2 Activation layer

In the neural network architecture, non-linear functions are introduced to extract non-linear features when the neurons are activated. The first types of function to be used traditionally are the sigmoid and hyperbolic tangent function. However, both functions present "vanishing gradient problem" for very large and very small arguments, making impossible for neurons acquire signals or transfer weights and data (Fig. 3). Subsequently, an effective activation function named rectified linear unit (ReLU) [18] is introduced, becoming the most popular:

$$reLU(Y) = max(0, Y) \qquad (2)$$

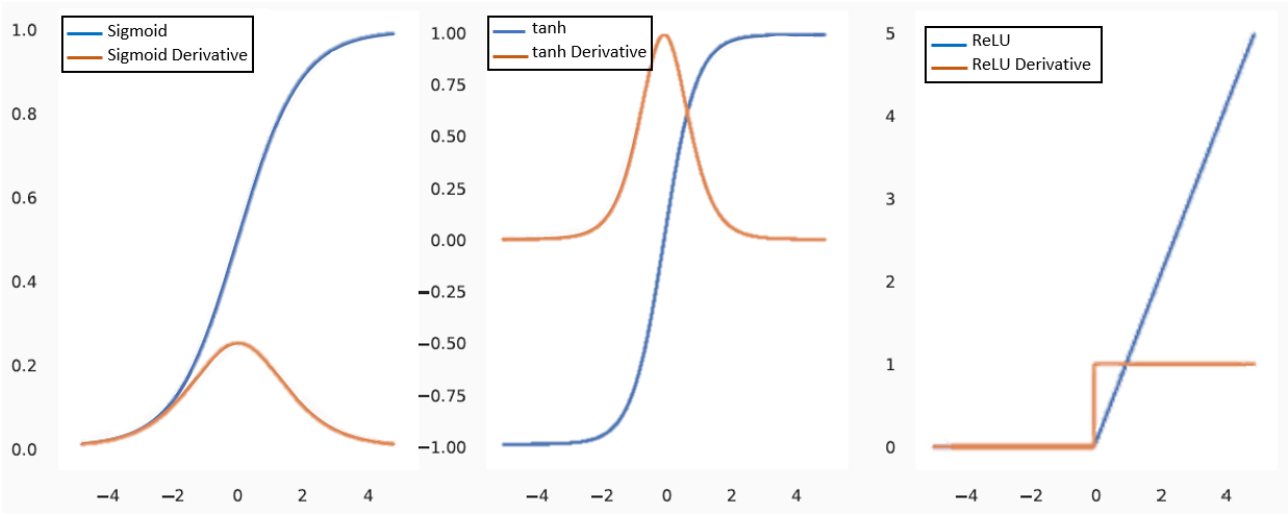

Fig. 3 Activation functions.

Comparing the gradients of the function ReLU with both the sigmoid function and tanh function, the gradients of the ReLU are always zero or one. The learning will be performed in a specific neuron when training samples generate positive input for ReLU. These features facilitate much faster computations and convergence speed.

2.3 Pooling layer

After the convolution process, the output neurons include some redundant information which increases calculations. The Pooling layer improves algorithm performance and decreases the computational cost required to process the data by reducing the size of the convolved feature. Furthermore, it can help to keep model training more robust to object orientation and scale changes by extracting the main features. The key features are extracted by Max Pooling or Average Pooling. The Max Pooling provides the maximum value from the region that the

filter covers on the image, the Average Pooling gives the average of the values contained within the kernel. Figure 4 shows the difference between Average pooling and Max pooling.

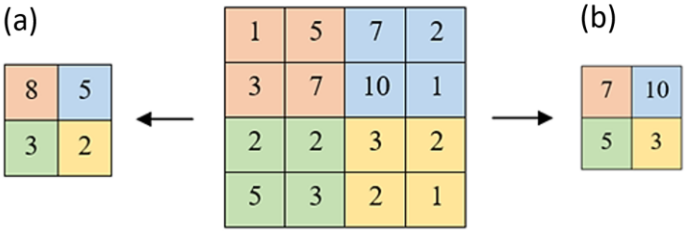

Fig. 4 Pooling operators: (a) Average pooling; (b) Max pooling.

Given an input image of size 4 x 4, a filter of size 2 x 2 and stride 2 is applied. The Convolution layer and the Pooling layer form a single layer of the CNN architecture.

2.4 Fully connected layer

After the convolutional layers it is required to take high-resolution data and solve into representations of objects. To this purpose, the fully connected layer interprets all features extracted by the previous layers to produce the class labels. It can be thought as a link between classifier and information output. As the name suggests, each neuron is connected to every neuron in the previous layer as the traditional neural networks. In such a way the features extracted and learned in the shallow layers can be mapped into the tag array. In order to attach the fully connected layer to the network, the size of the CNN output needs to be flattened into a vector of values. To obtain the classes of the input images, the Softmax layer is finally located at the bottom of the CNN architecture. The output unit activation function for multi-class classification problem is given by the Softmax function:

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \qquad (3)$$

where $0 \leq \sigma(z)_j \leq 1$ and $\sum_{j=1}^{K} \sigma(z)_j = 1$. The output of the Softmax layer represents the probability that the networks associates the i^{th} input with class j^{th}.

2.5 Softmax loss and stochastic gradient descent

Once designed the neural network, the free parameters need to be optimized with learning algorithms to obtain the correct network output. The most common algorithm used to train a neural network on the training dataset is backpropagation. Backpropagation allows to update parameters by moving forward and backward across the network, until the loss function between predicted and actual outputs reaches its local minimum. The deviations between predicted and actual classes are defined by:

$$loss = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{K} t_{ij} \, ln \, \sigma(z)_j \qquad (4)$$

where N is the samples number, K is the classes number, t_{ij} is a logical term that return one if i^{th} sample following the j^{th} class, and $\sigma(z)_j$ is the output for the sample i and class j. The most efficient way to minimize the deviations and update the weights is stochastic gradient descent (SGD) using backpropagation. Starting by initial values of weight and loss, the algorithm provides the best weights by computing the global loss minimum according to the equation:

$$w = w - \alpha \frac{\partial loss}{\partial w} \qquad (5)$$

where α indicates the learning rate and *w* is the weight. When the gradient is positive the weight is reduced, otherwise when it is negative the weight is increased. How much this contribute should be considered is defined by the learning rate. During the CNN training, the above process is repeated many times on mini-batch of images until an epoch of iterations over the entire dataset is completed. After each iteration, the gradient of the loss and the updating of the weights are performed.

3 Deep CNN with Transfer Learning

Humans have the inherent ability to acquire knowledge while learning an activity and use it alike to solve related tasks. Similarly, the transfer learning is a widespread deep learning application employed for pretrained network as the starting point of new classification scenario. Fine-tuning a network using the transfer learning is typically much faster than training a network from scratch and provides good accuracy even with fewer number of training images. Most pretrained networks have been developed for the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), the reference benchmark for large scale object recognition [19]. These networks have been trained on over a million images contained in the ImageNet database and can classify and detect 1000 object categories. The great variety and size of the training dataset guarantees a better ability to extract features from various kinds of images. Pretrained networks have different accuracy, speed and size that should be taken into account when looking for solution of a specific problem. In the present study, eight of the fastest pretrained CNN models have been experimented to perform the transfer-learning on concrete damages classification task. From the experimental study aimed to evaluate the influence of training parameters, the most performing neural network has been selected in terms of speed and accuracy.

3.1 Pre-trained neural networks

Deep learning models have layered architectures in which different each layer correspond to different feature learning. As mentioned above, the last layer is the fully connected layer to get the final output. This architecture allows to use the weights in feature extractor layers as the starting point for the training process and adapt them in response to the new problem. Furthermore, it allows to replace the last fully connected layer, Softmax layer and classification output layer for the fine-tuning on the different classification problem. In the present study, eight different pretrained networks have been modified, including AlexNet, SqueezeNet, ShuffleNet, ResNet-18, GoogLeNet, ResNet-50, MobileNet-v2 and NASNet-mobile.

AlexNet is a convolutional neural network trained on over a million images in the ImageNet LSVRC-2010 contest and can classify images belonging to 1000 different categories [20]. AlexNet won the ILSVRC-2012 contest with a top-5 error rate of 15.3%, compared to that of the second classified of 26.2%. The network contains five convolutional layers, some of which connected to Max pooling layers, and three fully connected layers with a Softmax layer and classification output layer at the end. To transfer the layers for new classification task, the last three layers need to be fine-tuned in order to have the fully connected layer with the size equal to the number of classes in the new data.

SqueezeNet is a smaller CNN that achieves the same accuracy as AlexNet on ImageNet, with 50 times fewer parameters and 510 times smaller than AlexNet [21]. The SqueezeNet

architecture is comprised of 1x1squeeze convolutional layers and expand layers with a mix of 1x1 and 3x3 convolution filters. Unlike of most networks, in which the last layer with learnable weights is a fully connected layer, in SqueezeNet the last learnable layer is the final convolutional layer. To retrain a pretrained network to classify new images, the convolutional layer should be replaced with a new convolutional layer having the same number of classes as the number of filters.

ShuffleNet is a very computation-efficient CNN designed for device with limited computing power [22]. The new architecture introduces pointwise group convolution and channel shuffle to reduce computational costs while maintaining accuracy. ShuffleNet achieves about 13 times less of actual speedup compared to AlexNet while maintaining comparable accuracy on ImageNet dataset.

ResNet-18 and ResNet-50 are a residual CNNs with an architecture 18 and 50 layers deep respectively [23]. Their introduction was the most revolutionary work in the deep network architecture. Thanks to the increasing network depth by simply stacking the layers can lead to vanishing gradient problem or to overfitting the data, the core idea of the Authors was the introduction of the so-called "identity shortcut connection" to bypass one or more layers. Residual connections allow to propagate more easily the parameter gradients from the output layer to previous network layers, making it possible to train deeper networks with greater accuracies.

GoogLeNet is the winner of the ILSVRC-2014 competition, and as the name suggests, it was developed by a Google team [24]. To reduce the use of the computing resources while increasing the depth and width of the network, a new architecture named "Inception" was proposed. The idea of the Inception layer was to have filters from the most accurate detailing (1x1) to the bigger one (5x5) that can work in parallel on the same level. To avoid a parameter explosion on the Inception layers and perform faster computations, a 1x1 convolution is added as a dimension reduction module. Using the bottleneck approach, depth and width can be increased without computational and overfitting problems. Instead of using fully connected layer, the ends of the inception modules are connected to the global average pooling layer.

MobileNet-v2 [25] is a refinement of v1 [26] mobile architecture, which in order to decrease the computational cost, introduced the idea to replace the convolutional layers with so-called depthwise separable convolutions. The convolution layer is split into a depthwise convolution layer to filter the input and a 1x1 pointwise convolution layer to combine the filtered values and create new features. The new architecture presents an inverted residual structure where the input and output residual block are opposite to traditional residual models. This kind of layer based on the bottleneck principle reduces the amount of data flowing through the network with a relevant computational gain. Furthermore, to help in the gradient propagation, the residual connections are introduced like in ResNet network.

NASNet-mobile was proposed as an algorithm to learn the network architectures directly on the dataset considered [27]. Authors searched for the best convolutional layer on the smallest CIFAR-10 dataset and applied it to the ImageNet dataset by stacking together multiple times. The NASNet model allows accurate results to be achieved with smaller model sizes and less complexity.

The following table lists the main properties of the pretrained networks analysed.

Table 1 Pretrained models properties

network	number of layers	size (Mb)	parameters (millions)
AlexNet	25	227	61
SqueezeNet	68	4.6	1.24
ShuffleNet	173	6.3	1.4
ResNet-18	72	44	11.7
GoogLeNet	144	27	7
ResNet-50	177	96	25.6
MobileNet-v2	155	13	3.5
NASNet-mobile	914	20	5.3

4 Experimental results and analysis

The classification accuracy of a deep learning model is strongly influenced by the size and quality of the image dataset. However, most common datasets contain a limited number of images often showing only ideal laboratory conditions with no real structure surfaces and environmental conditions as background. The proposed experimental program implementing a neural network for the automatic classification of concrete damages, is based on the construction of reference image dataset, on the analysis and detection of reference artificial intelligent system using the transfer learning approach and finally on the optimization and validation of the defined neural network. Before defining the reference neural network, the best set of hyperparameters is identified for each model. The training is performed in MATLAB 2019b environment on a workstation with the following configurations: Intel® Core™ i7-3537U CPU@ 2.00GHz - 2.50 GHz, 8.00 GB of RAM and a NVIDIA GeForce GT720M GPU.

4.1 Image dataset

The transfer learning is performed using data collected from the Internet, on field bridge inspections and Google Street View. This made it possible to obtain the selection of a greater and different variety of civil infrastructures images, increasing the practicality of research in the real field by also simulating the acquisition from robotic platforms or drone. To match the input size of the networks, all the images size has been adjusted by rescaling operation. The model trained on small images learns fewer features than one trained on large images, which are the most important ones to achieve a proper efficiency in the classification task. The robustness of the networks, with the aim to be invariant to distortions in image data, has been ensured by applying randomized augmentation such as rotation, reflection and translation. Furthermore, using an augmented image datastore to transform training data for each epoch increases the amount of training data and prevents the network from overfitting and learning the specific details on training images. Figure 5 shows examples of datastore images classified as “undamaged”, “cracked” and “delaminated”.

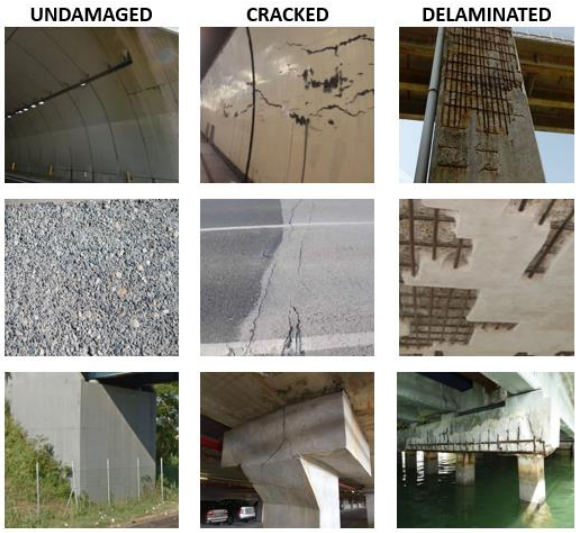

Fig. 5 Image dataset examples.

The original dataset includes a total of 1352 images. To evaluate the network performance during training and detect if the network overfits the details about the training data, the dataset is split into 80% in training set and 20% in validation set. The validation set contains images that the network has never seen before the validation process. When the training loss decreases and the validation loss increases, the training should be stopped because the network learns details about the training set that are not relevant to the new images. The dataset details for each class are summarized in Table 2.

Table 2 Dataset details

class	number of images	training set	validation set
Undamaged	443	355	88
Cracked	441	353	88
Delaminated	468	374	94

To avoid unbalanced classes, the same order of magnitude is considered for the quantity of images.

4.2 Neural networks training

The implementation of deep neural networks is performed with the Deep Learning Toolbox™ in MATLAB. Images are automatically labelled according to folder names and stored using the "imageDatastore" function. By installing the proper Deep Learning Toolbox™ of each model, the architecture can be visualized and the network layers modified, based on the layers replacements required by the transfer learning technique. To make the images compatible with the input size of the network and perform randomized preprocessing operations, the "augmentedImageDatastore" is used. Once the training options are specified, the network is trained employing the "trainNetwork" function, specifying the augmented image datastore, layers and options.

After creating the dataset and modified the networks architecture for the transfer learning, the best training algorithm settings, i.e. learning rate, mini-batch size and epochs number, should be found in order to increase the networks behaviour. Other parameters, such as the dropout rate, have not been taken into account in the optimization process, as they are not present in all networks and therefore are considered with the original value. Both classification accuracy

and training time are taken into account as the metric for the evaluation of such hyperparameters. Since their best value cannot be estimated from data, the optimization process is performed through the iteration on various permutation. By the monitoring of the training progress, their influence on the training dynamics and stability, can help in the detection of the best combination. Using mini-batch of training dataset to minimize error and update weights, guarantees faster convergence and better generalization being an approximation of the entire dataset. Smaller learning rates require a long training time given the minor changes made to the weights update, whereas larger training rates require fewer training epochs but could lead to a suboptimal final set of weights or divergence. Given the above considerations, a good combination is found with a learning rate of 0.001 and a mini-batch size of 32. Lastly, a maximum number of 12 epochs it has been selected as the appropriate number to reach asymptotic behaviour. The classifiers are trained with 396 iterations and validate every epoch. For each validation, the accuracy is evaluated as the ratio between the number of correctly classified images and the total number of images in the validation set.

The plot below (Fig. 6) shows the comparison among the different networks, between the validation accuracy and the time required to train the networks after 12 epochs; the magnitude of the marker representing the networks is related to their byte dimension.

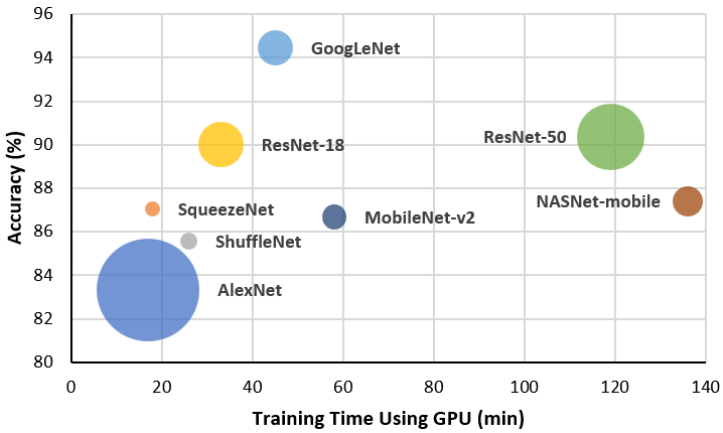

Fig. 6 Comparison of pretrained network.

A wide variety of CNN architectures are obtained with the training experiment, from the fastest and lowest precision to the most complex and accurate. The pretrained network more related to automate the damage assessment results GoogLeNet, with validation accuracy of 94.44% and training time of about 45 minutes. Furthermore, due to its relatively small size on disk and in memory, it is compatible also to perform predictions using mobile sensors with low computational resources or for its distribution over the Internet.

4.3 Neural network details and performance

The whole GoogLeNet architecture is 22 layers deep, with 3 convolution layers, 9 inception modules stacked linearly and deep 2 layers, and the last fully connected layer. All the convolutions inside the network uses ReLU as activation function. The input layer takes images of size 224 x 224 with RGB colour channels. The detailed GoogLeNet architecture is explained in Table 3.

Table 3 GoogLeNet pretrained model architecture

type	filter size / stride	output size	total learnables	depth
Convolution	7x7 / 2	112x112x64	9472	1
Max pool	3x3 / 2	56x56x64		0
Convolution	3x3 / 1	56x56x192	114944	2
Max pool	3x3 / 2	28x28x192		0
Inception (3a)		28x28x256	163696	2
Inception (3b)		28x28x480	388736	2
Max pool	3x3 / 2	14x14x480		0
Inception (4a)		14x14x512	376176	2
Inception (4b)		14x14x512	449160	2
Inception (4c)		14x14x512	510104	2
Inception (4d)		14x14x528	605376	2
Inception (4e)		14x14x832	868352	2
Max pool	3x3 / 2	7x7x832		0
Inception (5a)		7x7x832	1043456	2
Inception (5b)		7x7x1024	1444080	2
Avg pool	7x7 / 1	1x1x1024		0
Dropout		1x1x1024		0
Linear		1x1x1000	1025000	1
Softmax		1x1x1000		0

To retrain the pretrained network to classify “undamaged”, “cracked” and “delaminated” images, the last fully connected layer is replaced with a new fully connected layer containing 3 outputs. A new classification layer has been replaced, specifying the new class labels. Using the set of hyperparameters specified in the previous paragraph, the following accuracy and loss progress is obtained for each iteration, after a training time on single GPU of about 45 min (Fig. 7). A first aspect that can be observed is the quick drop in the loss function that motivate the optimal learning rate set. The slope of the accuracy trend highlights the effectiveness of the transfer learning to achieve high accuracy in a short time.

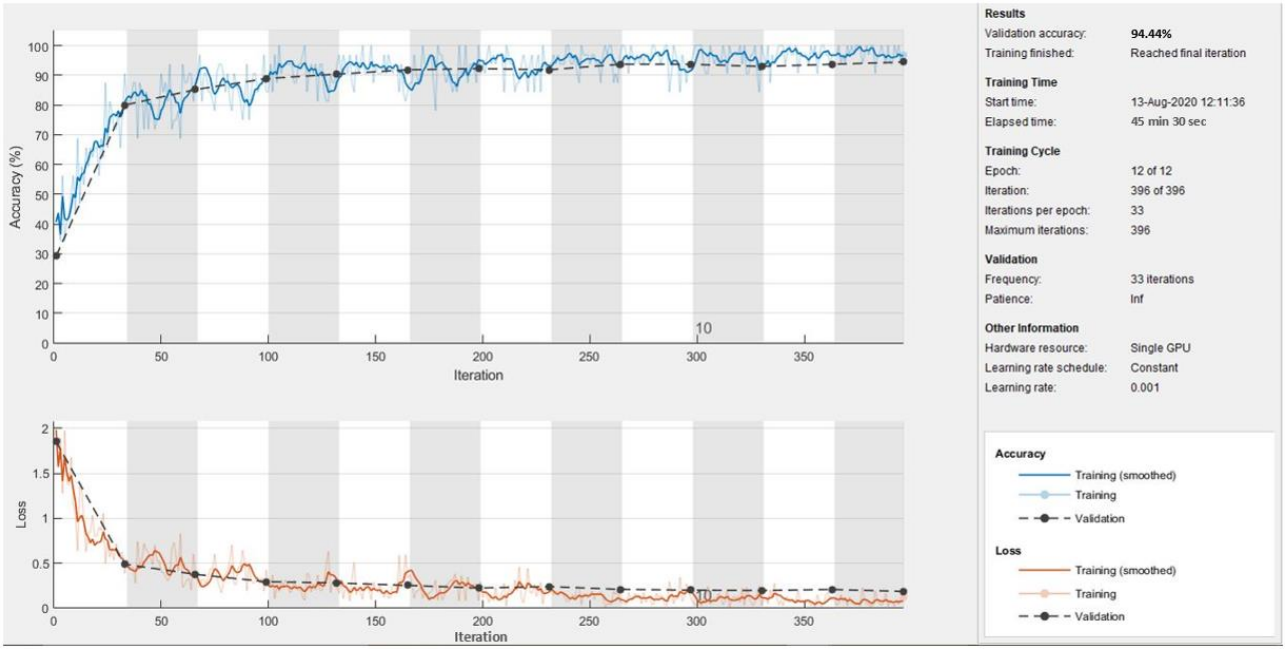

Fig. 7 Training progress.

Furthermore, the training progress shows the same error trends for both validation and training set. This means that the model behaves correctly on both training data and validation data showing general validity. For training dataset without data augmentation, the model achieves 100% accuracy after 150 iterations, leading to over-specialization on the noise and details of the training dataset. As a result, model performance decreases and the validation accuracy drops to 88.89%.
Once the network is trained, the confusion matrix is calculated between the true labels and the predicted labels from the validation dataset. Putting the true classes in the rows and the predicted classes in the column, the diagonal and off-diagonal regions correspond to correct and incorrect observations, respectively (Fig. 8).

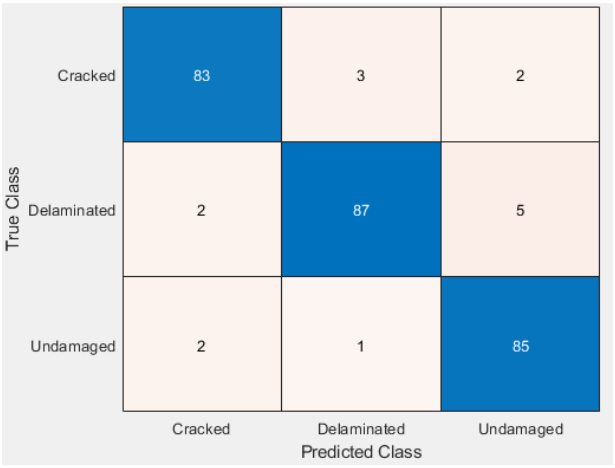

Fig. 8 Confusion matrix chart.

To better understand the performance measurement, the Receiver Operating Characteristics (ROC) curve has been plotted (Fig. 9). The ROC curve shows how the true positive and false positive rates relate, applying decision threshold values across the interval from 0 to 1. For each threshold, the True Positive Ratio (TPR), also named Sensitivity, and the False Positive Ratio (FPR) are calculated by:

$$TPR = \frac{TP}{TP + FN} \qquad (6)$$

$$FPR = \frac{FP}{FP + TN} \qquad (7)$$

where TP is the number of positive instances correctly classified, FN is the number of positive instances misclassified, FP is the number of negative instances incorrectly classified and TN is the number of negative instances correctly classified. Consequently, the proportion of negative instances correctly classified upon the total number of negative instances, can be defined as complementary metrics:

$$Specificity = 1 - FPR \qquad (8)$$

When the threshold decreases, more positive values are obtained thus increasing the Sensitivity and decreasing the Specificity. Similarly, when the threshold increase, more negative values are obtained thus increasing the Specificity and decreasing Sensitivity.

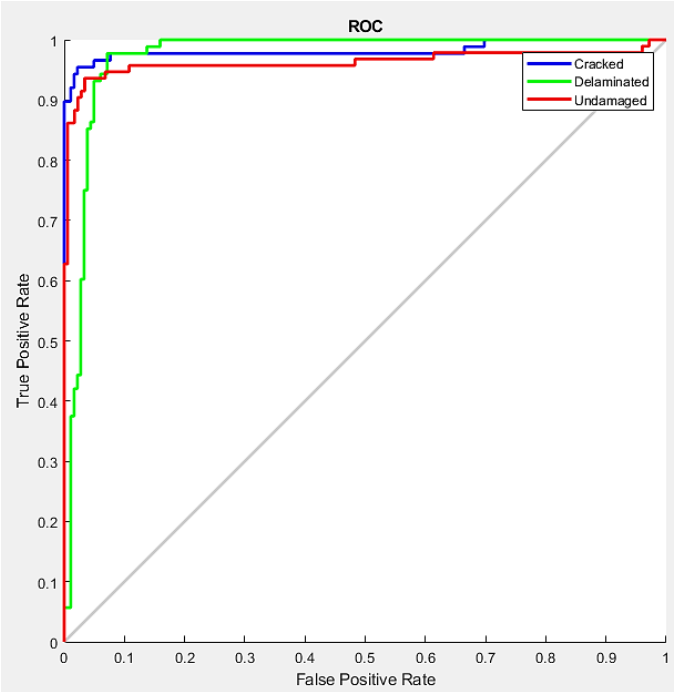

Fig. 9 Receiver Operating Characteristics.

Beside to detect the most appropriate trade-off between Sensitivity, Specificity and threshold, the ROC curve can be effectively employed to study the discriminative ability of the model. Classifier with curves close to the upper left corner, shows a good measure of separability. On the other hand, an Area Under the Curve (AUC) equal to 0 means that model reciprocating the classes and predicts negative class as a positive and vice versa. When AUC is 0.5, the model has no class discrimination capability and corresponds to a random model. The good performance of the model to distinguish between classes is confirmed by the AUC measurements being about 0.98, 0.97 and 0.96 for the “cracked”, “delaminated” and “undamaged” class respectively.

The network correctly predicted 255 images in the validation set containing a total of 270 images, achieving a global accuracy of 94.44%. More in detail, the “cracked” class presents an accuracy of 94.31%, the “delaminated” class has an accuracy of 92.56% and the “undamaged” class has the maximum accuracy of 96.59%. Figure 10 shows some validation results for correct and incorrect classifications, respectively.

Fig. 10 Validation results: (a) correct classification; (b) misclassification.

From the image above, some relevant considerations can be highlighted. First, the image correctly classified as "Delaminated" that contains the street art picture on the pier, confirms the robustness of the model compared to the noisy images. About the misclassification aspect, further observations can be made. Generally, the probability of belonging to the predicted class is low or in any case less than about 90%. Relating to the image misclassified as "Undamaged" with a probability of 97,9%, a possible explanation is the problem of distance from the delaminated area on the bottom deck between the two piers which makes the prediction hard even for human inspection. The further image incorrectly classified with a high probability of 99.9%, contains some relevant details proper of delaminated surface, such as the oxide colour, the texture of the aggregates, wide and deep cracks, that misleads the network. On the other hand, the images classified as "cracked" but containing delaminated elements, demonstrate the need to improve the automatic classification of damages in case of multiple presence of different classes.
In conclusion, GoogLeNet has shown the best performance to solve the problem to automate the inspection of the civil infrastructures. The high accuracy of this model, compared to the quality of the images, very close to real conditions, confirm its suitability to be a good basis on which to develop future studies for a fully-automated inspection.

5 Conclusions and perspective

This paper proposed a new deep neural network for the automatic classification of the main defect in civil infrastructures. The study employed an existing CNN modifying the last fully connected layer building a robust convolutional neural network able to classify images with noisy background, containing vary different defect configuration for bridge, tunnel and pavement structures. To define the most suitable network in concrete damage classification, an experimental investigation is performed among 8 of pretrained network. The major contribution has been to train the network with images that simulate the real collection on field by means of robotic platform, low-cost system or drone. Nevertheless, an accuracy of 94.44% is achieved, highlighting the suitability to be integrate in smart management system

for the automatic inspection and assessment of civil infrastructures. Future developments will concern the improvement of network learning not only to identify the defect classes but also their detection and quantification.

References

1. Anitescu C, Atroshchenko E, Alajlan N, Rabczuk T. Artificial Neural Network Methods for the Solution of Second Order Boundary Value Problems. Computers, Materials and Continua, 2019, 59(1): 345-359

2. Guo H, Zhuang X, Rabczuk T. A Deep Collocation Method for the Bending Analysis of Kirchhoff Plate. Computers, Materials and Continua, 59(2): 433-456

3. Fan Z, Wu Y, Lu J, Li W. Automatic pavement crack detection based on structured prediction with the convolutional neural network. arXiv preprint arXiv: 1802.02208, 2018

4. Zhang L, Yang F, Zhang Y D, Zhu Y J. Road crack detection using deep convolutional neural network. In: IEEE International Conference on Image Processing (ICIP). Phoenix: AZ, 2016, 3708-3712. doi: 10.1109/ICIP.2016.7533052

5. Kim B, Cho S. Automated vision-based detection of cracks on concrete surfaces using a deep learning technique. Sensors, 2018, 18(10): 3452-3469

6. Hung P, Su N, Diep V. Surface Classification of Damaged Concrete Using Deep Convolutional Neural Network. Pattern Recognition and Image Analysis, 2019, 29: 676-687. doi: 10.1134/S1054661819040047

7. Zhu J, Song J. An Intelligent Classification Model for Surface Defects on Cement Concrete Bridges. Applied Sciences, 2020, 10(3): 972 - 990. doi:10.3390/app10030972

8. Feng C, Zhang H, Wang S, Li Y, Wang H, Yan F. Structural Damage Detection using Deep Convolutional Neural Network and Transfer Learning. KSCE Journal of Civil Engineering, 2019, 23(10): 4493-4502. doi: 10.1007/s12205-019-0437-z

9. Song Q, Wu Y, Xin X, Yang L, Yang M, Chen H, Liu C, Hu M, Xuesong C, Li J. Real-time Tunnel Crack Analysis System via Deep Learning. IEEE Access, 2019, 7: 64186-64197. doi: 10.1109/ACCESS.2019.2916330

10. Loupos K, Makantasis K, Protopapadakis E, Doulamis A, Doulamis Nikolaos. Deep Convolutional Neural Networks for efficient vision based tunnel inspection. In: IEEE International Conference on Intelligent Computer Communication and Processing (ICCP). Cluj-Napoca, 2015, 335-342. doi: 10.1109/ICCP.2015.7312681

11. Patterson B, Leone G, Pantoja M, Behrouzi A. Deep Learning for Automated Image Classification of Seismic Damage to Built Infrastructure. In: Proceedings of the 11th National Conference in Earthquake Engineering. Los Angeles: Earthquake Engineering Research Institute, 2018

12. Gulgec N S, Takac M, Pakzad S N. Structural damage detection using convolutional neural networks. In: Barthorpe R, Platz R, Lopez I, et al, eds. Model Validation and Uncertainty Quantification. Conference Proceedings of the Society for Experimental Mechanics Series, Vol 3. Cham: Springer, 2017, 331-337

13. Cha Y J, Choi W, Suh G, Mahmoudkhani S, Büyüköztürk O. Autonomous structural visual inspection using region-based deep learning for detecting multiple damage types. Computer-Aided Civil and Infrastructure Engineering, 2018, 33: 731–747

14. Soukup D, Huber-Mörk R. Convolutional neural networks for steel surface defect detection from photometric stereo images. In: Bebis G, et al, eds. Advances in Visual Computing. ISVC 2014. Lecture Notes in Computer Science, Vol 8887. Cham: Springer, 2014, 668–677

15. Li J, Su Z, Geng J, Yin Y. Real-time Detection of Steel Strip Surface Defects Based on Improved YOLO Detection Network. IFAC-PapersOnLine, 2018, 51: 76-81

16. Rawat W, Wang Z. Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review. Neural Computation, 2017, 29: 1-98. doi: 10.1162/NECO_a_00990

17. LeCun Y, Bengio Y, Hinton G. Deep learning. Nature, 2015, 521(7553): 436‑444

18. Nair V, Hinton G E. Rectified linear units improve restricted Boltzmann machines. In: Proceedings of the 27th International Conference on Machine Learning (ICML-10). Haifa, 2010, 807–14

19. Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M, Berg A C, Fei F L. ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV), 2015, 115(3): 211–252

20. Krizhevsky A, Sutskever I, Hinton G E. ImageNet Classification with Deep Convolutional Neural Networks. Communication of the ACM, 2017, 60(6): 84-90

21. Iandola F N, Han S, Moskewicz M W, Ashraf K, Dally W J, Keutzer K. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5 MB model size. https://arxiv.org/abs/1602.07360

22. Zhang X, Zhou X, Lin M, Sun J. ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City, 2018, 6848-6856. doi: 10.1109/CVPR.2018.00716

23. He K, Xiangyu Z, Shaoqing R, Jian S. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, 770-778

24. Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A. Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, 1-9

25. Sandler M, Howard A, Zhu M, Zhmoginov A, Chen L C. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, 4510-4520

26. Howard A G, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, Andreetto M, Adam H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv:1704.04861, 2017

27. Zoph B, Vasudevan V, Shlens J, Le Q V. Learning Transferable Architectures for Scalable Image Recognition. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City, 2018, 8697-8710. doi: 10.1109/CVPR.2018.00907