Incorporating Inductive Bias into Deep Learning: A Perspective from Automated Visual Inspection in Aircraft Maintenance

Conference Paper · October 2018

CITATIONS
0

READS
23

5 authors, including:

Vincentius Ewald
Delft University of Technology
5 PUBLICATIONS **5** CITATIONS
SEE PROFILE

Xavier Goby
Delft University of Technology
1 PUBLICATION **0** CITATIONS
SEE PROFILE

Roger M Groves
Delft University of Technology
164 PUBLICATIONS **630** CITATIONS
SEE PROFILE

Incorporating Inductive Bias into Deep Learning: A Perspective from Automated Visual Inspection in Aircraft Maintenance

Vincentius EWALD [1], Xavier GOBY [1], Hidde JANSEN [1], Roger M. GROVES [1], Rinze BENEDICTUS [2]
[1] Aerospace Non-Destructive Testing Laboratory, Faculty of Aerospace Engineering, Delft University of Technology, Delft, Netherlands
[2] Structural Integrity and Composites Group, Delft University of Technology, Faculty of Aerospace Engineering, Delft, Netherlands

Contact e-mail: V.Ewald@tudelft.nl

Abstract. Narrow artificial intelligence, commonly referred as 'weak AI' in the last couple years, has developed due to advances in machine learning (ML), particularly deep learning, which has currently the best in-class performance among other machine learning algorithms. In the deep learning framework, many natural tasks such as object, image, and speech recognition that were impossible in the previous decades using classical ML algorithms can now be done by a typical home personal computer.

Deep learning requires a rapid collection of a large amount of data (also known as 'big data') to create robust model parameters that are able to predict future occurrences of certain event. In some domains, large datasets such as the CIFAR-10 image database and the MNIST handwriting database already exist. However, in many other domains such as aircraft visual inspection, such a large dataset of damage events is not available, and this is a challenge in training deep learning algorithms to perform well to recognize material damage in aircraft structures.

As many computer science researchers believe, we also think that in order to achieve a performance similar to human-level intelligence, AI should not start from scratch. Introducing an inductive bias into deep learning is one way to achieve this human-level intelligence in the aircraft inspection for damage. In this paper, we give an example of how to incorporate domain knowledge from aerospace engineering into the development of deep learning algorithms. We demonstrate the suitability of our approach using data from fatigue testing of an aerospace grade aluminum specimen to build a deep convolutional neural network that classifies crack length according to the crack propagation curve obtained from fatigue test. The results of this network were then compared to the same network that was not trained with domain knowledge and the biased learning achieved a validation accuracy of 97.55% on determining crack length, while unbiased network selected the unwanted parameter of sunlight intensity, however with 99.45% accuracy.

Keyword: *deep learning, convolutional neural network, image recognition, inductive bias, crack propagation, visual inspection, non-destructive testing*

1. Introduction

1.1. State-of-the art

As of 2018, the inspection of aircraft is primarily done manually with 80% of inspection being carried out visually [1]. This suggests that damage detection via image recognition systems has potential to assist in aircraft inspection. Ever since the thawing of the last, so called "AI-Winter", which took place during the early 90's, the field of artificial intelligence (AI) has experienced a dramatic reinvigoration. Since then, research and development in this field have been exponentially growing, particularly in the field of deep-learning, a broad family of machine learning algorithms within the field of AI. One of the reasons why deep learning is become increasingly more popular is that it eliminates hand-picked features, a time-consuming process in classical machine learning.

Among the many deep learning architectures, convolutional neural network (CNN) is one of the most popular architecture for image recognition [2]. In previous work, we have applied 1-hidden layer CNN for phase discontinuity predictions in NDT [3], however this was not with a deep architecture. To our knowledge, the very first deep CNN application for visual inspection was performed by Zhang et al. for road crack detection on concrete structure by using 6 layers ConvNet [4].

Since then, similar works using CNN for crack detection have been performed in civil infrastructure such as building, [5], pavement [6], and concrete surface [7 – 9]. While these results are promising and show the first step in advancing image recognition for crack detection by CNN, a generalization of the incorporation of domain knowledge is required to have a reasonable justification of the interpretable outcome of deep learning.

1.2. Objective

Taking into account the state-of-the art provided in [3 – 9], the objective of our paper is to show how to incorporate inductive bias in deep learning by injecting bias from aircraft maintenance. Precisely, we built and tested a deep CNN that classifies crack length based on crack propagation curves obtained from fatigue test. Thus, unlike image segmentation and thresholding, the network is built on the physical basis of crack propagation behavior.

In order to test the performance of injecting domain knowledge, the result of crack image classification of the CNN architecture with domain knowledge was compared to the result of image classification of the same network that was not injected by domain bias, but rather pre-clustered by unsupervised machine-learning.

Our paper is structured as follow: Section 2 describes the theoretical background of deep learning, particularly the deep CNN and the formalization of inductive bias. In section 3, we briefly describe the fatigue testing procedure and the machine learning algorithm we used. The results and discussion are presented in section 4, while our summary and conclusion are presented in section 5.

2. Machine and Deep Learning

2.1. Deep Convolutional Neural Network

A convolutional neural network is a class of feedforward artificial neural networks that has been proven to be an extremely effective tool in the field of computer vision. LeCun et al. (1998) [10] developed the first deep CNN back in the late 90's which was called the LeNet-5 architecture and consisted of 7 layers including the output layer as depicted in Fig. 1.

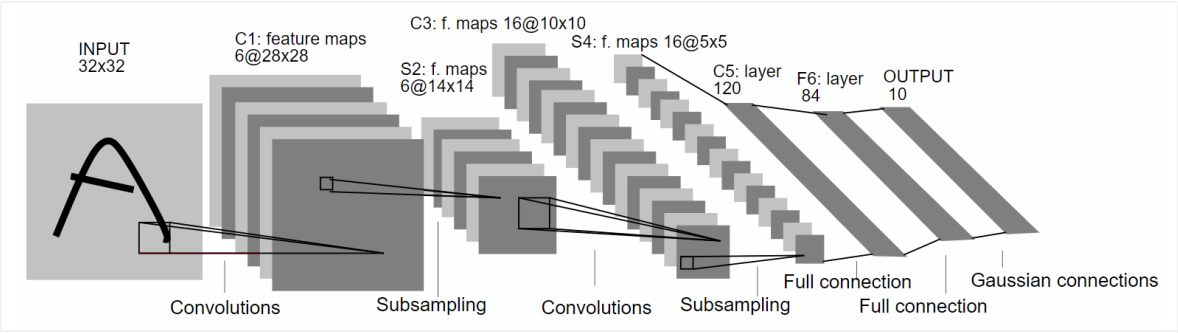

Fig. 1: LeNet-5 [10]

A CNN consists of multiple successively increasingly refined data filters – the layers of a CNN. Within the context of this analogy, the input data to a CNN, for instance a single black and white image can be represented as 3D tensor of size: height ×width × channels of pixel values ranging from 0 to 255. This tensor is multiplied by activation functions, and as it passes through each layer of the network, is transformed into more abstract representations for the networks to produce a prediction output. The transformations performed by each layer are parametrized by its neural weights. Then, the difference between the predicted output and the true output is computed. This difference is called the loss function and the central task of machine learning is to minimize the loss by adjusting the weights through an optimizer during the backpropagation.

There are two types of layers that are normally used in CNN core architecture: the convolutional layer and the pooling layer. The core CNN is typically attached to the fully-connected layer. The convolutional layer is made of a filter (or kernel) which performs a convolutional operation to the input tensor that it receives from previous layer for feature extraction. The filter performs the process of feature extraction by sliding across the input data by several pixels, also called the stride, as depicted in Fig. 2.

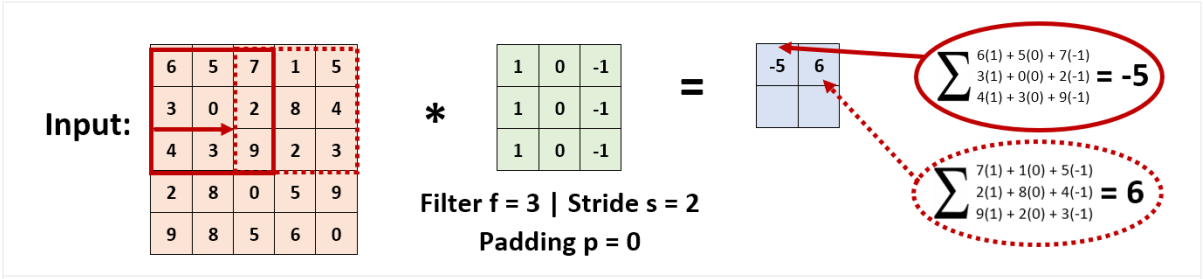

Fig. 2: Convolution operation in CNN using stride s = 2 and zero-padding

2.2. Inductive Bias

The inductive bias of a learning algorithm is the set of assumptions that the learning algorithm $\boldsymbol{A}$ uses to predict outputs for a given input that it has not yet encountered. Formally defined, the inductive bias of learning algorithm $\boldsymbol{A}$ is "*any minimal set of assertions* $\boldsymbol{B}$ *such that for any target concept* $\boldsymbol{c}$ *and corresponding training data* $\boldsymbol{D}$" [11] that following formula holds:

$$\forall \boldsymbol{x_i} \in \boldsymbol{X} : (\boldsymbol{B} \wedge \boldsymbol{D_c} \wedge \boldsymbol{x_i}) \vdash \boldsymbol{A}(\boldsymbol{x_i}, \boldsymbol{D_c}) \quad (1)$$

Where c the target concept, $\boldsymbol{D_c}$ set of training examples, $\boldsymbol{x_i}$ is $\boldsymbol{i}$-th instance of input set $\boldsymbol{X}$, and $\boldsymbol{A}(\boldsymbol{x_i},\boldsymbol{D_c})$ the classification assigned to $\boldsymbol{x_i}$ by $\boldsymbol{A}$ after training on set $\boldsymbol{D_c}$ is. To understand Eq. (1) in more precise way, we consider:

y_i:	i-th instance of output space Y
P:	Probability distribution on $X \times Y$
Q:	Distribution on P
L:	Loss function that maps $Y \times Y \rightarrow \mathbb{R}$
$\mathcal{H}$:	Hypothesis space which is a set of function h: $X \rightarrow Y$
$\mathbb{H}$:	Family of hypothesis space where $\mathcal{H} \in \mathbb{H}$
$\Delta_Q(\mathcal{H})$:	Loss of hypothesis space $\mathcal{H}$ on Q
$\Delta_P(h)$:	Loss of function h on distribution P

The hypothesis space $\mathcal{H}$ is the hypotheses set among which the approximated output space Y is searched, while the loss function, also sometimes called cost the function, is the error between the actual and predicted output. For a regression problem, there are several functions that can be chosen such as mean squared error, average absolute deviation, or mean difference. For a classification problem, it is more common to choose loss functions that map an output probability between 0 and 1, such as cross-entropy error or hinge loss [12]. The goal of bias learning is to find the hypothesis space $\mathcal{H} \in \mathbb{H}$ that minimizes $\Delta_Q(\mathcal{H})$:

$$\begin{aligned} \Delta_Q(\mathcal{H}) &\equiv \int_P \inf_{h \in \mathcal{H}} \Delta_P(h)\, dQ(P) \\ &= \int_P \inf_{h \in \mathcal{H}} \int_{X \times Y} L(h(x), y)\, dP(x, y)\, dQ(P) \end{aligned} \tag{2}$$

Typically, minimizing $\Delta_Q(\mathcal{H})$ cannot be done directly, therefore sampling n times from P according to Q is needed to yield: $P_1,\ldots,P_n$. Furthermore, we sample m times from $X \times Y$ according to each P_i to yield the pairs: $\{(x_i^1, y_i^1),\ldots,(x_i^m, y_i^m)\}$. In the sequel, an (n,m)-sample is denoted by z and written as a matrix in Eq. (3). In order to choose a hypothesis space $\mathcal{H} \in \mathbb{H}$, one needs to minimize the empirical loss on z $\Delta_z(\mathcal{H})$ as in Eq. (4).

$$z \equiv \begin{pmatrix} (x_{11}, y_{11}) & \cdots & (x_{1m}, y_{1m}) \\ \vdots & \ddots & \vdots \\ (x_{n1}, y_{n1}) & \cdots & (x_{nm}, y_{nm}) \end{pmatrix} = \begin{pmatrix} z_1 \\ \vdots \\ z_n \end{pmatrix} \tag{3}$$

$$\Delta_z(\mathcal{H}) \equiv \frac{1}{n} \sum_{i=1}^{n} \inf_{h \in \mathcal{H}} \Delta_{z_i}(h) \tag{4}$$

The bias induced learning algorithm $\hat{A}$ is then defined as [13]:

$$\hat{A}: \bigcup_{n>0, m>0} (X \times Y)^{(n,m)} \rightarrow \mathbb{H} \tag{5}$$

3. Experimental Procedure and Methodology

3.1. Fatigue Testing

To simulate crack growth in an aircraft fuselage, we selected aluminium 7075-T6 for the fatigue testing as it is commonly used for aircraft structures and because of its well known crack propagation behavior. The material parameters for aluminium 7075-T6 are shown in Table 1 (left column) and the test was performed according to the ASTM-E647 standard [14]. The fatigue test parameter is shown in Table 1 (right column). The test setup is shown in Fig. 3, while the dimension of the test specimen is shown in Fig. 4 and the experimental

and empirical crack propagation curve are depicted in Fig. 5, respectively. The empirical crack propagation curve was calculated from AFGROW analysis software from Lextech, Inc. which is available at our department at TU Delft.

Table 1. Left: Material Data [15]. Right: Fatigue Test Parameters According to [14].

Density	2.81 [g/cm³]	Specimen type	M(T)-Spec.
Ultimate Tensile Strength	572 [MPa]	Specimen thickness	6.4 [mm]
Young's Modulus	71.7 [GPa]	Specimen width	100 [mm]
Poisson's Ratio	0.33	Hole diameter ø	12 [mm]
Fatigue Strength	159 [MPa]	Pre-crack length ($2a_0$)	16 [mm]
Fracture Toughness T-L	25 [MPa√m]	Stress ratio R	0.1
Shear Modulus	26.9 [GPa]	Tensile force	32 [kN]
Shear Strength	331 [MPa]	Frequency	2 [Hz]

Fig. 3: Fatigue Test Setup

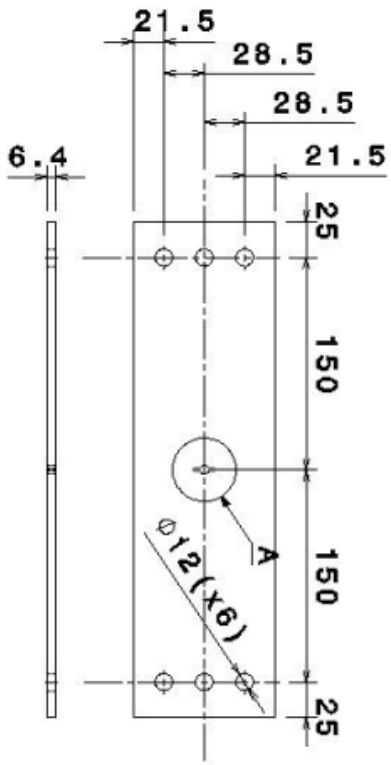

Fig. 4: Specimen Dimension

The test coupon was first modelled in AFGROW and the estimated critical crack length for the coupon was determined to be 80 mm and it would reach 122000 load cycles before failure. The design of experiment was based on this model, however since the hard-drive of the recording computer did not have enough space to store all raw images, it was decided to capture the image at every 10^{th} cycle. During the experiment, the coupon failed at 118030 load cycles and consequently 11803 images which summed up to 47 GB of raw bitmap RGB images were captured by the camera.

We decided to have 4 crack length categories according to the data provided in Fig. 4: no crack, small crack, medium crack, and long crack. The criteria are given in Table 2. Note that fewer or more categorizations can be simplified into binary classification (no crack vs crack) or modified into more categorization (e.g. more than 8 or 12 crack lengths, etc.) depend on the classification need.

Table 2: Crack length categorization criteria, initial notch length ($2a_0$) = 16 mm		
No crack	0 to 44000 cycles	Pre-crack length (16 mm)
Short crack	44001 to 74000 cycles	16.1 mm to 24.0 mm
Medium crack	74001 to 111000 cycles	24.1 mm to 55 mm
Long crack	111001 cycles to failure	more than 55.1 mm

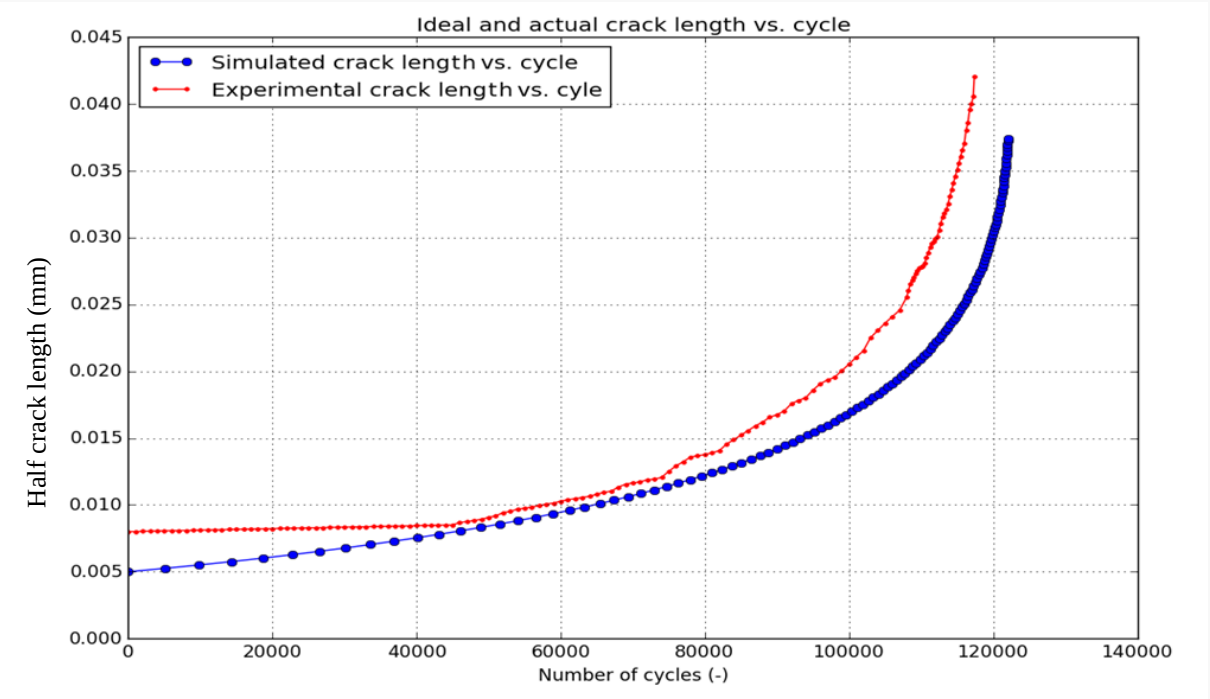

Fig. 5: Crack Propagation Curve

3.2. Pattern Discovery by Clustering Method

In the beginning, we talked about comparing the performance of the biased and non-biased CNN. Instead of manually dividing the fatigue test data in 4 different crack categorization, semi-supervised learning can be introduced to categorize the data according to patterns detected by the computer by pre-training using unsupervised learning.

A widely known method where unsupervised learning is often combined with supervised information to obtain patterns in a particular dataset is clustering, and this combination is called semi-supervised learning since it combines both learning algorithms in a single pipeline. Including a small amount of labelled data to unsupervised learning can lead to a considerable improvement in learning accuracy, but it requires assumptions to be made. The clustering method is based on the assumption that if datapoints share the same cluster, there is a high probability that they are in the same class [17].

Categorizing images with clustering can be performed with the uncomplicated K-means algorithm [18], which is a nearest centroid clustering technique. Concretely, this has athe objective to minimize the sum of the Euclidian distance δ between *n*-datapoints of the *i*-th cluster x_i and the cluster centroid μ_j for *m*-clusters according to:

$$\arg\min \sum_{i=1}^{m}\sum_{j=1}^{n}\delta = \arg\min \sum_{i=1}^{m}\sum_{j=1}^{n}\left(\left\| x_i - \mu_j \right\|\right)^2 \quad (6)$$

Afterwards the mean of the datapoints in each cluster becomes the cluster's new centroid and this process is then repeated for multiple iterations. The human influence in

this clustering method is the choice of the number and the locations of the initial centroids. For the categorization of the fatigue test data the middle image of each group of crack lengths from Table 2 were taken as initial centroids.

3.3. Building a biased-CNN for Aircraft Visual Inspection

From the 11803 RGB images, 9204 were dedicated for training (78%), 1947 for validation (16%) and 652 for testing the trained model (6%). Prior to feeding the CNN with training data, the RGB images were first pre-processed by converting them into grayscale images (i.e. pixel-averaged value) to reduce the computational work.

Furthermore, assuming that the images are rotation and translation invariant, every image was rotated over a uniformly distributed range of angles between -0.2° and 0.2° and zoomed by a factor of 1.0 to 0.2 of the original size. Lastly, half of all the images were flipped horizontally. The pre-processing methods were done by Keras 2.1.6, a high-level neural network API wrapper that simplifies library function commands which is built on top of Tensorflow 1.9 by Google [16], the current most popular back-end open source machine learning library with the most up-to-date contributions. The computational work was performed on an Intel® Core™ i7-6700 HQ CPU.

The architecture of our CNN is depicted in Fig. 5 and consists of four layers of 2D convolutional layers with identical filter size of 3 × 3 pixel, stride of 1, 32 and 64 filters for the first two and last two layers, respectively. A 2D max pooling layer with a pooling size of 2 x 2 pixel was attached to each convolutional layer. The intermediate result was flattened and fed into four dense layers of 16 hidden neurons. The final classification layer consists of only 4 units for the 4 class labels that represent initial, small, medium and large crack and these are the inductive bias that map the input images into 4 crack classifications as per Eq. (5). Thus, any unseen images such as those in the test dataset will be classified according to the crack propagation curve depicted in Fig. 5.

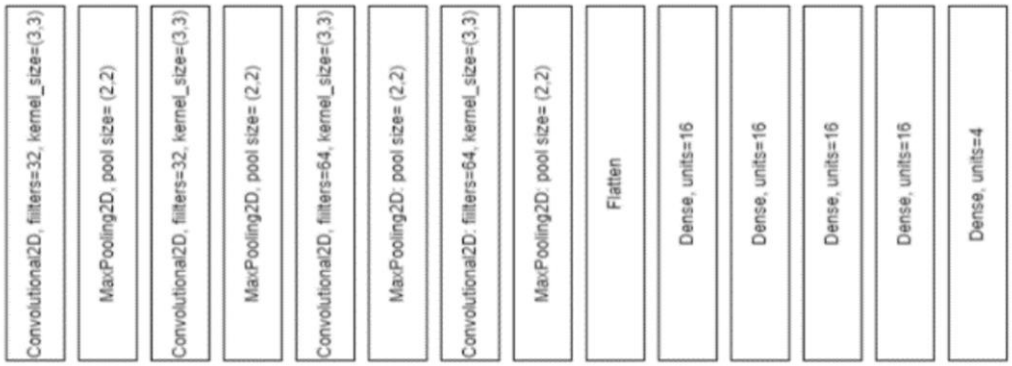

Fig. 5: The architecture of our CNN

4. Results and Discussion

4.1. Semi-Supervised Learning

The clustering method described in section 3.3 was applied to the fatigue test data but the desired result was not achieved. Instead of classifying the images according to their crack lengths, the result after five iterations of the K-means algorithm was that the images were categorized according to the light intensities in the images as depicted in Fig. 6 a – d. Due to large variations in the light intensities during the 23 hours of testing in the TU Delft aircraft hall, the computer did not cluster based on the the crack length.

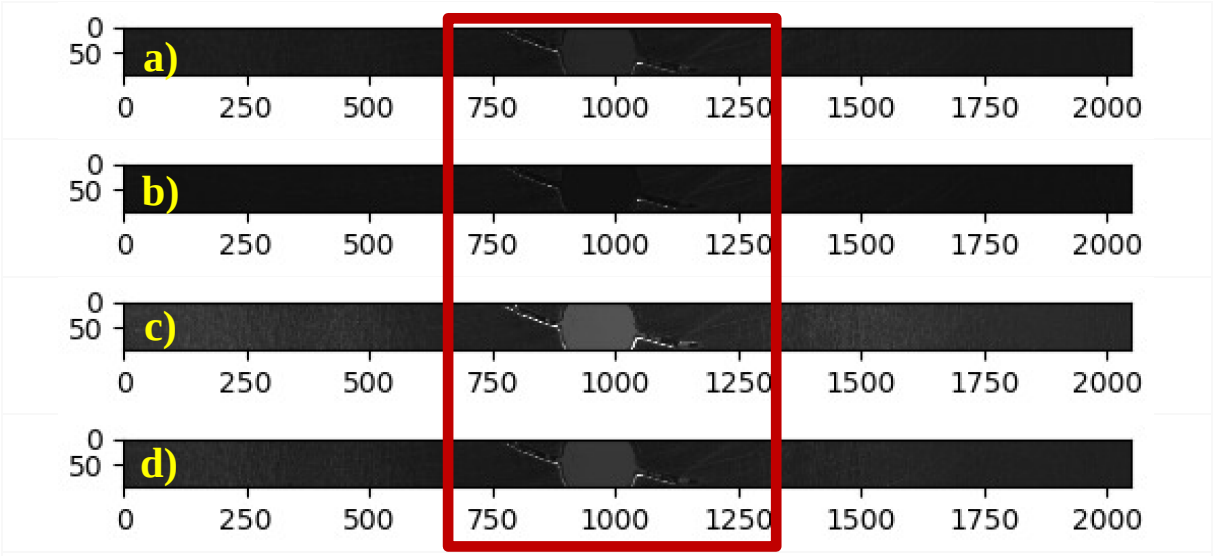

Fig. 6 a – d: Sample images taken from each cluster. X-Axis: Length in pixel, Y-Axis: Width in pixel Difference light intensities through the holes are shown in red rectangle.

Although the four different light intensity classes did not have any meaning from a structural point of view, the images were still fed into our CNN as depicted in Fig. 5 resulting in a 99.45% accuracy in the sixth epoch. This result confirms that semi-supervised learning leads to a considerably high learning accuracy, however it may not make any logical sense from a structural point of view.

4.2. Biased CNN from Supervised Learning

Training the biased CNN described in section 3.2 on the data set consisting of 9204 images for training and 1947 images for validation resulted in a final trained model possessing a training accuracy of 89.15% and validation accuracy of 97.55%, which is slightly lower than the unbiased CNN in Section 4.1.

In addition to this, the confusion matrix, shown in Fig. 7, was also generated to better visualize the correct and incorrect predictions. As it can be seen in Fig. 7, the final trained model experienced the greatest difficulty in making accurate predictions for small/no crack images.

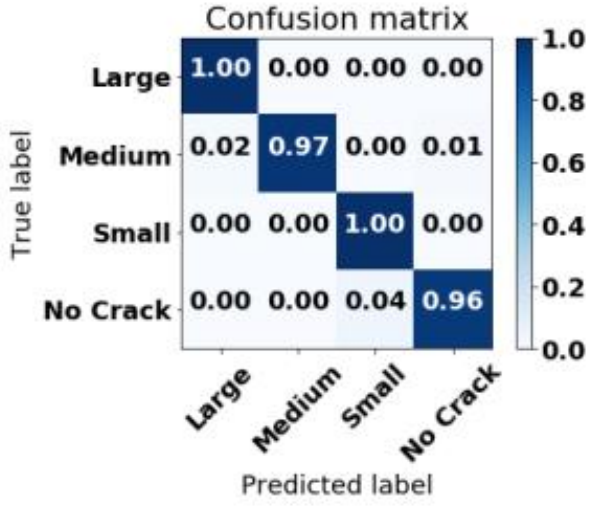

Fig. 7: Normalized confusion matrix of predicted and actual labels

5. Summary and Conclusion

In this paper, we have demonstrated the importance of domain knowledge in automated visual inspection for building a justifiable prediction from deep learning.

The clustering algorithm managed to cluster the 11803 images into 4 different clusters, however not according to 4 different crack lengths as per Table 2, but according to light intensity changes during the image recording. The network trained based on light intensity changes reached 99.45% accuracy. On the other side, the biased CNN that is trained according to different crack length achieved 97.55% accuracy.

These results are consistent with our prior assumption that AI-assisted NDT or Structural Health Monitoring (SHM) should involve human influence rather than just relying on pure unsupervised learning. Nevertheless, we would like also to point out that human decision can also be wrong – therefore we believe that the future NDT and SHM should be based on reciprocal human – computer interaction rather than computer only or human only.

References

[1]. Rice M, Li L, Ying G, Wan M, Lim ET, Feng G, Ng J, Jin Li MT, Bab VS. *Automating the Visual Inspection of Aircraft*. Singapore Aerospace Technology and Engineering Conference, Singapore, 2018.

[2]. Albawi S, Mohammed TA, Al Zawi S. *Understanding of a Convolutional Neural Network*. International Conference on Engineering and Technology (ICET), Antalya, 2017.

[3]. Sawaf F, Groves RM. *Phase Discontinuity Predictions Using Machine-Learning Trained Kernel*. Applied Optics, 2014. Vol. 53: pp 5439 – 5447.

[4]. Zhang L, Yang F, Zhang YD, Zhu YJ. *Road Crack Detection Using Deep Convolutional Neural Network*. IEEE International Conference on Image Processing (ICIP), Phoenix, 2016.

[5]. Cha YJ, Choi W, Büyüköztürk O. *Deep Learning Based Crack Damage Detection Using Convolutional Neural Networks*. Computer Aided Civil and Infrastructure Engineering, 2017. Vol. 32: pp 361 – 378.

[6]. Gopalakrishnan K, Khaitan SK, Choudhary A, Agrawal A. *Deep Convolutional Neural Networks with Transfer Learning for Computer Vision-Based Data-Driven Pavement Distress Detection*. Construction and Building Materials, 2017. Vol. 157: pp 322 – 330.

[7]. Da Silva WRL, De Lucena DS. *Concrete Cracks Detection Based on Deep Learning Image Classification*. 18th International Conference on Experimental Mechanics (ICEM18), Brussels, 2018.

[8]. Li S, Zhao X. *Convolutional Neural Networks-Based Crack Detection for Real Concrete Surface*. SPIE Smart Structures and Materials + Nondestructive Evaluation and Health Monitoring, Denver, 2018.

[9]. Yokoyama S, Matsumoto S. *Development of an Automatic Detector of Cracks in Concrete Using Machine Learning*. Procedia Engineering, 2017. Vol. 171: pp 1250 – 1255.

[10]. LeCun Y, Bottou L, Bengio Y, Haffner P. *Gradient-Based Learning Applied to Document Recognition*. Proc. of IEEE, 1998. Vol. 86: pp 2278 – 2324.

[11]. Mitchell TM. Machine Learning. McGraw Hill, 1997.

[12]. Rosasco L, De Vito E, Caponnetto A, Piana M, Verri A. *Are Loss Functions All The Same?* Journal of Neural Computation, 2003. Vol. 16: pp 1063 – 1076.

[13]. Baxter J. *A Model of Inductive Bias Learning*. Journal of Artificial Intelligence Research, 2000. Vol. 12: pp 149 – 198.

[14]. Standard Test Method for Measurement of Fatigue Crack Growth Rates: ASTM E647 – 15E1. American Society for Testing and Materials.

[15]. Aluminum 7075-T6; 7075-T651. ASM Aerospace Specifications Metals Inc. Site available online: http://asm.matweb.com/search/SpecificMaterial.asp?bassnum=ma7075t6 Last accessed: 30-AUG-2018

[16]. Tensorflow. Site available online: https://www.tensorflow.org/ Last accessed: 30-AUG-2018..

[17]. Chapelle O, Schölkopf B, Zien A. *Semi-Supervised Learning*. The MIT Press, 2006. pp 5 – 6.

[18]. Ju W. *Advances in K-means Clustering: A Data Mining Thinking.* Springer Publishing Company, 2012.