computation

MDPI

Article

A Hybrid Physics-Informed and Data-Driven Approach for Predicting the Fatigue Life of Concrete Using an Energy-Based Fatigue Model and Machine Learning

Himanshu Rana * and Adnan Ibrahimbegovic

Laboratoire Roberval, Centre de Recherche Royallieu, University of Technology of Compiègne-Alliance Sorbonne University, 60200 Compiègne, France; adnan.ibrahimbegovic@utc.fr
* Correspondence: himanshu.rana@utc.fr

Abstract: Fatigue has always been one of the major causes of structural failure, where repeated loading and unloading cycles reduce the fracture energy of the material, causing it to fail at stresses lower than its monotonic strength. However, predicting fatigue life is a highly challenging task and, in this context, the present study proposes a fundamentally new hybrid physics-informed and data-driven approach. Firstly, an energy-based fatigue model is developed to simulate the behavior of concrete under compressive cyclic fatigue loading. The data generated from these numerical simulations are then utilized to train machine learning (ML) models. The stress–strain curve and S-N curve of concrete under compression, obtained from the energy-based model, are validated against experimental data. For the ML models, two different algorithms are used as follows: k-Nearest Neighbors (KNN) and Deep Neural Networks (DNN), where a total of 1962 data instances generated from numerical simulations are used for the training and testing of the ML models. Furthermore, the performance of the ML models is evaluated for out-of-range inputs, where the DNN model with three hidden layers (a complex model with 128, 64, and 32 neurons) provides the best predictions, with only a 0.6% overall error.

Keywords: fatigue life; S-N curve; energy-based fatigue model; machine learning; k-nearest neighbors; deep neural networks

check for updates

Academic Editors: Andrés Amador Garcia-Granada and Vagelis Harmandaris

Received: 8 January 2025
Revised: 10 February 2025
Accepted: 21 February 2025
Published: 2 March 2025

Citation: Rana , H.; Ibrahimbegovic, A. A Hybrid Physics-Informed and Data-Driven Approach for Predicting the Fatigue Life of Concrete Using an Energy-Based Fatigue Model and Machine Learning. *Computation* **2025**, *13*, 61. https://doi.org/10.3390/computation13030061

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Structural failure has been a major concern in engineering design throughout history. This concern dates back to ancient times, such as during the era of Hammurabi, where the collapse of a building causing the death of its owner resulted in the builder being sentenced to death [1]. One of the major causes of structural failure is fatigue, where the material fails due to repeated loading and unloading cycles, ultimately leading to failure at stress levels much lower than the material's experimental strength [2,3]. The fatigue life of a structure is divided into the following three stages: crack initiation, crack propagation, and final failure. In the first stage, small cracks form at the micro level, which then nucleate and coalesce into a dominant macro crack. In the second stage, this macro crack propagates stably until it reaches a critical length. Finally, the structure enters the last stage, leading to failure. Predicting the fatigue life of a structure has always been a challenging task for engineers. Different methods used for predicting fatigue life are shown in Figure 1. The first recorded observations of fatigue phenomena were made by Wilhelm August Julius Albert in 1837, where he noted from their tests that failure due to fatigue results from repeated cycles of loads rather than accidental overloads [4]. However, the major breakthrough in the

study of fatigue occurred with the introduction of the Wöhler curve, also known as the S-N curve, by Albrecht Wöhler in 1860, which represents the fatigue life as the number of cycles needed to fail a material at a given stress level, based on experimental data [5]. A typical S-N curve is divided into finite-life and infinite-life regions. Additionally, the finite-life region is further divided into low-cycle fatigue and high-cycle fatigue. Later, this curve was used by Palmgren and Miner to develop the concept of cumulative damage for predicting fatigue life [6,7]. The next significant advancement in the study of fatigue life came with the introduction of the Paris law, which relates crack growth rate to the stress intensity factor [8,9]. These concepts remain integral to the study of material fatigue life to this day, and they continue to be incorporated into various design codes, such as those in [10,11].

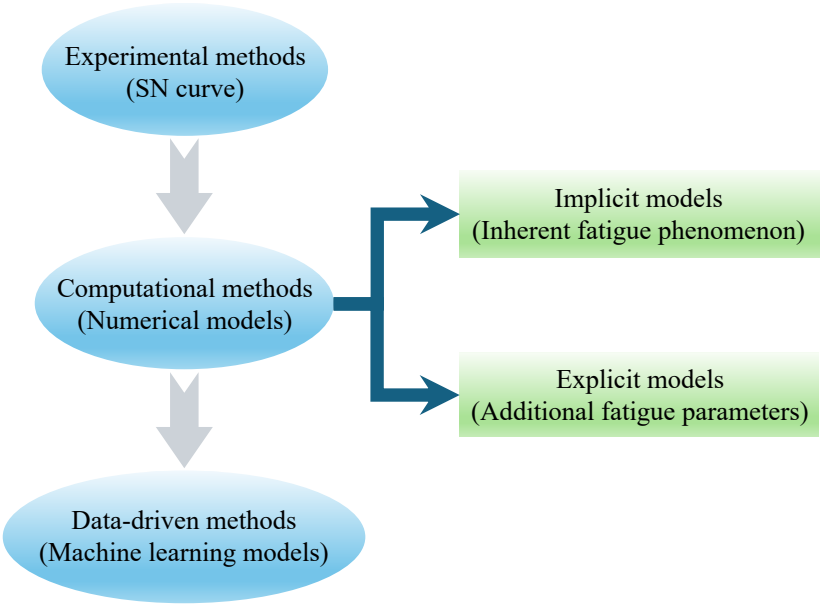


Figure 1. Methods for fatigue life prediction.

With the advancement of computational tools, physics-based numerical models have become increasingly popular, which has shifted research on predicting fatigue life solely from experiments to numerical computations. These models provide a more economical and time-efficient approach compared to traditional experimental methods. The fatigue phenomenon is generally embedded into these models by associating it with progressive crack growth due to fatigue loading, without the need to explicitly define new parameters for fatigue. Initial attempts involved formulating constitutive models based on cohesive force laws, where the traction near the crack tip is related to the crack opening displacement [12,13]. The graph between traction and crack opening displacement forms a locus of points with decreasing traction as the crack propagates, while the loading and unloading cycles generate hysteresis loops, gradually deteriorating material stiffness and accumulating plastic strains [14,15]. Subsequently, more generalized models for crack phenomena were developed that combined damage and plasticity, with cracking defined by incorporating softening laws into the constitutive models. This approach was employed by various authors such as [16–19]. In [16], the author described a constitutive model that uses a plasticity model during loading and a damage model during unloading. Meanwhile, in [17], a strong plastic yielding model was used at the micro-level to dissipate energy, which contributes to fatigue failure, while the structure remains elastic at the macro-level. In [18], a coupled damage–plasticity model was developed using a hybrid stress-based formulation, providing a non-iterative solution at the local level for linear laws, with the crack represented using the Embedded Discontinuity Finite Element Method (ED-FEM).

The numerical methods discussed above do not explicitly define the fatigue-related parameters, as fatigue is considered inherent in these models. However, parallel research efforts are focused on developing fatigue models that incorporate fatigue variables which evolve based on the load history [20–23]. Much of this work centers on phase-field models, where a discontinuous crack is represented by a continuous but steep transition from completely intact material to cracked material through the incorporation of a phase-field variable [24–27]. In [20,21], the authors introduced a history variable to drive the fatigue phenomenon as follows: strain accumulation in [20] and elastic energy accumulation in [21]. This variable accumulates only during loading and is responsible for reducing the fracture energy of the material. Additionally, a fatigue degradation function, dependent on the same fatigue variable, is defined to modify the dissipated energy. Phase-field models have become popular due to their ability to simulate complex crack behaviors, such as branching and merging. However, a significant drawback of these models is their mesh dependence on the length scale parameter, which necessitates the use of a very fine mesh, thereby increasing computational cost. Additionally, determining an appropriate value for the length scale parameter is challenging. As demonstrated in [21], varying the length scale parameter can lead to significant changes in the predicted fatigue life.

Although studies on the fatigue life of structures and materials have been conducted for a long time, earlier research primarily focused on building theoretical knowledge, with practical applications limited to simple examples. In the modern world, problems have become more complex, and with an increasing emphasis on sustainable and efficient development, the advanced numerical models discussed earlier have been developed to address these challenges. However, as model complexity increases, computational time grows exponentially. This presents the following new challenge for researchers: developing computationally efficient models that can accurately predict the fatigue life of structures. One such approach is to use data-driven machine learning (ML) models, which have already been proven in various fields [28,29] but their capabilities in the material engineering field is yet to be fully explored. Moreover, with the increasing popularity and numerous advantages of ML models, various researchers have used data-driven models to solve a variety of civil engineering problems [30,31]. Furthermore, machine learning has become such a hot and popular research topic globally that it is pushing researchers to integrate it in their research irrespective of their field of study [32,33]. In the context of fatigue life prediction for materials, various researchers have employed ML models to estimate the fatigue life of metals [34–37]. For instance, in [34], the authors employed support vector machine models to predict the fatigue life of titanium parts fabricated through selective laser melting, accounting for the defects introduced during the manufacturing process. Similarly, in [35], the authors utilized random forest models to predict the fatigue life of titanium alloys. A comprehensive review of various machine learning techniques for predicting the fatigue life of metals is presented in [36], where the authors explore advancements in hybrid physics-informed and data-driven methods that integrate physics-based models with data-driven approaches. Most of these studies focus on the fatigue life of metals, and very few explore the applicability of ML models in predicting the fatigue life of concrete [38,39].

In the present study, a hybrid physics-informed and data-driven method is proposed for predicting the fatigue life of concrete structures. The data are generated using a novel energy-based fatigue model, where damage is governed by the accumulation of energy in the material. The evolution equations for damage and plasticity are derived using the second law of thermodynamics and the principle of maximum dissipation, where temperature effects are ignored since the study is limited to static tests with low-frequency strain loading on brittle materials; therefore, the heat generated by inelastic dissipation can

be neglected. Sources of energy dissipation in this case arise due to microcrack formation and dislocation movements, causing damage and plasticity in the material, respectively. These data are then utilized to train ML models, employing algorithms such as k-Nearest Neighbors (KNN) and Deep Neural Networks (DNN) to predict fatigue life. In Section 2, the theoretical formulation and numerical implementation of the energy-based fatigue model within the finite element method framework are presented. Section 3 provides a detailed discussion on dataset classification, the theoretical background of the learning algorithms, and model evaluation for ML models. Section 4 presents the numerical validation of the energy-based fatigue model and ML models against the experimental results. Finally, the discussion and concluding remarks are provided in Sections 5 and 6, respectively.

2. Energy-Based Fatigue Model

Fatigue is a phenomenon that occurs when a material is subjected to repetitive loading and unloading cycles, leading to the formation of microcracks. As these microcracks grow in number and size, they reduce the fracture energy of the material. Eventually, the material reaches its ultimate stress limit, which is lower than its initial limit; this reduction depends on the number of fatigue cycles the material has undergone. Consequently, the material experiences softening at lower stress levels, ultimately leading to failure. To address this phenomenon, this section proposes an energy-based fatigue model, which is formulated by introducing energy in the material as a criterion governing the development of microcracks, leading to damage and plasticity in the material. This process ultimately reduces the energy required to fracture the material. As illustrated in Figure 2, as the number of cycles approaches failure, the fracture energy decreases due to the reduction in the traction limit at the discontinuity. The reduction in fracture energy is governed by the evolution of the fatigue variable.

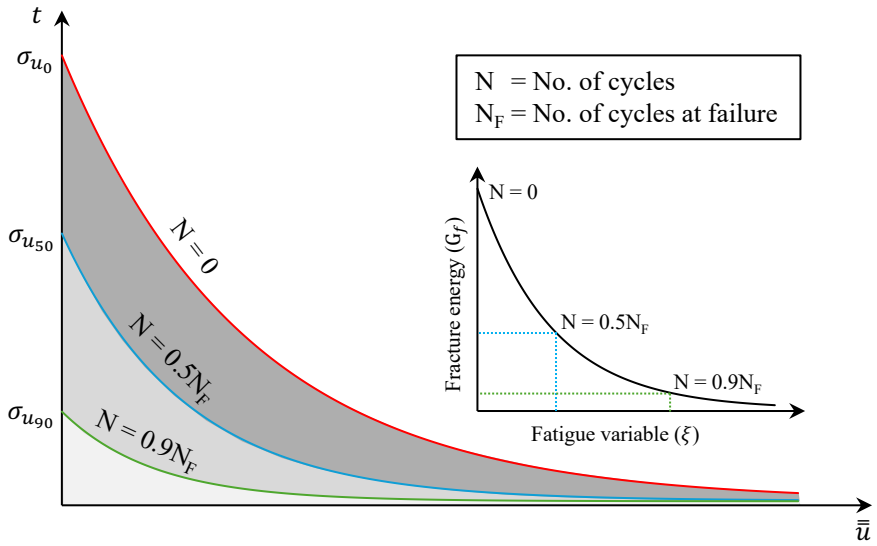


Figure 2. Traction-localized displacement curve illustrating fracture energy reduction under fatigue loading cycles.

The discontinuity is modeled using the Embedded Discontinuity Finite Element Method (ED-FEM) [40,41]. The typical criteria used in ED-FEM for damage and plasticity are stress-based. However, the issue with such criteria is that they trigger damage and plasticity phenomena only after exceeding specific stress limits, which are typically derived from standard laboratory tests. In contrast, the fatigue process is observed to cause damage even at very low stress levels, making it impractical to use stress limits as the governing parameter. This challenge is addressed by using energy as the primary governing parameter instead of stress. The proposed model, incorporating the energy criterion, is further enhanced with

kinematic plastic hardening and softening models to account for the monotonic loading behavior of the material [42,43]. In the following subsections, the theoretical formulation of the model is presented, along with a discussion on its numerical implementation.

2.1. Theoretical Formulation

In this section, the theoretical formulation of the energy-based fatigue model is presented. First, the kinematics of the finite element model is established, where the total strain ($\boldsymbol{\varepsilon}$) in the element is decomposed into elastic strain ($\boldsymbol{\varepsilon}^e$) and plastic strain ($\boldsymbol{\varepsilon}^p$) as

$$\boldsymbol{\varepsilon} = \boldsymbol{\varepsilon}^e + \boldsymbol{\varepsilon}^p \tag{1}$$

With this decomposition, the constitutive behavior of the material is defined by the stress–strain relationship, given by

$$\begin{aligned} \boldsymbol{\sigma} &= \mathbf{D}^{-1}\boldsymbol{\varepsilon}^e \\ \boldsymbol{\sigma} &= \mathbf{D}^{-1}(\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}^p) \end{aligned} \tag{2}$$

where $\boldsymbol{\sigma}$ is the stress vector, and $\mathbf{D}$ is the damage compliance matrix.

After establishing the constitutive model, the next step is to define the criterion that governs the evolution of damage and plasticity in the material. In this context, an energy-based criterion is employed, where the accumulation of energy during loading drives the evolution of damage and plasticity, while no damage or plasticity occurs during unloading. However, since the model incorporates kinematic hardening, plasticity evolution during unloading is attributed to kinematic hardening rather than fatigue. The energy criterion (ϕ) is defined as

$$\begin{aligned} &0 \leq \phi = \mathrm{G} - (\mathrm{G}_0 - q) \\ &\mathrm{G} = \int_t \dot{\mathrm{G}} d\tau;\ \dot{\mathrm{G}} = \begin{cases} \boldsymbol{\sigma}.\dot{\boldsymbol{\varepsilon}} & \text{if } \boldsymbol{\sigma}.\dot{\boldsymbol{\varepsilon}} > 0 \\ 0 & \text{if } \boldsymbol{\sigma}.\dot{\boldsymbol{\varepsilon}} \leq 0 \end{cases} \end{aligned} \tag{3}$$

where G is the accumulated energy during loading, G_0 is the limit value of the accumulated strain energy to initiate fatigue, and q is the energy-like fatigue variable representing the total energy input to the material responsible for fatigue which is defined as

$$q = \int_t \dot{q} d\tau;\ \dot{q} = -K\dot{\xi} \tag{4}$$

where K is the fatigue modulus, and ξ is the fatigue variable. The evolution equations for damage, plasticity, and the fatigue variable are then derived by maximizing the total material dissipation [44] subject to the energy criterion and can be expressed as

$$\dot{\mathbf{D}}\boldsymbol{\sigma} + \dot{\boldsymbol{\varepsilon}}^p = \frac{\partial \phi}{\partial \boldsymbol{\sigma}}\dot{\gamma};\ \dot{\xi} = \frac{\partial \phi}{\partial q}\dot{\gamma} \tag{5}$$

where $\dot{\gamma}$ is the Lagrange multiplier associated with the energy criterion. These evolution equations can subsequently be used to calculate the tangent modulus of the material by substituting them into the constitutive equation. Finally, the fatigue life of the material can be related to the remaining fracture energy, which depends on the accumulation of energy and is expressed as

$$\mathrm{G}_f = \mathrm{G}_{f_0}\left(1 - \frac{\mathrm{G}}{\mathrm{G}_T}\right) \tag{6}$$

where G_f is the current fracture energy, G_{f_0} is the initial fracture energy, and G_T is the total accumulated energy at fracture. Therefore, the number of cycles at failure (N_F) correspond to the number of cycles required to reduce the fracture energy of the material to zero.

2.2. Numerical Implementation

The numerical implementation of the model is carried out using the operator split method. In this method, the variables are divided into global variables and local variables. The global variables represent the nodal displacements, while the local variables are the internal variables that define the constitutive behavior of the material. By adopting this approach, only the displacements are solved globally, while the internal variables are solved independently for each element locally. For each iterative guess of the global variables, the true values of the local variables are first determined and subsequently used to update the global variables. This process is repeated until convergence is achieved for the given time step. The computational algorithm used to achieve this is illustrated in Figure 3.

As shown in Figure 3, with the best iterative guess of the nodal displacements ($\mathbf{d}_{n+1}^{(i)}$) at time t_{n+1}, where i is the iteration number and $\mathbf{d}_{n+1}^{(i=0)} = \mathbf{d}_n$, the strain is calculated first. Assuming an elastic trial step (i.e., no evolution of internal variables), the stresses are then computed. Subsequently, the trial value of the energy criterion function is evaluated. If $\phi_{n+1} \leq tol$, the process proceeds to the global level. Otherwise, the internal variables are updated, and the stresses and criterion functions are recalculated. This process is repeated until convergence is achieved.

Once convergence is achieved at the local level, the solution proceeds to the global level, where the goal is to update the nodal displacements based on the corresponding updated internal forces obtained from the local level computations. The iterative process continues until the global convergence criterion is satisfied (i.e., $\mathbf{r}_{n+1} \leq tol$).

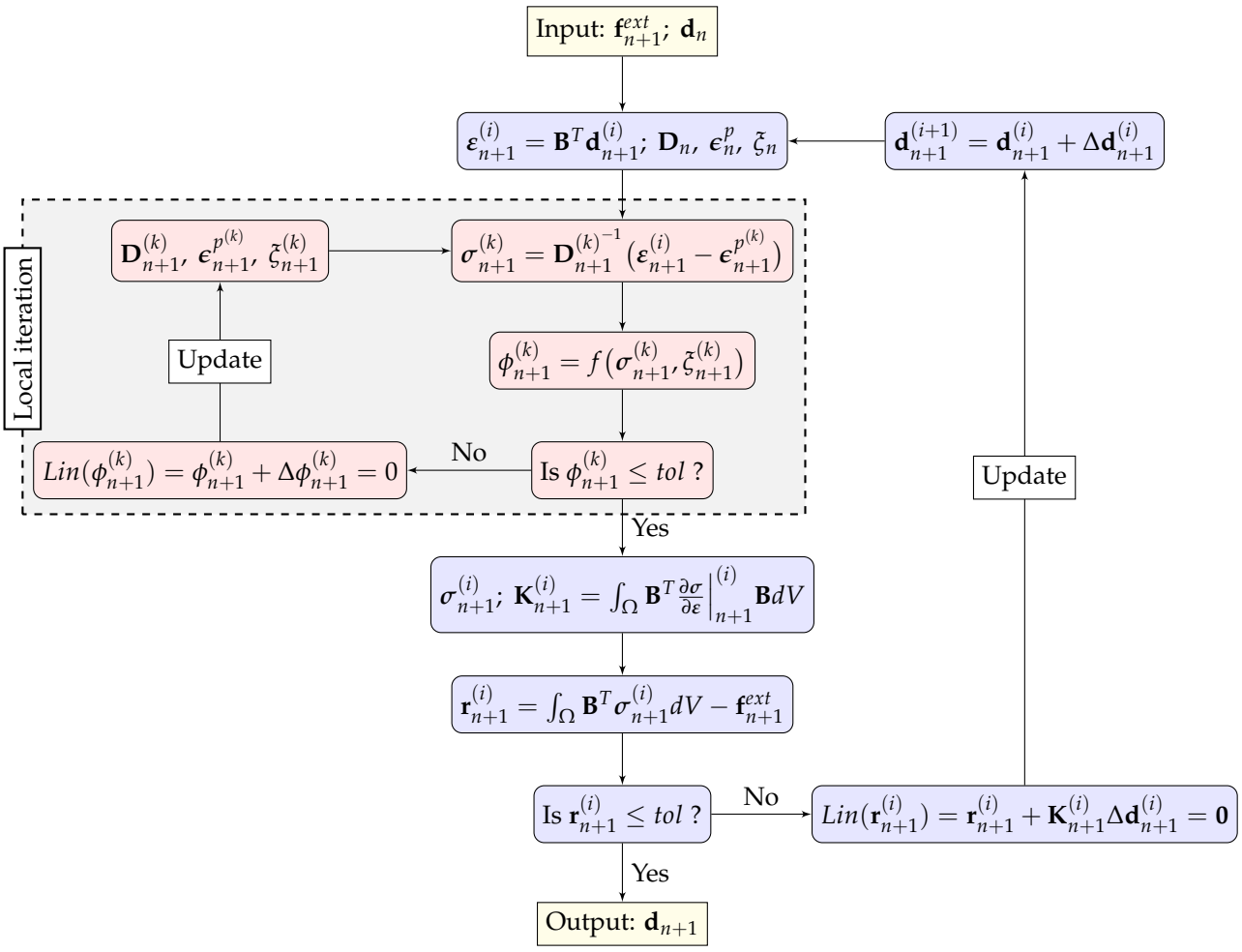


Figure 3. Computational algorithm for energy-based fatigue model.

3. Data-Driven Fatigue Prediction

In this section, a data-driven fatigue prediction model using machine learning (ML) algorithms is presented. A typical framework of the ML model is illustrated in Figure 4, where the entire process is divided into three stages. *Stage I:* Data generation, achieved either through experimental results or physics-based numerical simulations. *Stage II:* The learning algorithm is employed to uncover hidden patterns in the data and establish relationships between the input and output. *Stage III:* Model evaluation, which tests the accuracy of the model on unseen inputs. Each of these stages is discussed in detail in the following subsections.

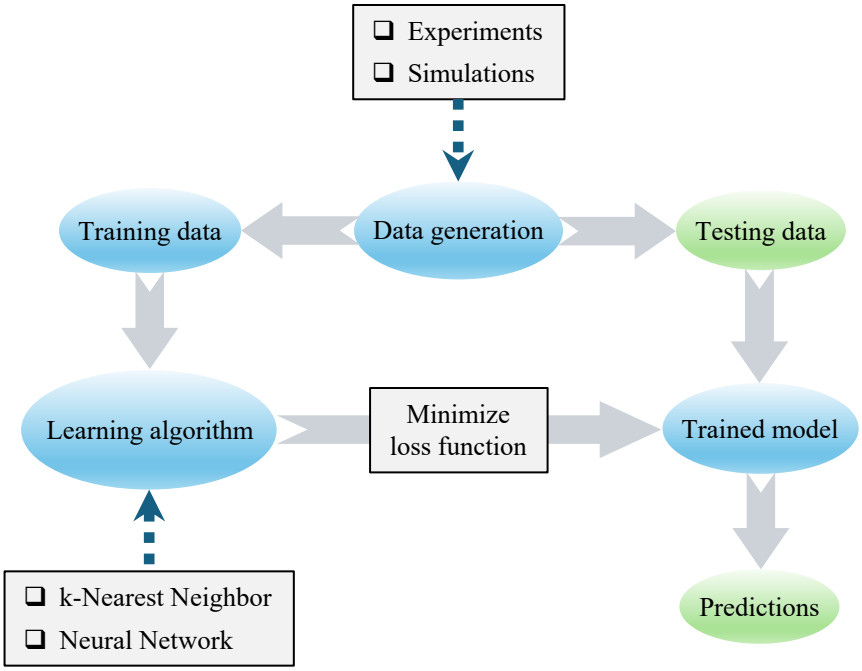


Figure 4. A typical framework for machine learning model.

3.1. Data Generation

Data generation is a critical aspect of any machine learning model, as the model's performance largely depends on the quality and quantity of the data provided. The quality of the input data relies on the selection of appropriate attributes and features. An attribute represents a broader classification of the data, while a feature refers to a specific property within that classification. For instance, in fatigue life prediction, material parameters are an example of an attribute, and within material parameters, specific features include the elastic modulus, hardening modulus, and so on, which serve as direct inputs to the ML model. The number of features is also tied to the amount of data needed to achieve a reasonably accurate ML model. More features typically require more data to establish a robust relationship between each feature and the output. Therefore, it is crucial to select features that have a meaningful impact on the output and exclude those that contribute minimally.

Considering the above points, the input data for data generation are classified into three attributes as follows: material parameters, material state, and loading program, with each attribute further divided into specific features as shown in Figure 5. Material parameters include information about the material properties, which are typically obtained from experimental tests. The attribute material state provides details about the current condition of the material after being subjected to previous fatigue cycles. Finally, the loading program attribute contains information about the fatigue loading program to be applied to the material. Each loading program includes a value for the stress ratio, maximum stress, and number of cycles.

If there are multiple loading cycles with different stress ratios and maximum stresses, each set of loading parameters is treated as a separate input data instance. Once the input data are classified, the next important step is to label each input with its corresponding output. It is possible to have a single output or multiple outputs for the same input. In the context of fatigue, the outputs are categorized into the following two types: primary output and secondary output. The primary output is the fracture energy of the material after undergoing fatigue cycles, which directly relates to its fatigue life. In contrast, the secondary output corresponds to the same attribute (material state) used in the input data, but updated with new feature values for the next loading program.

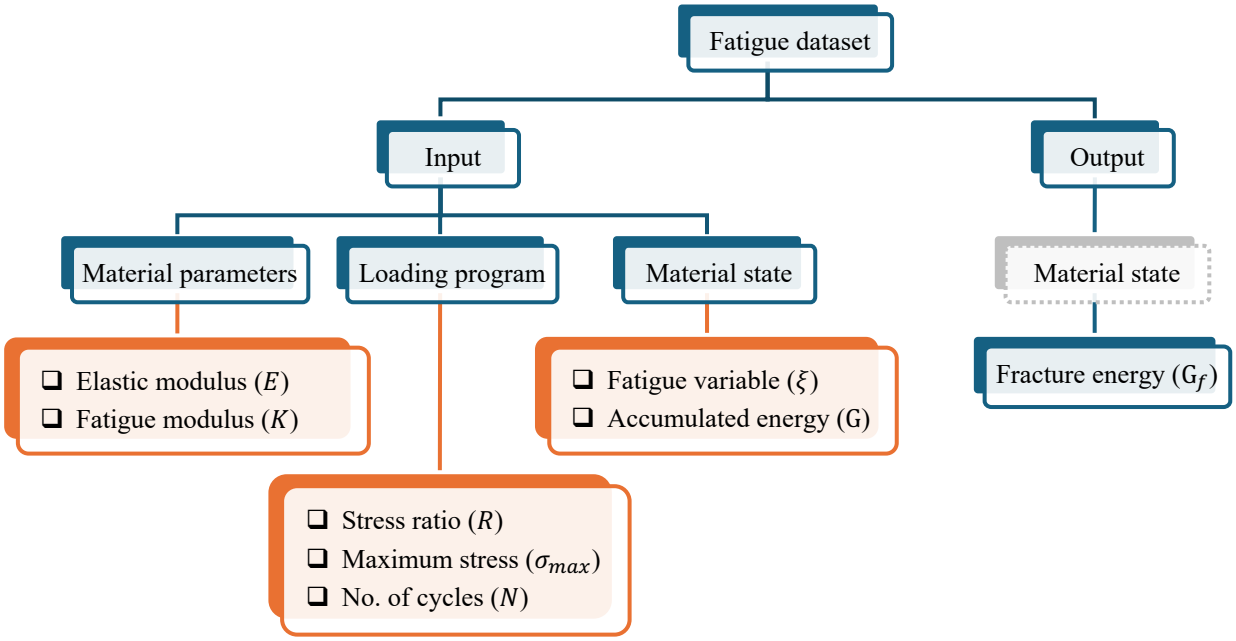


Figure 5. Classification of fatigue life prediction dataset.

After establishing the dataset, the next step is to generate the data. This is achieved by running numerical simulations using the energy-based fatigue model discussed in the previous section. A total of 1962 data instances are generated by varying the features of the input data within a specified range, as shown in Table 1, where K_0^i, β_σ, β_k, and σ_c are the parameters used to define the evolution of the fatigue modulus. A detailed discussion of these parameters is provided in Section 4. The entire dataset is then split into training and testing subsets, with 80% allocated for training and 20% for testing. Additional data with out-of-range inputs are also generated to evaluate the model's capability to predict fatigue life through extrapolation.

Table 1. Range of features for the fatigue life prediction dataset.

Features	Range	Unit
Elastic modulus (E)	[20, 30, 40]	GPa
Inital fatigue modulus (K_0^i)	[100, 150, 200]	MPa
Fatigue decay rate (β_σ)	[0.5, 1.0, 1.5]	-
Fatigue decay rate (β_k)	[0.3, 0.5, 0.7]	-
Threshold stress (σ_c)	[25, 30, 35]	MPa
Stress ratio (R)	[0, 0.3, 0.6]	-
Maximum stress (σ_{max})	[20, 30, 40, 50]	MPa
Number of cycles (N)	-	-
Fatigue variable (ξ)	-	-
Accumulated energy (G)	-	MPa

A total of 1962 data instances are generated.

3.2. Learning Algorithms

A learning algorithm in machine learning is a mathematical procedure that is used to learn a function or a model from a dataset. The goal of a learning algorithm is to discover patterns or relationships in the data that can be used to make predictions or decisions about new, unseen data. A learning algorithm typically takes a set of input data (also known as the training data) and produces a model that can be used to make predictions on new data. The process of learning involves iteratively adjusting the parameters of the model to minimize the difference between the predicted output and the actual output in the training data. Two different ML algorithms are used in the present study, which are *k*-Nearest Neighbors (KNN) and Deep Neural Network (DNN) algorithms. A simple framework for both algorithms is shown in Figure 6, whereas a detailed discussion of the algorithms is provided in the following subsections.

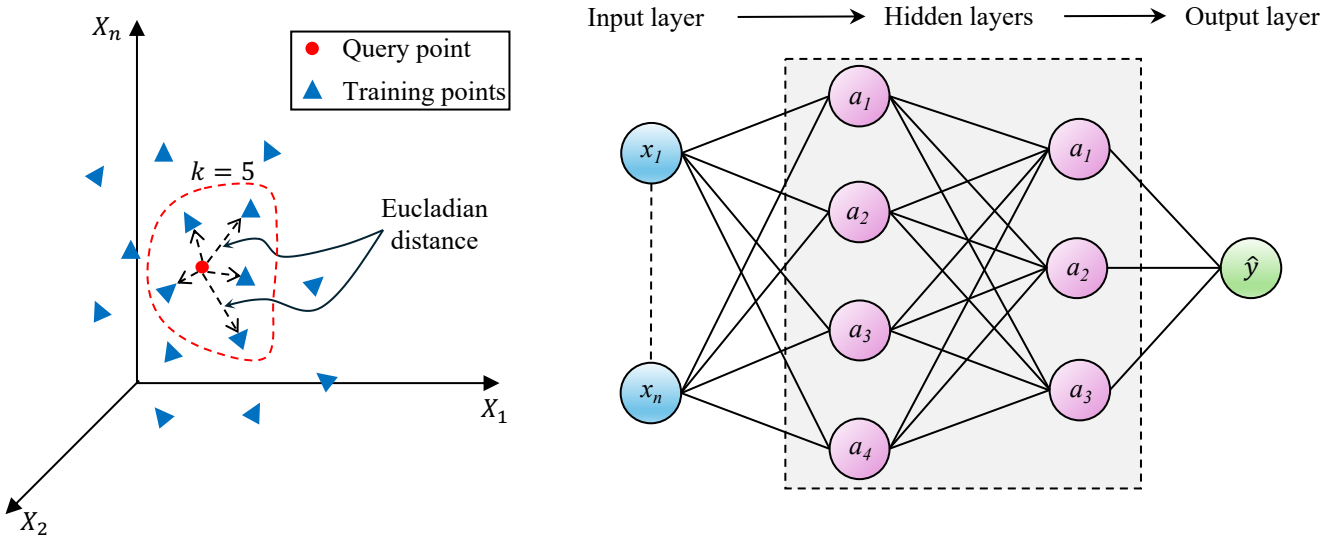


Figure 6. (**a**) KNN regression model ($k = 5$). (**b**) Three layers deep learning model.

3.2.1. KNN Learning Algorithm

The KNN algorithm is the simplest and most basic regression algorithm for situations where there is minimal or no prior understanding of the data distribution [45]. This approach involves keeping the complete training set during the learning phase and determining the value for each query by considering the average or weighted average of its *k*-nearest neighbors values within the training set. It can be helpful to weigh the contributions of the neighbors, so that the neighbors which are nearer to the query can contribute more to the average than the neighbors which are further away from the query. Providing a query point and a training set, the distance of the query point from all the samples in the training set can be calculated. The *k* distances which have the smallest value correspond to the training samples which are nearest to the query point. Hence, the value of the query point can be calculated based on the average or weighted average of these *k* sample training values.

As discussed earlier, the KNN algorithm makes predictions depending on the result of the *k* neighbors which are closest to a given query point. Therefore, to make predictions with the KNN algorithm, first, the distance between all the samples present in the training dataset and the unknown sample (query point) need to be calculated. One of the most

popular methods to measure this distance is known as Euclidean. Other methods include Euclidean squared, City-block, and Chebychev, as shown in the following equation

$$D(x,p) = \begin{cases} \sqrt{(x-p)^2}, & \text{Euclidean} \\ (x-p)^2, & \text{Euclidean squared} \\ |x-p|, & \text{City-block} \\ \max(|x-p|), & \text{Chebychev} \end{cases} \tag{7}$$

where x and p are the query point and the training sample, respectively.

After the selection of the value of hyperparameter k, predictions can be made based on the examples with which the model is trained. For regression, KNN prediction is the average of the k-nearest neighbors outcome values. The prediction can be calculated using the following equation

$$\hat{y} = \frac{1}{k}\sum_{i=1}^{k} y_i \tag{8}$$

where $\hat{y}$ is the predicted value, k is the number of neighbors, and y_i is the value of the ith nearest neighbor in the training set. As KNN predictions rely on the intuitive notion that objects in close proximity may be similar, it is logical to give more weight to the closest points among the k-nearest neighbors that would have a greater influence on determining the outcome of the query point. To accomplish this, a set of weights, denoted as W, can be introduced, with each weight corresponding to a nearest neighbor. These weights are determined based on the relative proximity of each neighbor to the query point. Thus,

$$\begin{aligned} W(x,p_i) &= \frac{e^{-D(x,p_i)}}{\sum_{i=1}^{k} e^{-D(x,p_i)}} \\ \hat{y} &= \sum_{i=1}^{k} W(x,p_i) y_i \end{aligned} \tag{9}$$

where $D(x,p_i)$ is the distance between the query point and the ith nearest neighbor.

3.2.2. DNN Learning Algorithm

The DNN algorithm is one of the most popular ML algorithms developed so far, inspired by the functioning of neurons in the human brain. It has been proven exceptionally successful in various fields [46,47]. As shown in Figure 6b, a typical DNN model consists of multiple layers of neurons, where the neurons in one layer are interconnected with those in neighboring layers. The model architecture starts with the input layer as the first layer, which passes data to the next layer, called the hidden layer. In hidden layers, the inputs are processed in two steps. First, a linear transformation of the inputs is performed, and then, non-linearity is added using a non-linear transformation called the activation function. Finally, the output from the last hidden layer is passed to the output layer. Similar to the hidden layers, two-step input data processing is performed in the output layer to produce the predictions. A DNN model is represented by the number of layers in the model, which is the sum of the hidden layers and the output layer. The output layer is included because it also applies a transformation to the data before producing the final output.

Developing a DNN model starts with the selection of the number of layers, referred to as the depth of the model, and the number of neurons in each layer, referred to as the width of each layer. These are the hyperparameters of the model that need to be assumed before

training. After selecting these hyperparameters, the solution process progresses from one layer to the next. In each layer, a linear transformation is performed first, as shown below

$$\mathbf{z}^{[L]} = \mathbf{W}^{[L]}\mathbf{a}^{[L-1]} + \mathbf{b}^{[L]} \tag{10}$$

where $\mathbf{z}^{[L]}$ is the output of the Lth layer after performing the linear transformation, $\mathbf{W}^{[L]}$ is the weight matrix associated with the Lth layer, $\mathbf{a}^{[L-1]}$ is the output of the $(L-1)$th layer ($\mathbf{a}^{[0]} = \mathbf{x}$), $\mathbf{b}^{[L]}$ is the bias vector associated with the Lth layer, and $\mathbf{x}$ is the input from the training dataset. It is very important that $\mathbf{W}^{[L]}$ and $\mathbf{b}^{[L]}$ are initialized randomly for each layer; otherwise, the model parameters associated with each neuron in a layer will be the same, preventing the network from learning effectively. There are various methods to randomly generate these parameters, with one of the most common being Gaussian initialization, as shown below

$$\mathbf{W}^{[L]} \sim \mathcal{N}\left(0, \sqrt{\frac{2}{n^{[L-1]} + n^{[L]}}}\right) \tag{11}$$

where $n^{[L-1]}$ and $n^{[L]}$ are the number of neurons in the $(L-1)$th and Lth layers, respectively. For biases, they are typically initialized to zero by default. However, in some cases, biases are learned implicitly during the training process. The output after the linear transformation is then used to introduce non-linearity into the model using an activation function. Generally, two different activation functions are used in the model as follows: one for the neurons in the hidden layers and another for the output layer. Different activation functions and their use cases are shown in Table 2. The output from the activation function is then passed to the next layer, and the same process is repeated until the final output is obtained.

Table 2. Common activation functions used in the DNN model.

Activation Function	Formula	Range	Use Case
Linear	$a = z$	$(-\infty, \infty)$	Output layer
Sigmoid	$a = \frac{1}{1+e^{-z}}$	$(0, 1)$	Output layer
Tanh	$a = \frac{e^{z}-e^{-z}}{e^{z}+e^{-z}}$	$(-1, 1)$	Hidden layer
ReLU	$a = \max(0, z)$	$(0, \infty)$	Hidden layer
Leaky ReLU	$a = \begin{cases} z & \text{if } z > 0 \\ \alpha z & \text{if } z \leq 0 \end{cases}$	$(-\infty, \infty)$	Hidden layer

The output from the last layer, representing the predicted value, is then used to formulate the loss function, which measures the error between the predicted value and the real value. For models where the final output is a real-valued number, the most common loss functions are the mean squared error (MSE) and mean absolute error (MAE), as shown below

$$\mathcal{L} = \begin{cases} \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2; & \text{MSE} \\ \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i|; & \text{MAE} \end{cases} \tag{12}$$

where $\hat{y}_i$ and y_i are the prediction and actual values, respectively, for the ith input data instance, and N is the total number of input data instances. After formulating the loss function, the goal of the DNN algorithm is to minimize the loss function, which is achieved

using the gradient descent method. In this method, the model parameters are updated iteratively until the loss function is minimized. The update can be written as

$$\begin{aligned} \mathbf{W}_{new}^{[L]} &= \mathbf{W}_{old}^{[L]} - \alpha \frac{\partial \mathcal{L}}{\partial \mathbf{W}^{[L]}}\Big|_{old} \\ \mathbf{b}_{new}^{[L]} &= \mathbf{b}_{old}^{[L]} - \alpha \frac{\partial \mathcal{L}}{\partial \mathbf{b}^{[L]}}\Big|_{old} \end{aligned} \tag{13}$$

where α represents the learning rate. The computational graph of a two-layer DNN model is illustrated in Figure 7.

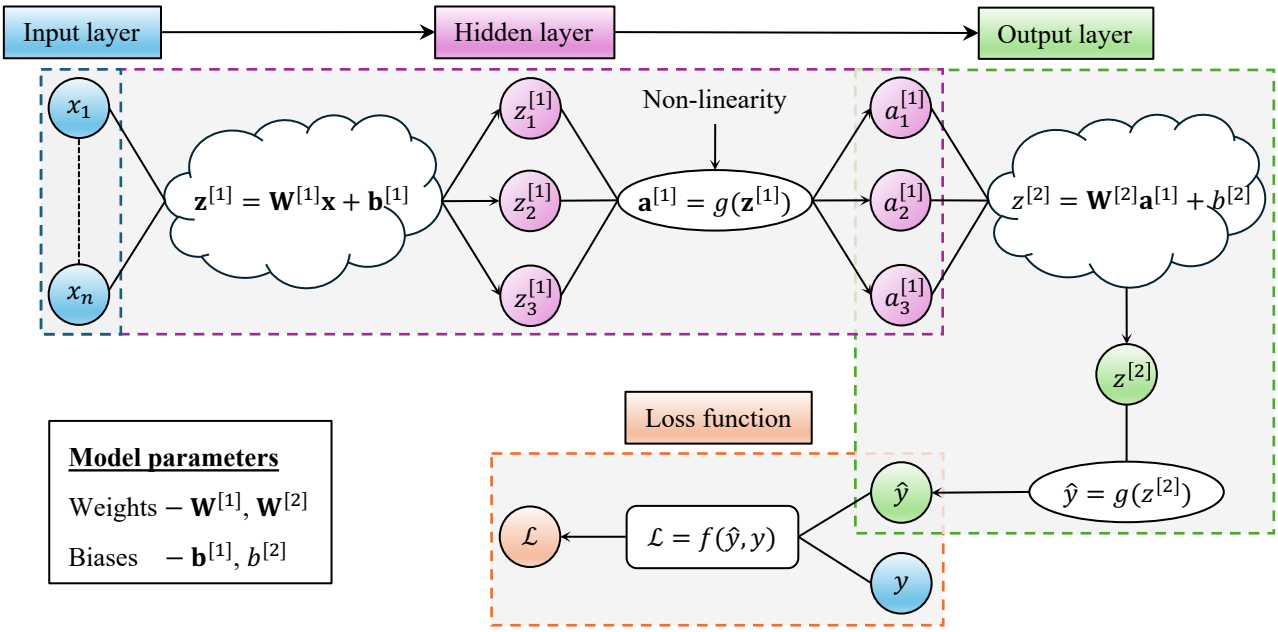


Figure 7. Computational graph view of a two-layer deep learning model.

3.3. Model Evaluation

Model evaluation is the final step of data-driven modeling, where the accuracy of the model on unseen data is assessed. The goal of an ML model is to achieve a low generalization error $\mathcal{E}_g$, which is the expected error between the true value and the predicted value. Mathematically, $\mathcal{E}_g$ can be represented as

$$\mathcal{E}_g = \mathbb{E}\big[\hat{f}_n(x) - y\big]^2 \tag{14}$$

where y is the true value, and $\hat{f}_n(x)$ is the predicted value from the model trained on the nth input dataset. It can be further decomposed into irreducible error, bias, and variance as

$$\mathcal{E}_g = \sigma^2 + \underbrace{\mathbb{E}\big[\hat{f}_n(x) - f(x)\big]^2}_{\text{bias}^2} + \underbrace{\mathbb{E}\big[\hat{f}_n(x) - \mathbb{E}[\hat{f}_n(x)]\big]^2}_{\text{Variance}} \tag{15}$$

where σ^2 is the irreducible error and $f(x)$ is the true function. The bias of a model represents the difference between the true function value and the predicted function value. This arises due to an "expressivity handicap", meaning the model is too simple to represent the true function. Whereas variance results from training on a finite sample of the dataset, causing the model to be sensitive to the specific dataset used for training. In practice, calculating $\mathcal{E}_g$ is challenging because the true function is unknown, and also, the dataset is limited, making it infeasible to train the model many times on different datasets.

In instances where the actual value of $\mathcal{E}_g$ cannot be calculated directly, $\mathcal{E}_g$ can be estimated based on the model's performance on the training and testing datasets. A heuristic estimate of the bias can be derived from the training error, while the variance can be approximated by subtracting the training error from the validation error. Once the bias and variance are estimated, they can be reduced by taking appropriate actions in the next iteration of model building.

The following actions can be taken to address high bias:

- Increase the complexity of the model.
- Reduce regularization.

Whereas to address high variance, the following can be applied:

- Collect more data for training.
- Use a simpler model.
- Increase regularization.

4. Numerical Examples

This section presents numerical examples that demonstrate the proposed energy-based fatigue model and the data-driven fatigue prediction model. The first example provides the numerical validation of the energy-based fatigue model against experimental data, focusing on the stress–strain curve obtained from cyclic loading on concrete. The second example presents results for predicting the fatigue life of a structure through numerical simulations and validates these results with experimental data. Finally, the last example highlights the capability of the machine learning model to predict the fatigue life of a structure based on data generated from the energy-based fatigue model.

Each numerical example is performed on a concrete specimen modeled as a two-noded 2D Timoshenko beam element, incorporating material non-linearity due to plasticity and damage. The specimen is fixed at one end and subjected to compressive axial loading at the other end. Numerical simulations are conducted using the Finite Element Analysis Program (FEAP), developed by Prof. R.L. Taylor, for solving static and dynamic FEM problems [48]. The loading is applied in increments of 0.1 MPa, and the simulations are stopped when the model fails to converge for a particular loading cycle while attempting to achieve the desired maximum value of stress range.

4.1. Validation of the Energy-Based Fatigue Model

In this example, the capabilities of the fatigue model are demonstrated in simulating the experimental behavior of concrete material subjected to compressive cyclic loading. The concrete model is implemented by combining the fatigue model with the kinematic plastic hardening model. The material has an initial Young's modulus (E) of 33.5 GPa, initial fatigue modulus (K_0) of 0.37 MPa, and yield stress limits of 10 MPa and 20 MPa, with kinematic hardening moduli of 250 GPa and 110 GPa, respectively. The decay of the fatigue modulus (K) is governed by the parameter β_k, which is set to 0.37. The evolution equation for K is given as

$$K = K_0 exp(-\beta_k \xi) \tag{16}$$

The loading program applied to the concrete bar is shown in Figure 8a, where eight cycles with varying stress amplitudes are imposed. The comparison between the experimental data [49,50] and the numerical results for the selected material parameters is presented in Figure 8b. The numerical results demonstrate that the proposed model can capture the behavior of the concrete material observed in the experiment with very good accuracy.

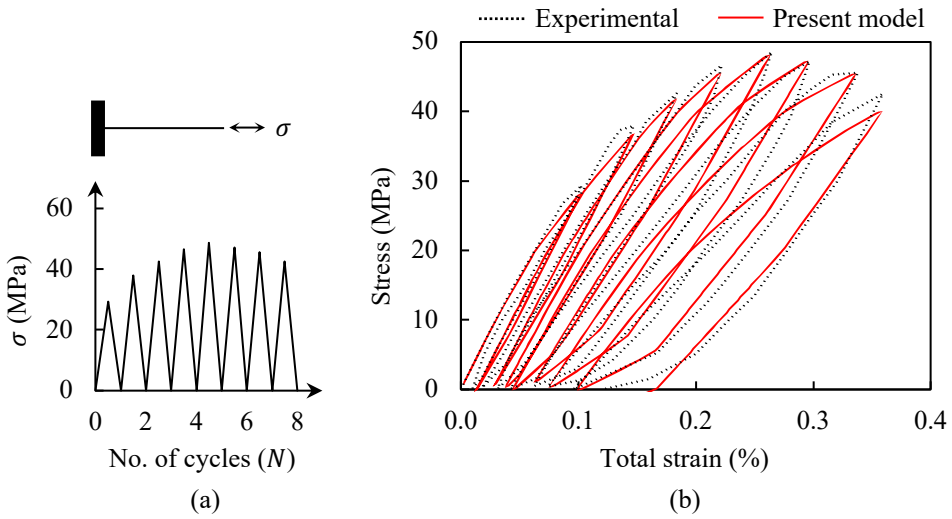


Figure 8. (**a**) Loading program for concrete bar. (**b**) Comparison between experimental and numerical results.

Furthermore, a parametric study is performed to demonstrate the accuracy of the model. The dissipation and plastic strain obtained from the model are compared with the experimental data, with the results presented in Figure 9a and Figure 9b, respectively. Figure 9c illustrates the evolution of ξ as a function of the number of cycles, showing that the evolution occurs only during the loading phase, with no evolution during the unloading phase. Additionally, Figure 9d demonstrates the reduction in fracture energy and the accumulation of energy as a function of the fatigue variable, where after eight cycles, the fracture energy is reduced to approximately 55% of its initial value.

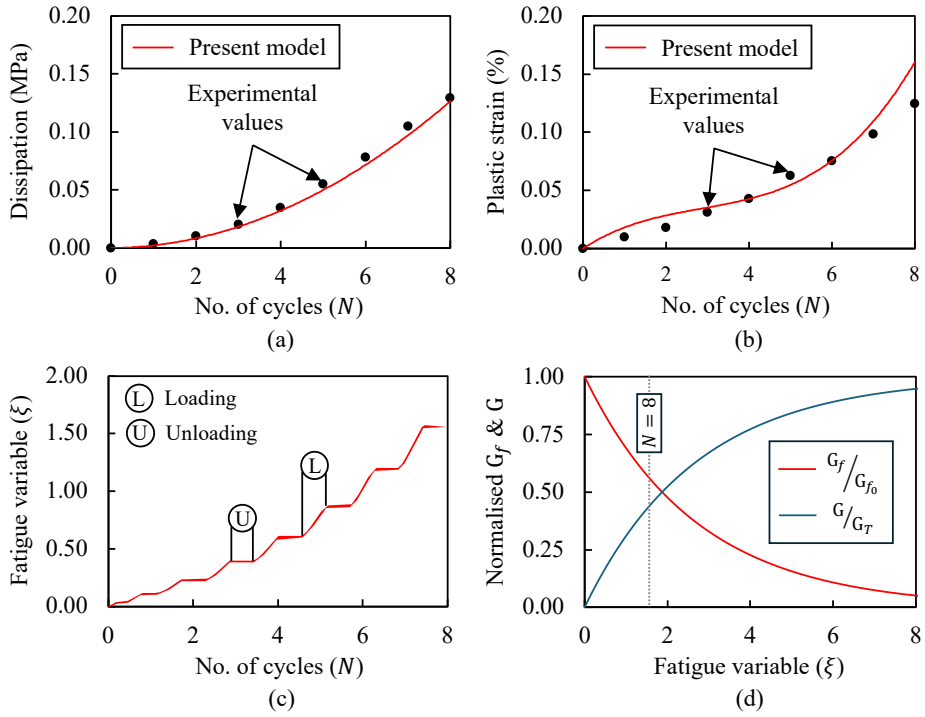


Figure 9. Comparison of the (**a**) dissipation and (**b**) plastic strain between experimental and numerical results. (**c**) Evolution of ξ as a function of number of cycles. (**d**) Reduction in fracture energy and accumulation of energy as a function of the fatigue variable.

4.2. Fatigue Life Prediction Using the Energy-Based Model

For predicting fatigue life, the model is employed to simulate the behavior of concrete subjected to cyclic compressive loading to obtain the S-N curve. The model used in this example includes only the fatigue model, without the kinematic hardening model. The concrete has an initial Young's modulus of 33.5 GPa, while the initial fatigue modulus is not constant but follows a sigmoid decay function of stress. Initially, at low stress levels, the decay is very slow, followed by rapid decay at higher stress levels, as illustrated in Figure 10. The evolution equation for K_0 is given as

$$K_0 = K_0^i \frac{1}{1 + exp\{\beta_\sigma(\sigma - \sigma_c)\}} \tag{17}$$

where K_0^i, β_σ, and σ_c are the parameters of the decaying function, chosen as 27 MPa, 0.88, and 22 MPa, respectively. This function ensures that at low stress levels, a large number of cycles are required to cause significant damage to the material, whereas at high stress levels, even a few cycles can result in substantial damage to the material. Similarly to the previous example, an exponential decay of the modulus due to the evolution of ξ is also considered, with β_k being equal to 0.3.

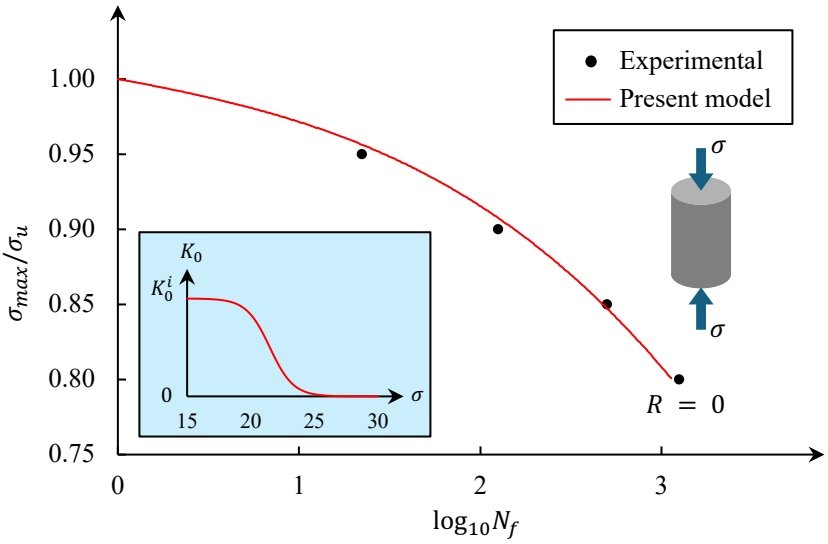


Figure 10. S-N curve obtained from energy-based model.

Using these chosen parameters, numerical simulations are conducted for varying stress amplitudes. The results are presented in Figure 10, where σ_{max} represents the maximum stress value, and σ_u is the ultimate stress value, taken as 30 MPa. The S-N curve obtained from the model is compared with experimental results from the literature [51] and shows good agreement with the experimental data.

Additionally, the effect of the loading sequence on failure cycles is studied by applying low-stress cycles first, followed by high-stress cycles, and vice versa. A graph is then plotted showing the relationship between the cycle ratio for high-stress cycles (η^H) and the cycle ratio for low-stress cycles (η^L). As shown in Figure 11, the loading sequence significantly affects the number of cycles to failure. When the material is subjected to high-stress cycles first, the number of low-stress cycles required for failure is significantly reduced, and vice versa.

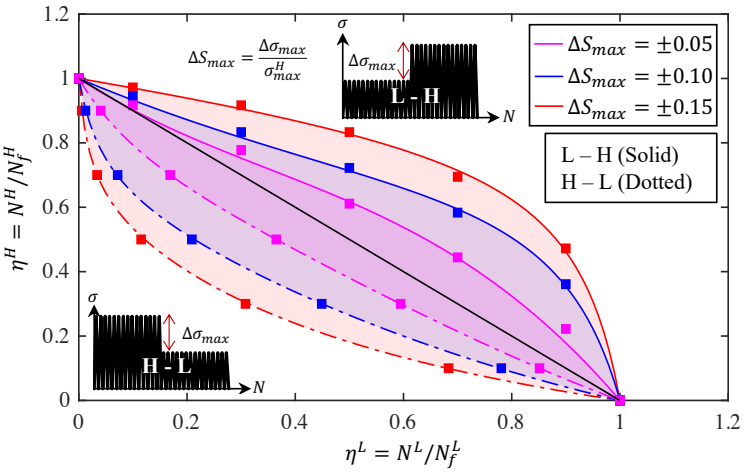


Figure 11. Effect of loading sequence on the number of cycles to failure.

4.3. Fatigue Life Prediction Using the Machine Learning Model

In this example, a total of 1962 simulations were performed using FEAP software (version 8.6) to generate a dataset for training and testing of ML models. The various parameters and their respective ranges are presented in Table 1. The probability density function (PDF) and cumulative distribution function (CDF) of the entire dataset for failure cycles at different values of σ_{max} are shown in Figure 12. The total simulated data are then split into 80% for training and 20% for testing.

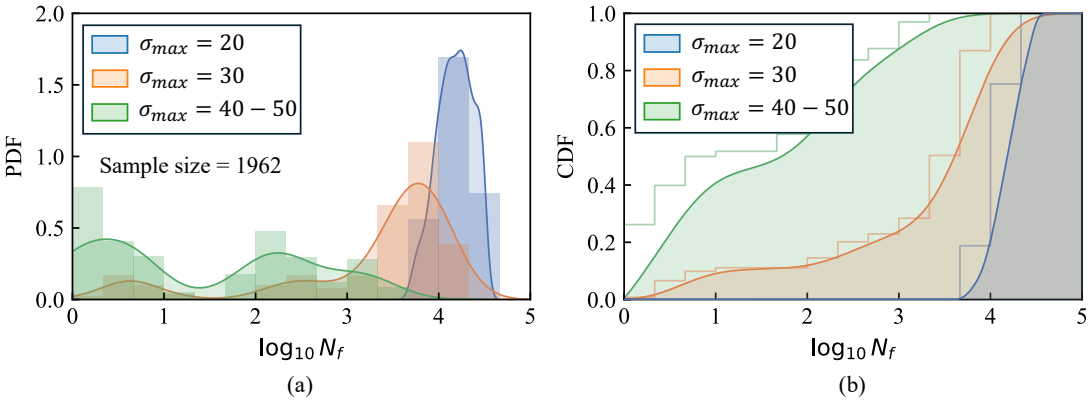


Figure 12. (**a**) Probability density function (PDF) and (**b**) cumulative distribution function (CDF) of the dataset.

After splitting the data, the training dataset is used to fit various ML models, as listed in Table 3, along with the associated error obtained from each model. In Model 1, the KNN algorithm is used with $k = 5$, employing the Euclidean distance metric and exponential weighting. Meanwhile, in Models 2, 3, and 4, the DNN algorithm is used with 1 (simple model), 2 (moderate model), and 3 (complex model) hidden layers, respectively. Each model utilizes the Leaky ReLU activation function in the hidden layers with $\alpha = 0.01$, while the output layer uses a linear activation function, with no non-linearity applied.

Once the hyperparameters for the different models are selected, the models are trained on the training dataset and then used to make predictions on the testing input dataset. The comparison between the results obtained from the energy-based fatigue model and those from the ML models is presented in Figure 13, along with the corresponding MSE and R^2 values. The R^2 value, also called the coefficient of determination, is calculated as

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2} \tag{18}$$

where $\bar{y}$ is the mean value of the actual output dataset. Model 1 performs well, as expected, because the input features of the testing examples belong to the same range as those of the training examples. Whereas the error obtained in Models 2, 3, and 4, which use the DNN algorithm, decreases exponentially as the number of hidden layers (and thus model complexity) increases in the network. To further evaluate the performance of the models, the S-N curve is obtained for the material with $E = 33.5$ GPa, $K_0^i = 27$ MPa, $\beta_\sigma = 0.88$, $\beta_k = 0.3$, and $\sigma_c = 22$ MPa. The values of the fatigue parameters K_0^i and σ_c are purposely chosen to be out-of-range from the typical values used to generate the training dataset for the ML models. Figure 14 compares the predictions of different models with the experimental data [51], showing that even though the query points are outside the range of the training dataset, Model 4, with three hidden layers, still provides accurate predictions, with only 0.6% MSE compared to original values.

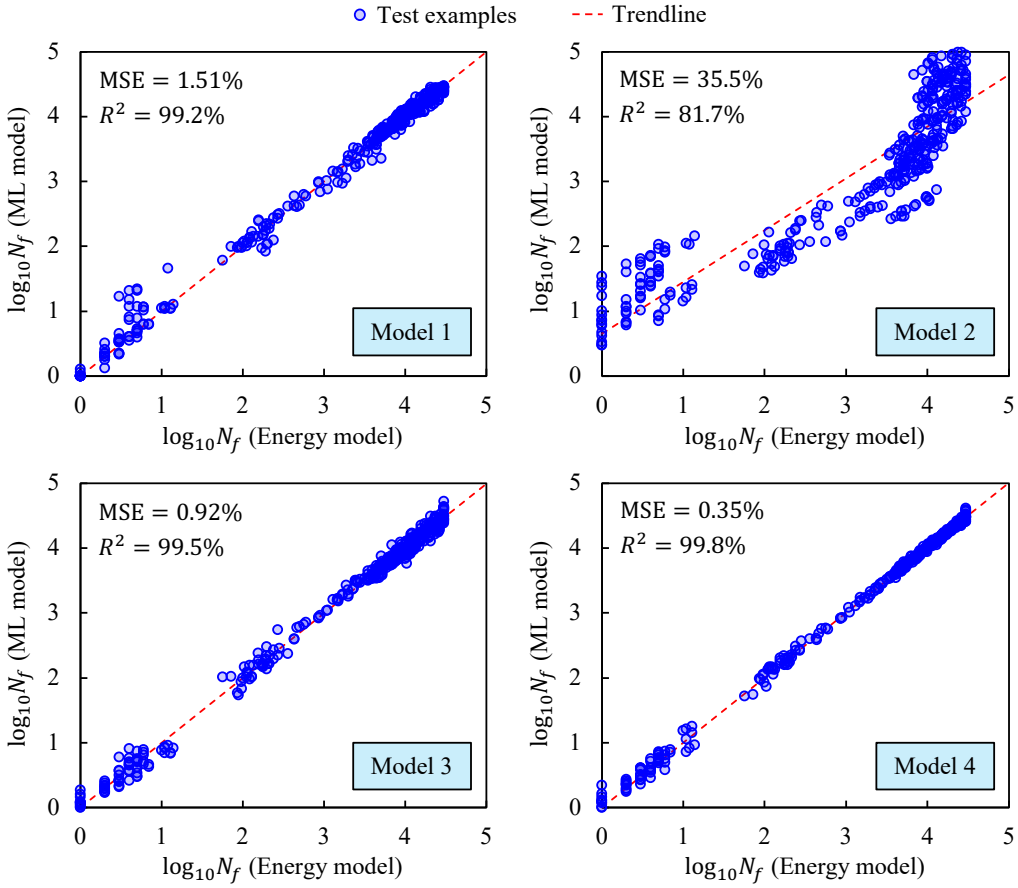


Figure 13. Comparison of fatigue life predictions from various ML models with those obtained from the energy-based fatigue model.

Table 3. Measured error for different models.

Name	Algorithm	Parameter	MSE (%)
Model 1	KNN	$k = 5$	1.51
Model 2	DNN	$H = [32]$ *	35.5
Model 3	DNN	$H = [64, 32]$ *	0.92
Model 4	DNN	$H = [128, 64, 32]$ *	0.35

* No. of neurons in each hidden layer.

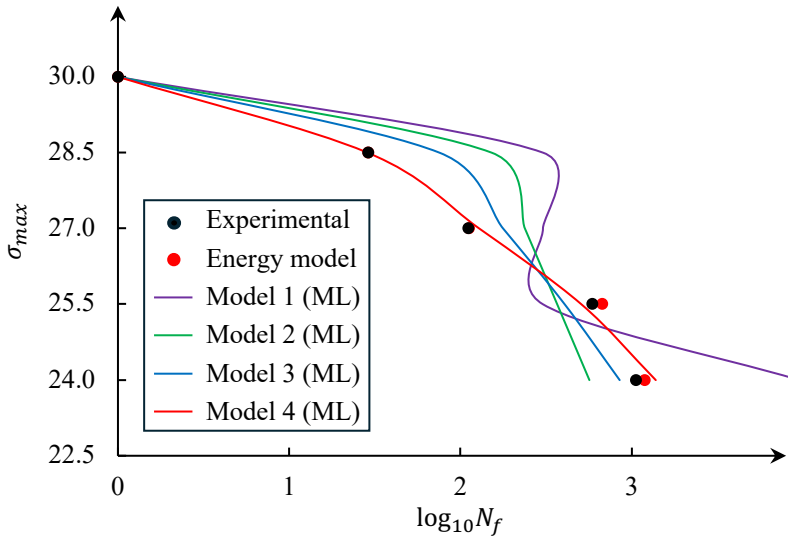


Figure 14. Comparison of S-N curves obtained for out-of-range inputs using various ML models against the energy-based fatigue model and experimental values.

5. Discussion

The present study aims to propose a fundamentally new hybrid physics-informed and data-driven approach for predicting the fatigue life of concrete material. Fatigue issues are always one of the leading causes of structural failure in many concrete structures. Various researchers have attempted to develop advanced numerical models to accurately simulate material behavior under fatigue loading, but these models are computationally intensive and time-consuming. So, researchers shifted to another approach for predicting fatigue life using data-driven ML models. These models are trained on a known dataset, and once trained, they can predict the fatigue life of an unknown material almost instantly. However, in such approaches, the reliability of the obtained results is questionable. One way to address this issue is by using high-quality data generated from advanced numerical models, supported by experimental validations, and carefully selecting input features that have a meaningful impact on the output. Keeping these points in mind, the proposed hybrid approach generates data for the ML models using a novel energy-based fatigue model validated through experiments, and it then uses these data to train ML models. The numerical examples of concrete using the energy-based model and hybrid models demonstrated the capability of these models to simulate the real behavior of concrete material. Additionally, as shown in Figure 14, the present model generalizes the relation between input and output space so well that predictions for out-of-range data inputs are also highly accurate compared to the actual results.

Although the present approach is used to simulate material behavior for a laboratory-sized specimen, it is still part of the broader research objective to integrate FEM models for large structures, such as wind turbine towers and bridge beams, into the proposed hybrid models. Generally, in FEM models for large structures, most of the computational time is spent on communicating information from the element level to the global level. Obtaining this information for complex numerical models requires time-consuming iterative calculations at the finite element level. To overcome this limitation of purely physics-based models, the future scope of this study is to replace the typical computationally expensive element-level communication with information transmitted from trained ML-based finite elements. Moreover, in the present study, temperature effects are ignored. However, they become significant when the heat generated during dissipation plays an important role in enhancing damage and plastic behavior. Thus, the future scope of the study includes

expanding the computational framework to incorporate thermomechanical coupling, as shown in [44,52].

6. Conclusions

In this study, a hybrid physics-informed and data-driven approach is employed to predict the fatigue life of concrete structures, using a novel energy-based fatigue model for data generation with numerical validation to train data-driven ML models. The following key points can be drawn from the present study:

1. A novel energy-based fatigue model is proposed, driven by the accumulation of energy rather than stress in the material, which is the typical approach in most previous studies. Such a criterion can trigger damage and plasticity in the material from low to high stress levels, with the effect growing over time as energy continues to accumulate with each loading cycle.
2. The energy-based fatigue model is validated using two experimental datasets as follows: one for the stress–strain curve of concrete under low-cycle compressive fatigue loading, and the other for obtaining the S-N curve of concrete under compression across different stress ranges.
3. High-quality data are generated using the proposed physics-based model, which are then used to train various ML models, employing algorithms like KNN and DNN. Additionally, the DNN model, with three hidden layers consisting of 128, 64, and 32 neurons, outperforms the others by providing the most accurate predictions for both in-range and out-of-range query points.
4. The present study provides a more economical approach to simulating material behavior in a laboratory-sized concrete specimen, which can be used as ML-trained finite elements in FEM models of large structures, such as wind turbine towers and bridge beams.

Author Contributions: Conceptualization, H.R. and A.I.; formal analysis, H.R.; writing—original draft preparation, H.R.; writing—review and editing, A.I.; supervision, A.I. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the French Ministry of Higher Education and Research (MESR) and Agence Nationale de la Recherche (ANR) in France under grant number ANR-R115RRO/PAYPAFETA.

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ML	Machine Learning
KNN	k-Nearest Neighbors
DNN	Deep Neural Networks
ED-FEM	Embedded Discontinuity Finite Element Method
MSE	Mean Squared Error
MAE	Mean Absolute Error
FEAP	Finite Element Analysis Program
PDF	Probability Density Function
CDF	Cumulative Distribution Function

References

1. Chudnovsky, A.; Sehanobish, K. *Long-Term Strength of Materials: Reliability Assessment and Lifetime Prediction of Engineering Structures*; CRC Press: Boca Raton, FL, USA, 2008.
2. Suresh, S. *Fatigue of Materials*, 2nd ed.; Cambridge University Press: Cambridge, UK, 1998; pp. 1–7.
3. Stephens, R.I.; Fatemi, A.; Stephens, R.R.; Fuchs, H.O. *Metal Fatigue in Engineering*; John Wiley & Sons: Hoboken, NJ, USA, 2000.
4. Albert, W.A.J. Uber Treibseile am Harz. Archiv fur Mineralogie, Georgnosie. *Bergbau Huttenkd.* **1837**, *10*, 215.
5. Wöhler, A. Versucheüber die festiykeit eisenbahnwagenuchsen. *Z. Bauwes.* **1860**, *10*, 8.
6. Palmgren, A. The service life of ball bearings. *Z. Vereines Dtsch. Ingenieure* **1924**, *68*, 339–341.
7. Miner, M.A. Cumulative damage in fatigue. *Am. Soc. Mech. Eng.* **1945**, *12*, A159–A164. [CrossRef]
8. PC, P. A rational analytic theory of fatigue. *Trends Engin* **1961**, *13*, 9–14.
9. Paris, P.; Erdogan, F. A critical analysis of crack propagation laws. *J. Basic Eng.* **1963**, *85*, 528–533. [CrossRef]
10. British Standards Institution. *Code of Practice for Fatigue Design and Assessment of Steel Structures*; British Standards Institution: London, UK, 1995.
11. *EN 1993-1-9*; Eurocode 3: Design of Steel Structures—Part 1–9: Fatigue. European Committee for Standardization: Brussels, Belgium, 2005.
12. Gylltoft, K. A fracture mechanics model for fatigue in concrete. *Mater. Constr.* **1984**, *17*, 55–58. [CrossRef]
13. Bažant, Z.P.; Li, Y.N. Cohesive crack with rate-dependent opening and viscoelasticity: I. mathematical model and scaling. *Int. J. Fract.* **1997**, *86*, 247–265. [CrossRef]
14. Yang, B.; Mall, S.; Ravi-Chandar, K. A cohesive zone model for fatigue crack growth in quasibrittle materials. *Int. J. Solids Struct.* **2001**, *38*, 3927–3944. [CrossRef]
15. Nguyen, O.; Repetto, E.A.; Ortiz, M.; Radovitzky, R.A. A cohesive model of fatigue crack growth. *Int. J. Fract.* **2001**, *110*, 351–369. [CrossRef]
16. Carrara, P.; De Lorenzis, L. A coupled damage-plasticity model for the cyclic behavior of shear-loaded interfaces. *J. Mech. Phys. Solids* **2015**, *85*, 33–53. [CrossRef]
17. Ma, Z.; Le Tallec, P.; Maitournam, H. An energy-based strategy for fatigue analysis in presence of general multi-axial time varying loadings. *Int. J. Fatigue* **2020**, *132*, 105367. [CrossRef]
18. Ibrahimbegovic, A.; Jehel, P.; Davenne, L. Coupled damage-plasticity constitutive model and direct stress interpolation. *Comput. Mech.* **2008**, *42*, 1–11. [CrossRef]
19. Jehel, P.; Davenne, L.; Ibrahimbegovic, A.; Léger, P. Towards robust viscoelastic-plastic-damage material model with different hardenings/softenings capable of representing salient phenomena in seismic loading applications. *Comput. Concr. Int. J.* **2010**, *7*, 365–386. [CrossRef]
20. Alessi, R.; Vidoli, S.; De Lorenzis, L. A phenomenological approach to fatigue with a variational phase-field model: The one-dimensional case. *Eng. Fract. Mech.* **2018**, *190*, 53–73. [CrossRef]
21. Carrara, P.; Ambati, M.; Alessi, R.; De Lorenzis, L. A framework to model the fatigue behavior of brittle materials based on a variational phase-field approach. *Comput. Methods Appl. Mech. Eng.* **2020**, *361*, 112731. [CrossRef]
22. Schröder, J.; Pise, M.; Brands, D.; Gebuhr, G.; Anders, S. Phase-field modeling of fracture in high performance concrete during low-cycle fatigue: Numerical calibration and experimental validation. *Comput. Methods Appl. Mech. Eng.* **2022**, *398*, 115181. [CrossRef]
23. Borden, M.J.; Verhoosel, C.V.; Scott, M.A.; Hughes, T.J.; Landis, C.M. A phase-field description of dynamic brittle fracture. *Comput. Methods Appl. Mech. Eng.* **2012**, *217*, 77–95. [CrossRef]
24. Karma, A.; Kessler, D.A.; Levine, H. Phase-field model of mode III dynamic fracture. *Phys. Rev. Lett.* **2001**, *87*, 045501. [CrossRef]
25. Miehe, C.; Hofacker, M.; Welschinger, F. A phase field model for rate-independent crack propagation: Robust algorithmic implementation based on operator splits. *Comput. Methods Appl. Mech. Eng.* **2010**, *199*, 2765–2778. [CrossRef]
26. Miehe, C.; Welschinger, F.; Hofacker, M. Thermodynamically consistent phase-field models of fracture: Variational principles and multi-field FE implementations. *Int. J. Numer. Methods Eng.* **2010**, *83*, 1273–1311. [CrossRef]
27. Spatschek, R.; Brener, E.; Karma, A. Phase field modeling of crack propagation. *Philos. Mag.* **2011**, *91*, 75–95. [CrossRef]
28. Tufail, S.; Riggs, H.; Tariq, M.; Sarwat, A.I. Advancements and challenges in machine learning: A comprehensive review of models, libraries, applications, and algorithms. *Electronics* **2023**, *12*, 1789. [CrossRef]
29. Noor ali, K.; Hemmati, M.; Miraftabzadeh, S.M.; Mohammadi, Y.; Bayati, N. A Mini Review of the Impacts of Machine Learning on Mobility Electrifications. *Energies* **2024**, *17*, 6069. [CrossRef]
30. Tang, Y.; Wang, Y.; Wu, D.; Chen, M.; Pang, L.; Sun, J.; Feng, W.; Wang, X. Exploring temperature-resilient recycled aggregate concrete with waste rubber: An experimental and multi-objective optimization analysis. *Rev. Adv. Mater. Sci.* **2023**, *62*, 20230347. [CrossRef]
31. Lim, Y.T.; Yi, W.; Wang, H. Application of Machine Learning in Construction Productivity at Activity Level: A Critical Review. *Appl. Sci.* **2024**, *14*, 10605. [CrossRef]

32. Naffer-Chevassier, K.; De Vuyst, F.; Goardou, Y. Enhanced Drag Force Estimation in Automotive Design: A Surrogate Model Leveraging Limited Full-Order Model Drag Data and Comprehensive Physical Field Integration. *Computation* **2024**, *12*, 207. [CrossRef]
33. Vo, V.T.; Noeiaghdam, S.; Sidorov, D.; Dreglea, A.; Wang, L. Solving Nonlinear Energy Supply and Demand System Using Physics-Informed Neural Networks. *Computation* **2025**, *13*, 13. [CrossRef]
34. Bao, H.; Wu, S.; Wu, Z.; Kang, G.; Peng, X.; Withers, P.J. A machine-learning fatigue life prediction approach of additively manufactured metals. *Eng. Fract. Mech.* **2021**, *242*, 107508. [CrossRef]
35. Zhan, Z.; Hu, W.; Meng, Q. Data-driven fatigue life prediction in additive manufactured titanium alloy: A damage mechanics based machine learning framework. *Eng. Fract. Mech.* **2021**, *252*, 107850. [CrossRef]
36. Wang, H.; Li, B.; Gong, J.; Xuan, F.Z. Machine learning-based fatigue life prediction of metal materials: Perspectives of physics-informed and data-driven hybrid methods. *Eng. Fract. Mech.* **2023**, *284*, 109242. [CrossRef]
37. Mirzaei, A.M. Stress, Strain, or Energy? which one is superior predictor of fatigue life in notched Components? A novel Machine Learning-Based framework. *Eng. Fract. Mech.* **2024**, *309*, 110401. [CrossRef]
38. Son, J.; Yang, S. A new approach to machine learning model development for prediction of concrete fatigue life under uniaxial compression. *Appl. Sci.* **2022**, *12*, 9766. [CrossRef]
39. Liang, G.Q.; Chen, X.; Jiang, B.Y.; Jiang, C.S. Fatigue life prediction of concrete under cyclic compression based on gradient boosting regression tree. *Mater. Struct.* **2023**, *56*, 172. [CrossRef]
40. Ibrahimbegovic, A. *Nonlinear Solid Mechanics: Theoretical Formulations and Finite Element Solution Methods*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2009; Volume 160.
41. Ljukovac, S.; Ibrahimbegovic, A.; Mejia-Nava, R.A.; Imamovic, I. Geometrically exact 3D beam theory with embedded strong discontinuities for modeling of localized failure in bending. *Int. J. Solids Struct.* **2024**, *297*, 112850. [CrossRef]
42. Ibrahimbegović, A.; Chorfi, L. Viscoplasticity model at finite deformations with combined isotropic and kinematic hardening. *Comput. Struct.* **2000**, *77*, 509–525. [CrossRef]
43. Kassab, M.P.; Campello, E.D.M.B.; Ibrahimbegovic, A. A geometrically exact thin-walled rod model with warping and stress-resultant-based plasticity obtained with a two-level computational approach. *Comput. Methods Appl. Mech. Eng.* **2025**, *433*, 117497. [CrossRef]
44. Suljević, S.; Ibrahimbegovic, A.; Dolarević, S. Hybrid stress and heat-flux formulation of thermodynamics for long-term simulations in thermo-viscoplasticity. *Int. J. Numer. Methods Eng.* **2024**, *125*, e7436. [CrossRef]
45. Imandoust, S.B.; Bolandraftar, M. Application of k-nearest neighbor (knn) approach for predicting economic events: Theoretical background. *Int. J. Eng. Res. Appl.* **2013**, *3*, 605–610.
46. Li, J.; Cheng, J.H.; Shi, J.Y.; Huang, F. Brief introduction of back propagation (BP) neural network algorithm and its improvement. In *Advances in Computer Science and Information Engineering*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 553–558.
47. Abiodun, O.I.; Jantan, A.; Omolara, A.E.; Dada, K.V.; Mohamed, N.A.; Arshad, H. State-of-the-art in artificial neural network applications: A survey. *Heliyon* **2018**, *4*, e00938. [CrossRef]
48. Taylor, R.L. FEAP-A Finite Element Analysis Program. 2014. Available online: http://projects.ce.berkeley.edu/feap/ (accessed on 7 January 2025).
49. Ramtani, S. Contribution to the Modeling of the Multi-Axial Behavior of Damaged Concrete with Description of the Unilateral Characteristics. Ph.D. Thesis, Paris 6 University, Paris, France, 1990.
50. Mazars, J.; Berthaud, Y.; Ramtani, S. The unilateral behaviour of damaged concrete. *Eng. Fract. Mech.* **1990**, *35*, 629–635. [CrossRef]
51. Awad, M.E.M. *Strength and Deformation Characteristics of Plain Concrete Subjected to High Repeated and Sustained Loads*; University of Illinois at Urbana-Champaign: Champaign, IL, USA, 1971.
52. Suljević, S.; Ibrahimbegovic, A.; Dolarević, S. Regularized variational formulation and 3D discrete approximation with enhanced performance in thermoelasticity and thermo-damage. *Comput. Mech.* 2025, *in press*.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.