# Introduction to Convolutional Neural Networks

Sakib Ashraf Zargar

Department of Mechanical and Aerospace Engineering, North Carolina State University, Raleigh, North Carolina 27606, USA

## Table of Contents

1. Introduction to Deep Learning ..... 1
2. ANNs vs CNNs for Images ..... 2
3. Convolutional Neural Networks ..... 3
4. Common CNN models and Transfer Learning ..... 5

## 1. Introduction to Deep Learning

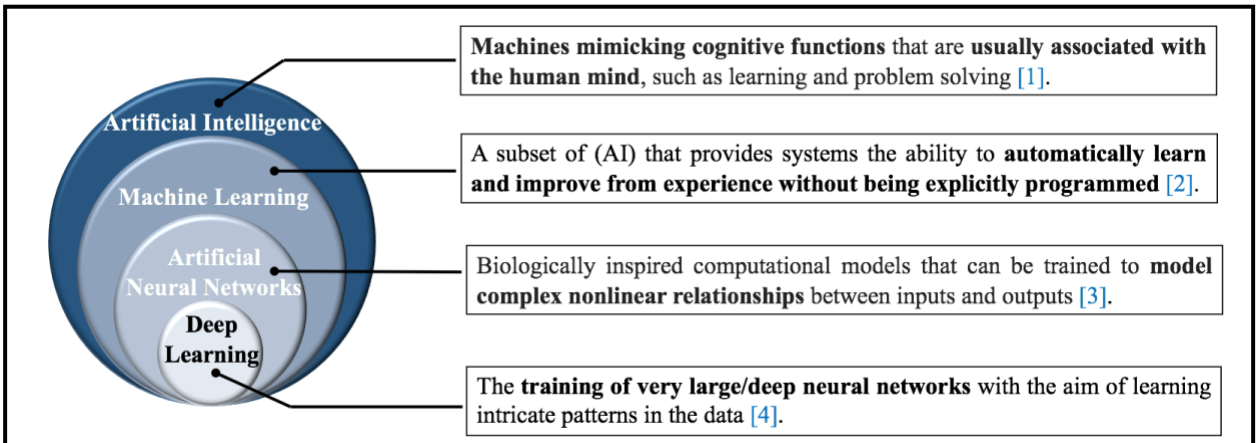
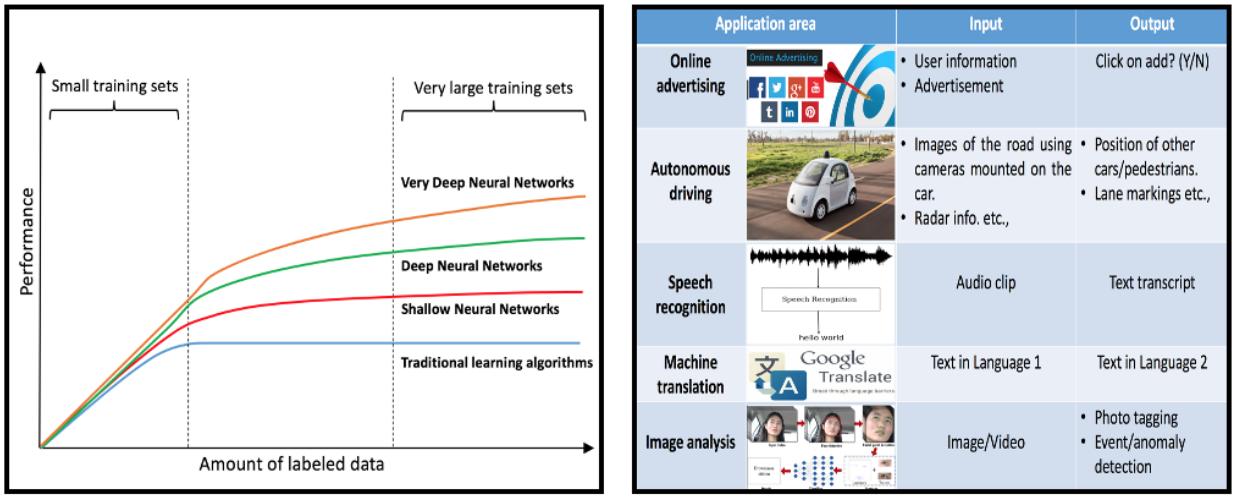



Figure 1: Deep learning as a subset of machine learning (ML) and artificial intelligence (AI).

John McCarthy, widely regarded as one of the founding fathers of artificial intelligence (AI), defined AI as the science and engineering of making intelligent machines. While this can be achieved, to some extent, by explicitly programming the machines; machine learning (ML) is an approach to achieve AI (i.e., it is a subset of AI) that involves learning from experience/historical data without the need for explicit programming. ML uses various numerical and statistical approaches including (but not limited to) artificial neural networks (ANNs) which are biologically inspired computational models that can be trained to model complex nonlinear relationships between inputs and outputs. Conventional ML techniques (including feed-forwards ANNs) are limited in their ability to process high-dimensional data in their raw form. Oftentimes, careful engineering and domain expertise is required to extract suitable features from the raw data which are then fed to the ML model. Deep learning (DL), which refers to the training of specialized deep

neural networks, circumvents this problem by automatically discovering representations in the data (so-called automatic feature extraction) thereby allowing the data to be used in their raw form. This has led to a dramatic improvement in the state-of-the-art for many AI applications like object detection [5][6], natural language processing [7][8] etc.




(a) (b)

Figure 2: (a) Performance of ML algorithms as a function of the available data (b) Various application areas of DL.

Figure 2(a) shows the performance of various learning algorithms as a function of the available data. In the big data regime, deep learning models almost always outperform traditional learning models. Figure 2(b) shows some of the common application areas of DL. The choice of the DL model/network-architecture employed for a particular task depends on the input data type and how the available information is encoded: Images, for example, have information encoded in space and the data comes in the form of multiple arrays. Sequences, on the other hand, have information encoded in time and the data has an inherent temporal structure which must be respected during the problem formulation.

## 2. ANNs vs CNNs for Images

Figure 3 shows a standard feed-forward neural network (ANN)/multi-layer perceptron (MLP) used for image analysis. The input image is unfolded into a vector which is then fed into the ANN. For a 1 Mega Pixel RGB image, the length of the unfolded vector = the number of pixels = $n_0 = 3\times10^6$. In an ANN, each neuron in a particular layer is fully connected to all the neurons in its adjoining layers. If the first layer of the ANN has $n_1 = 10^3$ neurons, then the weight vector $\boldsymbol{W}^{[1]}$ corresponding to the first layer has $3\times10^6\times10^3$ = 3 billion tunable parameters. For such a large number of parameters, the number of training examples has to be very large, otherwise it will result in overfitting of the data.

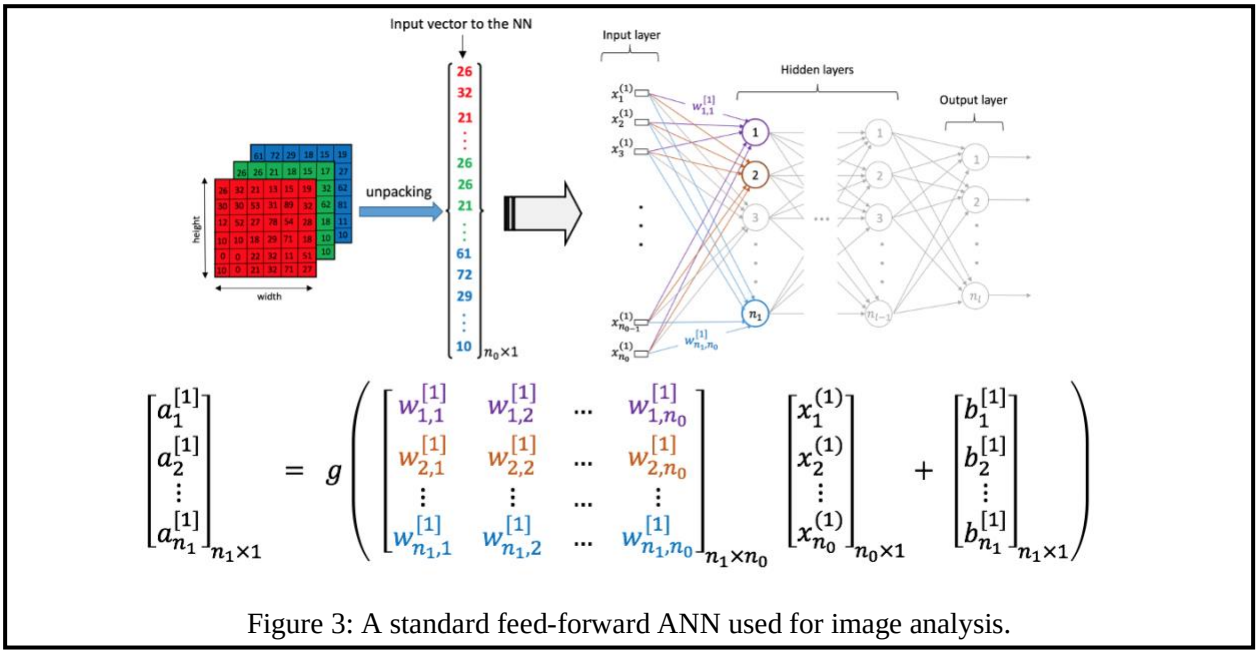



Figure 3: A standard feed-forward ANN used for image analysis.

## 3. Convolutional Neural Networks

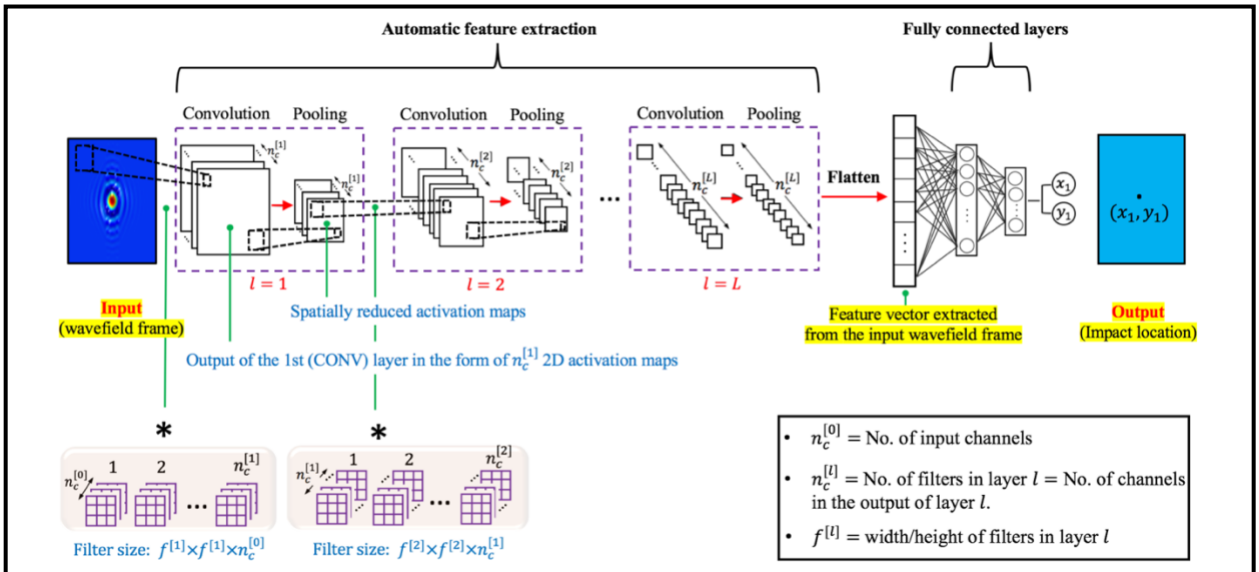



Figure 4: A sample CNN architecture.

A modification of the traditional feed-forward neural network is the convolutional neural network (also called as ConvNet or CNN). CNNs draw inspiration from the mammalian visual cortex and have become the de-facto standard for analyzing image data. In its most basic form, a CNN consists of three types of layers:

**1. Convolutional (CONV) layers:** These are the basic building blocks of a CNN and each CONV layer $l$ consists of a set of $n_c^{[l]}$ filters/kernels with trainable parameters/weights. A filter in layer $l$ can be thought of as a 3-D volume of $f^{[l]} \times f^{[l]} \times f^{[l]}$ neurons, where $f^{[l]}$ is the width/height and

$n_c^{[l-1]}$ is the depth of the filter. The superscript ($l$-1) for the depth of the filter points to the fact that the depth of a filter in layer $l$ has to be equal to depth of the output of layer ($l$-1) which in turn is equal to the number of the filters in layer ($l$-1). For the first Conv layer, it is simply the depth of the input image (3 in case of a color image and 1 in case of a grey-scale image). The convolution operation that lies at the heart of the CNN performs the task of feature extraction with filters in the earlier layers extracting low level features while the deeper layers look for more high-level features. Unlike the classical computer-vision techniques, where the parameters of these filters are hand-crafted, CNNs have the ability to learn these filter weights by themselves through back-propagation when provided with sufficient training data. Only the number of filters $n_c^{[l]}$ and the size of these filters $f^{[l]}$ has to be specified for each layer. It is important to note that, at a time, a filter is connected only to a local region of the input (i.e., there is **sparsity of connections**) and the parameters learned for a particular filter are used throughout the spatial extent of the image (i.e., **parameter sharing** is involved). In-fact, it is because of these two features that CNNs become a viable option for image data unlike the fully connected ANNs where-in each neuron in a particular layer is fully connected to all the neurons in its adjoining layers there-by making the number of trainable parameters infeasibly large.

**2. Pooling (POOL) layers:** The output of the CONV layer is normally fed into a pooling layer. The function of the pooling layers is to progressively reduce the spatial extent (height, width) of the representation without affecting its depth. Doing so reduces the number of trainable parameters in the network there-by reducing the computational cost and controlling overfitting. The convolutional layers and the pooling layers make up the feature extraction part of the CNN.

**3. Fully connected (FC) layers:** These are just like the fully connected layers in an ordinary ANN. The output of the feature extractor part of the CNN is flattened and represents a reduced version

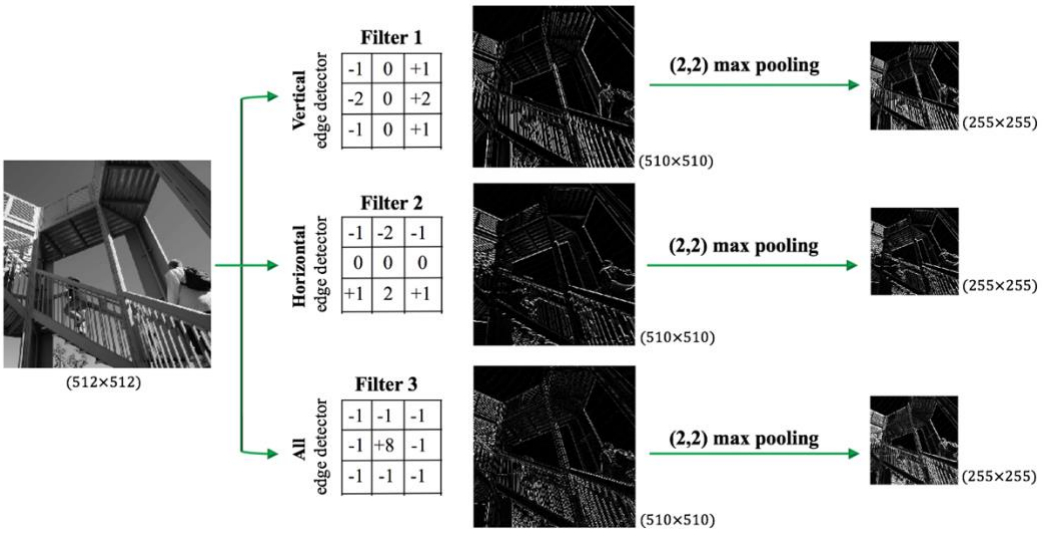



Figure 5: Convolution and pooling in action.

of the original input. This is then fed into one or many FC layers followed by an output layer. For a detailed understanding of CNNs, the reader is referred to [9][10][11][12][13]. In recent years,

CNNs have shown a lot of promise across a wide variety of disciplines including structural health monitoring (SHM) [14][15][16][17][18][19][20][21][22]. Figure 5 shows the effect of convolution and pooling layers on the input image.

## 4. Common CNN models and Transfer Learning

Rather than training all the weights of a network from scratch, much faster progress can often be made by downloading weights that someone else has already trained on the network architecture and use that as pre-training and transfer that to a new task that one might be interested in. This is referred to as transfer learning and based on the size of the new data set available for training, one may freeze all but the last (small dataset available), some (medium dataset available) or none (large dataset available) of the layers of the base network. In the last case, the weights of the base network simply act as initial weights. Some common CNN models used as base models during transfer learning are shown next:

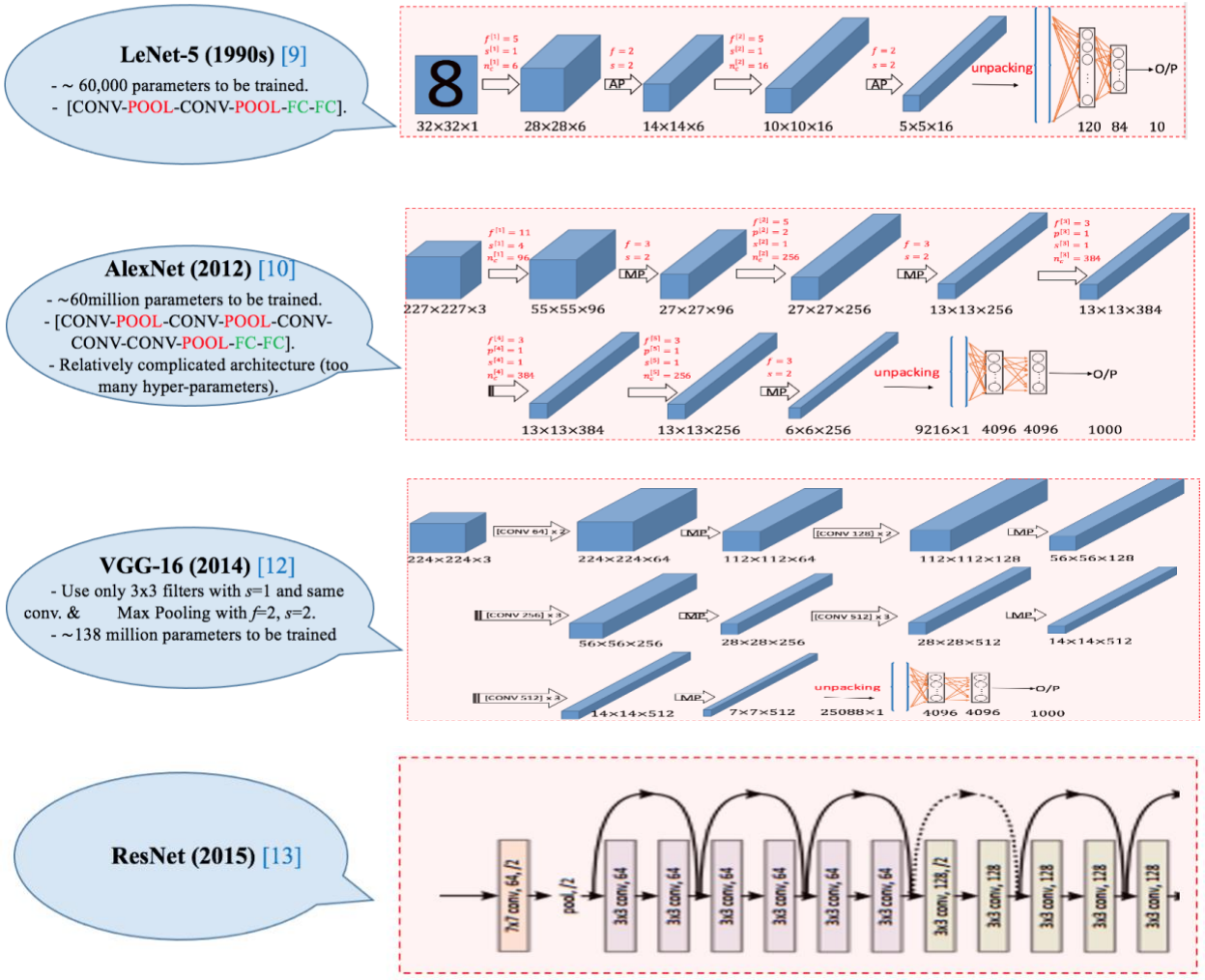



Figure 6: Common CNN models used for transfer learning.

## References


[1] Russell, S. J., & Norvig, P. (2016). *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited.

[2] Bishop, C. M. (2006). *Pattern recognition and machine learning*. springer.

[3] Jain, A. K., Mao, J., & Mohiuddin, K. M. (1996). Artificial neural networks: A tutorial. *Computer*, *29*(3), 31-44.

[4] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature*, *521*(7553), 436.

[5] Zhao, Z. Q., Zheng, P., Xu, S. T., & Wu, X. (2019). Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, *30*(11), 3212-3232.

[6] Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., & Pietikäinen, M. (2020). Deep learning for generic object detection: A survey. *International journal of computer vision*, *128*(2), 261-318.

[7] Young, T., Hazarika, D., Poria, S., & Cambria, E. (2018). Recent trends in deep learning based natural language processing. *ieee Computational intelligenCe magazine*, *13*(3), 55-75.

[8] Deng, L., & Liu, Y. (Eds.). (2018). *Deep learning in natural language processing*. Springer.

[9] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278-2324.

[10] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097-1105).

[11] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1-9).

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

[13] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

[14] Zargar, S. A., & Yuan, F. G. (2020). Impact diagnosis in stiffened structural panels using a deep learning approach. *Structural Health Monitoring*, 1475921720925044.

[15] ZARGAR, S. A., & YUAN, F. G. (2019). A deep learning approach for impact diagnosis. *Structural Health Monitoring 2019*.

[16] Yuan, F. G., Zargar, S. A., Chen, Q., & Wang, S. (2020, April). Machine learning for structural health monitoring: challenges and opportunities. In *Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems 2020* (Vol. 11379, p. 1137903). International Society for Optics and Photonics.

[17] Wang, S., Zargar, S. A., & Yuan, F. G. (2020). Augmented reality for enhanced visual inspection through knowledge-based deep learning. *Structural Health Monitoring*, 1475921720976986.

[18] WANG, S., ZARGAR, S. A., XU, C., & YUAN, F. G. (2019). An efficient augmented reality (AR) system for enhanced visual inspection. *Structural Health Monitoring 2019*.

[19] Khajwal, A. B., & Noshadravan, A. (2020). Probabilistic hurricane wind-induced loss model for risk assessment on a regional scale. *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering*, *6*(2), 04020020.
[20] Khajwal, A. B., & Noshadravan, A. (2021). An uncertainty-aware framework for reliable disaster damage assessment via crowdsourcing. *International Journal of Disaster Risk Reduction*, *55*, 102110.
[21] Lyathakula, K. R., & Yuan, F. G. (2021, March). Fatigue damage prognosis of adhesively bonded joints via a surrogate model. In *Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems 2021* (Vol. 11591, p. 115910K). International Society for Optics and Photonics.
[22] Lyathakula, K. R., & Yuan, F. G. (2021, March). Probabilistic fatigue life prediction for adhesively bonded joints via surrogate model. In *Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems 2021* (Vol. 11591, p. 115910S). International Society for Optics and Photonics.