DOI: 10.21857/ygjwrce11y

Pregledni članak / Review paper
Prihvaćeno / Accepted: 15. svibnja 2025.

ON ARTIFICIAL NEURAL NETWORKS APPLICATION IN SOLID MECHANICS AS AN ALTERNATIVE TO CONVENTIONAL FINITE ELEMENT MODELLING

Jurica Sorić

Summary

Due to the complexity of engineering problems and the advances in computer performance, a novel computational strategy employing artificial neural networks has recently arisen as an alternative to numerical modelling by conventional finite element application. Neural networks are the core technology in the framework of machine learning, which is a subfield of artificial intelligence, and have been adopted for solving computational mechanics problems, especially in the field of solid mechanics. In the present paper, a short review of the neural networks is given, while the feedforward neural network and the physics-informed neural network are presented and discussed in more detail. In the framework of the physics-informed neural network formulations, both the governing partial differential equations and the energy functional are employed in the loss functions. The feedforward neural network approach is tested by linear elastic analysis, while the efficiency of the physics-informed neural network is demonstrated by modelling of elastoplastic structural responses and two-dimensional crack propagation using phase-field theory. All results are compared by the finite element solutions. It is shown that the neural network algorithms reproduce the finite element results correctly, and that they have an advantage in computational efficiency.

Keywords: artificial neural networks; feedforward neural network; physics-informed neural network; linear elastic analysis; elastoplastic analysis; crack propagation.

1. INTRODUCTION

In recent years, modern structures have become more complex, and the requirements for safety, reliability and structural integrity represent an increasing challenge in the scientific and engineering community. To assess their load-carrying capacity,

advanced numerical methods are unavoidable in the framework of computational solid mechanics. Solid mechanics is a subfield of more general computational mechanics, in which approximate solutions for a variety of partial differential equations should be found, where natural, physical and chemical phenomena are mathematically described. Various numerical methods have been employed therein to solve simultaneous linear equations derived by the discretization of partial differential equations. The most popular is the finite element method (FEM), where the discretization of the continuous domain has been performed into small non-overlapping parts. Each of these small sub-domains is called a finite element, in which the Galerkin weak formulation is established.

Due to the complexity of engineering problems, the size of simultaneous linear equations to be solved has been growing significantly. Therefore, many solution techniques have been developed to tackle large-scale problems, which has been supported by the development of computers. Major progress of digital computers has a large impact on the information science and technology including computational mechanics. Due to advances in computer performance, the machine learning (ML) has recently arisen as a complementary approach for solving computational mechanics problems. In particular, it has been adopted in recent research for computational studies in the field of solid mechanics. The ML enables us to address a variety of complex problems by collecting and processing large amounts of data. Using enough data and appropriate algorithms, it is possible to determine all known physical laws, as well as those that are currently unknown. An alternative numerical modeling based on the ML to the conventional finite element (FE) technology has been considered extensively. The ability of the ML to accelerate numerical simulations has encouraged its application as a surrogate of the FE formulation.

The ML is a subfield of a more general artificial intelligence (AI), which is based on algorithms by means of which machines perform cognitive tasks like a human brain. Different disciplines of the AI and their relationship are presented in Figure 1.

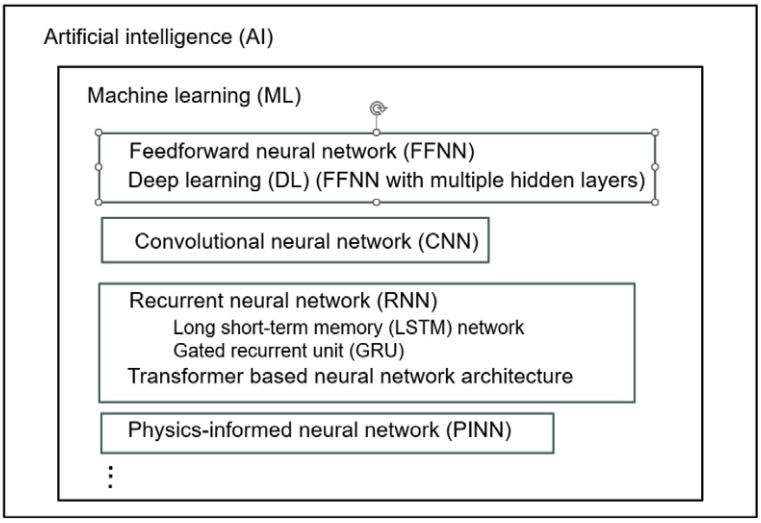


Figure 1. Different AI disciplines
Slika 1. Različite discipline umjetne inteligencije

Depending on the type of data set collected and the problem posed, countless ML methods are available. Therein, the artificial neural network (ANN) is the core technology. The following methods are most common in use: feedforward neural network (FFNN), deep learning (DL), convolutional neural network (CNN), recurrent neural network (RNN), transformer based neural network, and physics-informed neural network (PINN). They are described in more detail in the following text.

As mentioned above, the application of the AI and its disciplines is currently a very active research field in computational mechanics, as shown in Figure 2, where the number of publications concerning the AI and some of its subtopics since 1999 is graphically presented. The diagram was taken from [1].

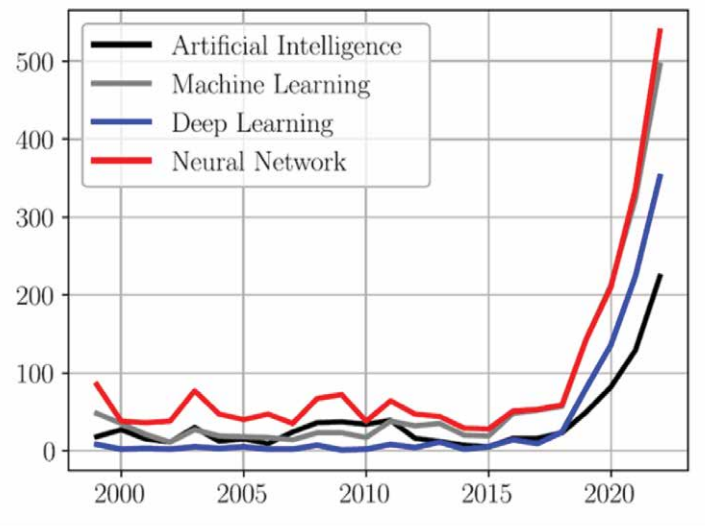


Figure 2. Publications within computational mechanics [1]
Slika 2. Publikacije iz računalne mehanike [1]

Considering the input-output structures, ML approaches are classified into the following categories: supervised, semi-supervised or unsupervised [2]. The learning is supervised if the training data consist of sets of input and associated output values. Therein, the network compares the given output with the predicted one. The goal of the algorithm is to reach the lowest possible error between the given output and the one produced by the network by means of a minimization procedure. If available data are only input values, the ML is unsupervised. The ML is semi-supervised if a large amount of input data, but only a limited amount of corresponding output data, is given. A great challenge in the ML technique is reducing the training error efficiently. The underfitting and overfitting are distinguished. In case the model is not able to obtain a sufficiently low error, the underfitting happens. The overfitting appears when the difference between the training error and the test error is too large [3].

Artificial neural networks (ANNs) are built by the neurons assembled into the network architectures in the form of layers. The neurons are also called processing units [4]. There are input, output, and hidden layers. The neurons of neighbouring layers are connected to each other. This means that each neuron of the current layer is connected to all the neurons in the previous and in the next layer. The signal flows from the neurons of the input layer over the hidden layers to the neurons of the output layer, reproducing

overall network response in the framework of a learning process. There are network parameters such as the connection weight between neurons and the bias that should be adjusted. The neuron outputs are computed by appropriate activation functions, which are also called transfer functions [4, 5, 6].

In the feedforward neural network (FFNN), the signal flows only forward – from the input layer to the output layer through hidden layers, via connections between neurons. The learning process is iterative, in which the network parameters are computed through the update procedure, where the error in the form of loss function is minimized by using the backpropagation based on the gradient descent algorithm [5, 6]. The database consisting of input and target output quantities, created from experimental investigations, is mostly used if available, however the data obtained by computational analyses may also be an option.

The FFNN approach can efficiently predict the stresses and the strains in solid mechanics analyses instead of solving a conventional FE model [7]. The conventional modelling of constitutive laws, especially for materials with complex heterogeneous microstructure, may be replaced by the FFNN [8]. The neural networks are efficiently used for evaluating the local behaviour of the representative volume element (RVE) in the framework of multi-scale numerical algorithms [9]. The modelling of deformation responses of hyperelastic materials by means of the FFNN is presented in [10]. In [11], the FFNN is used to predict the fracture behaviour of particulate composite materials under impact loading. The material fracture modelling based on the phase field approach has been efficiently performed by using the FFNN in [7].

To solve complex nonlinear problems more efficiently, neural networks with multiple hidden layers are needed, and the training is called deep learning (DL) [3, 12]. A comprehensive training database and a large number of computations is required in the learning process; this requires high performance computing and more complex computer architecture [6]. The DL algorithms have achieved great success in various fields such as image recognition, speech recognition, language translation, biotechnology, and medicine sciences, which is coupled with predicting the activities of drug molecules and the discovery of new drugs. A historical survey of relevant works dealing with the DL is presented in [13].

Recently, the application of the DL to solid mechanics modelling has been significantly increased. The DL is used in the homogenization procedure in the framework of material modelling based on the multi-scale algorithms [14]. The study of fracture prediction on the material nano-level by means of the DL approaches is presented in [15]. Deep neural networks are used for damage identification in structural health monitoring

in [16]. A great challenge is the application of DL algorithms to the discovery, design, and development of new materials [17, 18, 19]. An efficient DL model is used for the consideration of fracture propagation and failure prediction in brittle materials [20]. The deep learning neural networks are further applied in the finite element technology. The finite element interpolation functions may be generated, and the optimization of nodal positions may be achieved through the training process, which is equivalent to the adaptivity procedure in a standard finite element method (FEM). The DL has been applied to develop the optimized numerical quadrature to calculate the FEM element matrices in [21]. Better accuracy and efficiency in comparison to the conventional FEM are achieved, especially in capturing the stress concentration in case of a coarse mesh [22, 23]. A historical overview of the DL algorithms development, in particular their applications to the design and behaviour of composite materials, is presented in [24]. Although deep learning has reached a great progress in accuracy and reliability in recent years, more research is still needed to improve its application in complex material systems.

The convolutional neural network (CNN) is a special type of deep neural network, which is usually used in computer vision and applied in fields such as image classification, video recognition, medical image analysis, as well as self-driving cars. Besides the fully connected hidden layers as in the DL networks, the CNN architecture consists of additional hidden layers such as the convolutional and pooling layers. The input data from the input layer feed the convolutional layers. In contrast to the DL networks, where the neurons from two adjacent layers are connected to each other, in the CNN, only small and localized regions of neurons are connected to a neuron in the next layer. In the convolution layers, the mathematical convolution operations are employed instead of the conventional matrix multiplication used in the fully connected layers. The network weights are shared, which contributes to the increase of network efficiency. After the convolutional layers, the data are transferred to the pooling layers, where they are compressed and converted into a single vector by means of a flattering procedure. Thereafter, the data are further transferred to a feedforward network based on the fully connected layers [9, 3, 25, 26]. In addition to general image processing, the CNN may also be used in the materials science to analyse properties of composite materials with the distribution of various constituents [27], as well as microstructural material properties in microstructure images in the framework of multi-scale modelling [9]. The prediction of material properties of heterogeneous materials by means of the CNN is discussed in [28, 29, 30].

The recurrent neural network (RNN) is another class of deep learning network architecture, and it is dealing with processing sequential data such as speech, text, or

other time-series data. While in the standard deep neural networks, the signal flows only forward from input to output (which are independent of each other), in the RNN, the learning process is performed over several time steps [3, 31, 32]. The output of the previous time step is stored so that the stored data are used with the input to calculate the current output. Due to the sequential processing, the backpropagation through time (BPTT) algorithm is employed in the learning process. Therein, the BPTT sums errors that propagate through the entire length of the sequence because the same network parameters are used, which may lead to undesired phenomena such as the vanishing and exploding gradients. To reduce these effects, more sophisticated versions of the RNN architecture such as the long short-term memory (LSTM) network [33] and the gated recurrent unit (GRU) [34] have been proposed.

The RNN methods have been efficiently applied to the history dependent problems in solid mechanics, such as plasticity, as well as damage and fracture. The plasticity constitutive laws of microstructural RVEs of heterogeneous materials have been found by using the GRU architecture in [35]. The RNN based on a GRU unit is used as a surrogate model of micro-scale simulations in the context of computational multi procedure in [36]. A smart constitutive law (SCL) for the homogenization of inelastic microstructure of RVE has been formulated by means of the LSTM in [37]. Therein, the loading histories applied to RVEs have been expressed by strain sequences. The SCL has been implemented into a finite element (FE) solver to model deformation responses of various engineering problems on the macrolevel. The same SCL is applied to all finite element integrations points, which is an advantage in comparison to the standard multi-scale procedure, where the FE simulation of the microstructure is required for each integration point at each load increment as discussed in [37]. A new type of recurrent neural network based on the self-consistent concept has been proposed for the modelling of elastoplastic solids in [38]. An RNN model using a modified LSTM approach has been applied as a surrogate for the micro-model in the framework of multi-scale FE procedure as shown in [39]. The RNN architecture has been efficiently used to predict the fracture evolution in brittle materials in [40]. The brittle fracture has also been modelled by using the LSTM networks as displayed in [41].

As an alternative to recurrent neural networks (RNNs), the transformer network architecture developed in [42] is used. Unlike RNNs that process sequences sequentially, the transformer architecture uses a novel self-attention mechanism that allows parallel processing of sequence. This achieves faster training and inference, as well as better performance on long-range dependences in sequence modelling tasks. The key property of the transformer is that the vectors of different time step flow through their own paths.

There are dependencies between these paths in the self-attention layer. However, the feed-forward layer does not have those dependencies; thus, various paths are executed in parallel.

It is well-known that the transformer NNs have impressive performance not only in Natural Language Processing (NLP) tasks, but also in a wide range of domains, including computer vision, audio and speech processing, healthcare and a variety of other applications [43]. To date, the transformers have seldom been used for modelling a history-dependent material response. The NLP tasks and the history dependent problems in mechanics such as plasticity, as well as damage and fracture, share the same fundamental characteristics. The structural response established in preceding time steps exercise significant influence on the state variables encountered in the current step. This intricate interdependence between sequences of load conditions highlights the inherently temporal nature of transient modelling. Therefore, the transformer architecture may be a promising formulation in the modelling history dependent mechanical problems [44]. The GRU-based RNN and the transformer- based NN architecture in the framework of encoder-decoder formulation have efficiently been applied to the modelling of constitutive relations in solid mechanics problems in [45]. Both architectures demonstrate gut performance in capturing the underlying stress-strain relations in testing, as well as out-of-domain scenarios. It has been shown that the transformer encoder-based model demonstrates a slightly better performance.

It is known that the training neural networks require a comprehensive database, which is not always available for complex scientific problems; the data collection is often very costly and time-consuming. In addition, the governing physical laws are mostly ignored. Employing the known physical laws into the training process can significantly reduce the size of required training data, as well as increase numerical efficiency, which is particularly evident in the case of modelling engineering systems. Therefore, a new network architecture called the physics-informed neural network (PINN) is proposed, where the physics constraints are imposed. The PINN may be considered as a special class of DL, where the physical conditions are enforced by an extended loss function [46, 47]. Like in the standard neural networks, the NN parameters are computed by minimizing the loss function, which now additionally contains the residual of governing equations and their boundary conditions. The PINN has been successfully applied in solid mechanics in [48], where the linear elasticity, as well as the von Mises elastoplasticity, are studied. The elastic-viscoplastic deformation in solids in dependence on the strain-rate and temperature is modelled in [49]. An advanced PINN technique for modelling two-dimensional and three-dimensional

problems in solid mechanics is developed in [50], where the linear and geometrically nonlinear problems are solved.

In addition to the PINN approach dealing with partial differential equations (PDEs), the PINN computational strategy has been proposed, where the loss function is defined by the energy functional, and the loss is minimized according to the principle of minimum potential energy [51]. Therein, the tension, deflection and buckling of an elastic thin square plate are considered. An efficient PINN algorithm is applied for the phase-field modelling of fracture in [52], where the loss function employs the energy functional consisting of elastic strain energy and fracture energy.

The paper has been structured as follows: In Section Two, the feedforward neural network with multiple layers is described in more detail, and the results obtained by computation of linear elastic problem are presented. Section Three deals with the physics-informed neural network formulation using differential equations, and the application to modelling of an elastoplastic problem. In the same section, the energy-based physics-informed neural network using the phase-field approach is applied to crack modelling. Finally, the concluding remarks and possible future work directions are presented in Section Four.

2. FEEDFORWARD NEURAL NETWORK

As mentioned already, the feedforward neural network (FFNN) represents the fundamental network architecture consisting of the input, output and hidden layers, which contain multiple neurons. The signal flows from the neurons of the input layer to each neuron of the first hidden layer and further forward to the neurons of the final output layer through the connections between the neurons. Therein, each neuron receives the input signal from all neurons of the preceding layer. The neurons output is computed by means of the activation function in terms of the input value. Each connection between the neurons is characterized by its weight parameter, and the bias is another parameter associated to the neurons. A multi-layered feedforward neural network with the same number of neurons in each hidden layer is graphically presented in Figure 3. In general, the layers may contain different number of neurons.

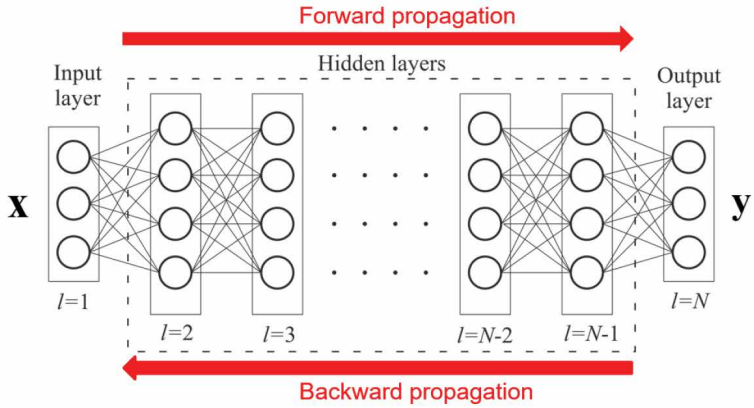


Figure 3. Feedforward neural network with multiple layers
Slika 3. Unaprijedna neuronska mreža s više slojeva

The input values of the neurons in the *l*-th layer are expressed by the vector:

$$\mathbf{u}^l = \mathbf{W}^{l-1}\mathbf{h}^{l-1} + \mathbf{b}^l, \tag{1}$$

where $\mathbf{W}^{l-1}$ is the matrix of the connection weights between the neurons in the $(l-1)$ -th layer and in the *l*-th layer. The vector $\mathbf{h}^{l-1}$ denotes the outputs of the neurons in the $(l-1)$ -th layer, while $\mathbf{b}^l$ comprises the biases of the neurons in the *l*-th layer. The output of the neurons in the *l*-th layer is computed in terms of $\mathbf{u}^l$ by means of an activation function as

$$\mathbf{h}^l = f(\mathbf{u}^l). \tag{2}$$

In the graph in Figure 3, the neurons are illustrated by the nodes (small circles) connected to each other by straight lines. There are three neurons of the input layer, *N*-2 hidden layers of four neurons each, and three output neurons. Accordingly, the input vector of the second layer ($l = 2$) may be written by the following relation:

$$\mathbf{u}^2 = \mathbf{W}^1\mathbf{x} + \mathbf{b}^1, \quad \mathbf{h}^2 = f(\mathbf{u}^2), \tag{3}$$

and the vector of the output layer ($l = N$) is expressed as

$$l = N, \quad \mathbf{y} = \mathbf{W}^{N-1}\mathbf{h}^{N-1} + \mathbf{b}^N. \tag{4}$$

The nonlinear activation functions which are usually employed are the *sigmoid* function and the *hyperbolic tangent* (*tanh*) expressed by

$$f(x) = \frac{1}{1+e^{-x}} \quad \text{and} \quad f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}, \tag{5}$$

respectively, and the linear functions are the rectified linear unit (*ReLU*) written as

$$f(x) = \begin{cases} x\ (x > 0) \\ 0\ (x \le 0) \end{cases} \tag{6}$$

and the identity function $f(x) = x$. Besides, there are several activation functions that may be employed in neural networks, as described in [5]. The functions *ReLU* (*x*), *sigmoid* (*x*) and *tanh* (*x*) are graphically presented in Figure 4.

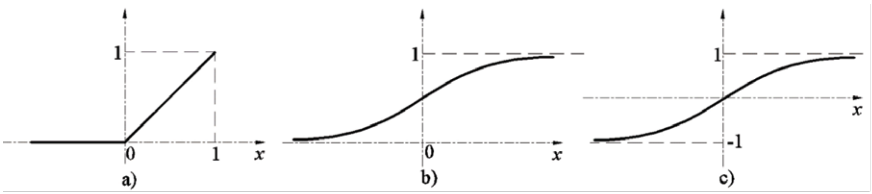


Figure 4 Activation functions: a) *ReLU* (*x*), b) *sigmoid* (*x*), c) *tanh*(*x*)
Slika 4. Aktivacijske funkcije: a) *ReLU* (*x*), b) *sigmoid* (*x*), c) *tanh*(*x*)

According to supervised learning, the training data consist of inputs and target outputs, which may be collected from experimental or computational analyses. A pair of the input and corresponding output data is called a training pattern [6]. The collected data are generally divided into three different sets such as training, validation, and testing. The training set comprises approximately 70% or 80% of the complete data, while the validation and testing contain 15% or 10 % each. The size of the training data used for the learning depends on the complexity of the problems that should be solved, and on the neural network architecture. Both the size of the data set and the complexity of the problem to be solved influence the number of neurons and layers, as well as the type of activation functions.

For simple problems, the networks with two or three hidden layers should be enough, but deep neural networks employing multiple layers are required for complex problems. Unfortunately, there are no uniform rules on how to select the network architecture. If too many neurons are used, the undesired phenomena such as overfitting may appear; this can be prevented by early stopping or by means of regularization procedures. The Bayesian regularization is usually applied [5]. The simplest choice is the size of the output layer, which is the same as the number of the target data. However, the choice of the

input layer may be difficult, especially in the case when the potential number of input data is too large, and some reducing techniques are required. To ensure the efficiency and accuracy of the learning, it is suggested to normalize the input data so that they fall into the unit range of -1 to 1 or 0 to 1. In such a case, undesired major differences between the computed values through the learning process may be avoided. Furthermore, using the normalized data improves the learning process, because the weight parameters and biases do not have large values.

During the learning process, the network parameters such as weight and bias are iteratively computed through the update procedure, where the error expressed by the loss function is minimized. Therein, the back propagation based on the gradient descent algorithm is employed. Using the predicted output data and the corresponding given target data, the loss function as the mean squared error is computed by the following expression:

$$L = \frac{1}{n_s} \sum_{s=1}^{n_s} (\mathbf{y}_s - \hat{\mathbf{y}}_s)^2 \ , \tag{7}$$

where $\mathbf{y}_s$ is the predicted vector of the output layer for the s-th training pattern. The vector $\hat{\mathbf{y}}_s$ comprises the corresponding given target data, and n_s stands for the total number of training patterns.

According to the gradient descent concept, the change of the connection weight between the neurons in the l-th layer and $(l-1)$-th layer is expressed by

$$\begin{aligned} &\Delta \mathbf{W}^{l-1} = \frac{\partial L}{\partial \mathbf{W}^{l-1}} \ , \\ &\mathbf{W}^{l-1}\Big|_{k+1} = \mathbf{W}^{l-1}\Big|_{k} - \alpha \, \Delta \mathbf{W}^{l-1}\Big|_{k} \ . \end{aligned} \tag{8}$$

The change of the bias is performed analogously:

$$\begin{aligned} &\Delta \mathbf{b}^{l} = \frac{\partial L}{\partial \mathbf{b}^{l}} \ , \\ &\mathbf{b}^{l}\Big|_{k+1} = \mathbf{b}^{l}\Big|_{k} - \beta \, \Delta \mathbf{b}^{l}\Big|_{k} \ . \end{aligned} \tag{9}$$

In the above relations, $\big|_k$ denotes iteration k, while α and β are the parameters that define the step size. In the training algorithm, the backpropagation follows after the forward procedure and computing the loss function. There, the connection weights and the biases are updated from the output layer to the input layer by using expressions (8) and (9). This gradient descent process may be considered as an optimization approach, in which the weights and biases converge to their optimal values, for which the mini-

mal error expressed by (7) is reached. The backpropagation algorithm starts after the forward computations are performed for all training patterns. The sum of the forward computations for all training patterns is called the epoch [6]. Accordingly, the error expressed by Eq. (7) gradually decreases epoch by epoch. Before the updating process, the weights and biases should be initialized, and the stopping criteria should be defined [5].

An application of the feedforward neural network (FFNN) has been presented in [7] by solving an elastic boundary value problem of the notched bar under tension, as depicted in Figure 5. The geometric data are L= 1 mm and r = 0.25 mm. The bar is clamped along the left end, and the displacement of u = 0.02 mm is imposed on the right end. The material data are Young's modulus $E = 21$ GPa and Poisson's ratio $\nu = 0.3$. Both the neural network (NN) computation and the finite element (FE) analysis have been performed. The geometric domain of the structure has been discretized by unstructured FE meshes with a total of 306 triangular elements, leading to 189 nodes. The data set is split into training, validation and test subsets. 70% of the data are used for the training, and the same amount of 15 % is set aside for the validation and test.

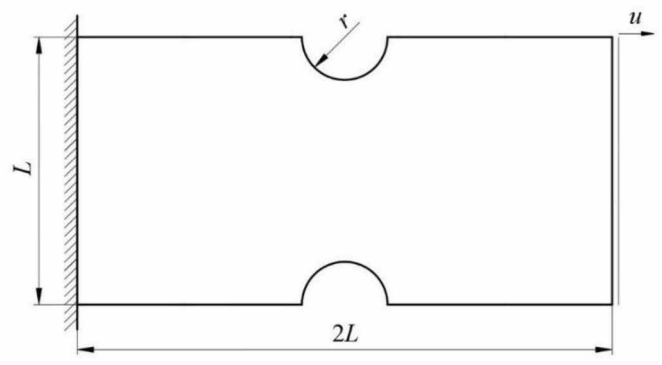


Figure 5. Notched bar under tension
Slika 5. Vlačno opterećen urezani štap

The NN architecture consists of the input layer with three neurons, two hidden layers with ten neurons each, and the output layer of three neurons. The 2D strain and stress components have been chosen as the input and output, respectively. The training of the neural network is performed for 1400 epochs by using the mean squared error as the loss function and the Levenberg-Marquardt algorithm [5] as optimizer.

The stress predicted by the NN approach has been compared with those computed by the FEM, as shown in Figure 6.

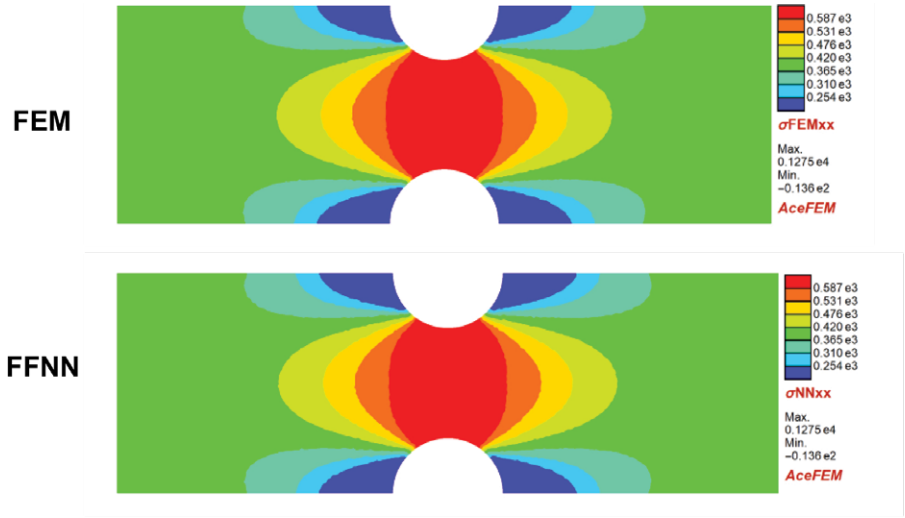


Figure 6. Contour plot of stresses computed by FEM and by NN [7]
Slika 6. Grafički prikaz raspodjele naprezana izračunat metodom konačnih elemenata i neuronskim mrežama [7]

As evident from the contour plot, there are no differences between the results obtained by the NN and FEM. It can be concluded that the presented FFNN formulation works well and predicts the stress distribution accurately. As displayed in [7], it should be noted that NN simulations show a slight superiority in computation time compared to FE solutions, which is expected for problems in elasticity. The CPU Mathematica time is 0.546 s in the FEM, and 0.407 s in the case of the NN analysis. However, the computational effort depends heavily on the machine on which the simulations are running.

3. PHYSICS-INFORMED NEURAL NETWORK

As mentioned in the introductory section, in order to reduce the size of the required training data and, consequently, to increase numerical efficiency, a new network architecture called the physics-informed neural network (PINN) has been proposed. In the multilayered NN presented above, the governing physical laws are mostly ignored. In contrast to traditional deep learning formulations, the PINN can integrate physics information related to the partial differential equations (PDEs) to train neural networks with a relatively small number of training data. Therein, the loss function contains physics-based equations in addition to data-driven knowledge. Beside the PINN approach,

where the loss function is extended with the physics-based PDEs, another PINN formulation with the loss function expressed by the energy functional has also been proposed.

In the following the PINN computational strategy imposing the PDEs governing elastoplastic material response, as well as the PINN employing the energy functional for brittle fracture modelling are considered. The phase-field approach has been applied to modelling of crack evolution.

3.1. PINN application to elastoplastic problem governed by differential equations

A data-driven framework based on the PINNs theory for the solution of an elastoplastic problem is presented. As mentioned above, an improved loss function including additional physics-constrained term is used in the standard FFNN architecture to efficiently incorporate physical information, as shown in Figure7.

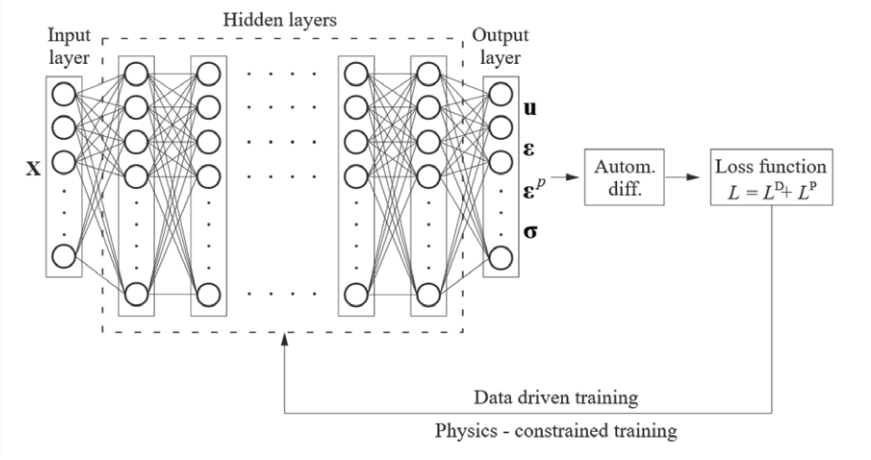


Figure 7. Schematic of PINN architecture for elastoplastic modelling
Slika 7. Shematski prikaz PINN arhitekture za elastoplastično modeliranje

The presented neural network model maps the coordinates **x** from the input layer to the predicted elastoplastic field variables of the output layer using the feedforward propagation. After performing the derivatives by means of the automatic differentiation algorithms [53], the multi-objective loss function consisting of the data-driven loss L^D and the physics-based loss L^P is computed. In the data-driven loss the elastoplastic predicted variables are compared to the target values obtained by the conventional

elastoplastic FE simulation. The physics-based loss consists of the residual of governing PDEs, elastoplastic constitutive relations, flow rules, complementary conditions, and various boundary conditions. Like in the standard FFNN, during the learning procedure, the optimization problem is solved using an optimizer, where the gradients are calculated by means of the backpropagation process.

The elastoplastic formulation presented employs the isotropic hardening von Mises model under the assumption of small strain. For two-dimensional problem, a monotonic and proportional loading is assumed, so that the incremental flow rule could be represented by total form. The basic equations of the elastoplastic analysis are displayed in Table 1. More details can be found in [54].

Table 1. Basic equations of elastoplastic analysis
Tablica 1. Osnovne jednadžbe elastoplastične analize

Equilibrium equation: $\nabla\boldsymbol{\sigma}+\mathbf{f}=\mathbf{0}$ in Ω
Dirichlet boundary condition: $\mathbf{u}=\bar{\mathbf{u}}$ on Γ_u
Neumann boundary condition: $\boldsymbol{\sigma}\,\mathbf{n}=\bar{\mathbf{t}}$ on Γ_t
Constitutive law: $\boldsymbol{\sigma}=\mathbf{C}:(\boldsymbol{\varepsilon}-\boldsymbol{\varepsilon}^p)$
Total strain: $\boldsymbol{\varepsilon}=\frac{1}{2}\left[\nabla\mathbf{u}+\nabla(\mathbf{u})^T\right]$
Plastic strain: $\boldsymbol{\varepsilon}^p=\lambda\frac{\partial F}{\partial\boldsymbol{\sigma}}$, λ - plastic multiplier,
$\lambda=\varepsilon^p_{eq}$, $\boldsymbol{\varepsilon}^p=\frac{3}{2}\varepsilon^p_{eq}\frac{\mathbf{S}}{\sigma_{eq}}$
Deviatoric part of stress tensor: $\mathbf{S}=\boldsymbol{\sigma}-\frac{1}{3}tr\boldsymbol{\sigma}\,\mathbf{I}$
Local equivalent plastic strain: $\varepsilon^p_{eq}=\sqrt{\frac{2}{3}\boldsymbol{\varepsilon}^p:\boldsymbol{\varepsilon}^p}$
J_2 plasticity yield function (linear isotropic hardening): $F=\sigma_{eq}-(\sigma_{y0}+H\varepsilon^p_{eq})$
$\sigma_{eq}=\sqrt{\frac{3}{2}\mathbf{S}:\mathbf{S}}$ - equivalent stress, σ_{y0} -initial yield stress, H - isotropic hardening modulus
Kuhn-Tucker (KT) conditions: $\lambda\geq 0,\quad F\leq 0,\quad \lambda F=0$
Since $\lambda=\varepsilon^p_{eq}$, KT conditions may be written as: $\varepsilon^p_{eq}\geq 0,\quad F\leq 0,\quad \varepsilon^p_{eq}F=0$

In the relations presented in Table 1, $\boldsymbol{\sigma}$ is the stress tensor, $\mathbf{f}$ denotes the body force, and $\boldsymbol{u}$ stands for the displacement vector. The values $\overline{\mathbf{u}}$ is the prescribed displacement on the boundary Γ_u, and $\overline{\mathbf{t}}$ is the prescribed traction on the boundary Γ_t, respectively. The vector $\mathbf{n}$ represents the unit normal to boundary Γ_t. The total strain measure $\boldsymbol{\varepsilon}$ is additively decomposed into the elastic $\boldsymbol{\varepsilon}^e$ and plastic $\boldsymbol{\varepsilon}^p$ contributions: $\boldsymbol{\varepsilon}= \boldsymbol{\varepsilon}^e + \boldsymbol{\varepsilon}^p$, and $\boldsymbol{C}$ is the fourth-order elasticity tensor.

During the training process, the following field variables are approximated at the coordinates $\mathbf{x}^T = \begin{bmatrix} x & y \end{bmatrix}$ of the domain considered for a two-dimensional elastoplastic problem

$$\mathbf{u}^T(\mathbf{x}) = \begin{bmatrix} u(\mathbf{x}) & v(\mathbf{x}) \end{bmatrix}, \tag{10}$$

$$\boldsymbol{\varepsilon}(\mathbf{x}) = \begin{bmatrix} \varepsilon_{xx} & \varepsilon_{xy} \\ \varepsilon_{yx} & \varepsilon_{yy} \end{bmatrix}, \tag{11}$$

$$\boldsymbol{\varepsilon}^p(\mathbf{x}) = \begin{bmatrix} \varepsilon^p_{xx}(\mathbf{x}) & \varepsilon^p_{xy}(\mathbf{x}) & \\ \varepsilon^p_{yx}(\mathbf{x}) & \varepsilon^p_{yy}(\mathbf{x}) & \\ & & \varepsilon^p_{zz}(\mathbf{x}) \end{bmatrix}, \tag{12}$$

$$\boldsymbol{\sigma}(\mathbf{x}) = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} & \\ \sigma_{yx} & \sigma_{yy} & \\ & & \sigma_{zz} \end{bmatrix}. \tag{13}$$

The data-driven loss may be formulated for different field variables as

$$L^D = L^D_{\mathbf{u}} + L^D_{\boldsymbol{\varepsilon}} + L^D_{\boldsymbol{\varepsilon}^p} + L^D_{\boldsymbol{\sigma}}, \tag{14}$$

where $L^D_{\mathbf{u}}$ is the displacement data loss function, $L^D_{\boldsymbol{\varepsilon}}$ is the strain data loss function, $L^D_{\boldsymbol{\varepsilon}^p}$ stands for the plastic strain data loss function, and $L^D_{\boldsymbol{\sigma}}$ represents the stress data loss function. The loss components in (14) are expressed using the regular L_2 norm $\|\bullet\|$ in the following form:

$$L^D_{\mathbf{u}} = \frac{1}{N^{\Omega}} \sum_{c=1}^{N^{\Omega}} \left\| \mathbf{u}(\mathbf{x}_c) - \hat{\mathbf{u}}(\mathbf{x}_c) \right\|, \tag{15}$$

$$L^D_{\boldsymbol{\varepsilon}} = \frac{1}{N^{\Omega}} \sum_{c=1}^{N^{\Omega}} \left\| \boldsymbol{\varepsilon}(\mathbf{x}_c) - \hat{\boldsymbol{\varepsilon}}(\mathbf{x}_c) \right\|, \tag{17}$$

$$L_{\boldsymbol{\varepsilon}^p}^D = \frac{1}{N^{\Omega}} \sum_{c=1}^{N^{\Omega}} \left\| \boldsymbol{\varepsilon}^p(\mathbf{x}_c) - \hat{\boldsymbol{\varepsilon}}^p(\mathbf{x}_c) \right\|, \tag{18}$$

$$L_{\boldsymbol{\sigma}}^D = \frac{1}{N^{\Omega}} \sum_{c=1}^{N^{\Omega}} \left\| \boldsymbol{\sigma}(\mathbf{x}_c) - \hat{\boldsymbol{\sigma}}(\mathbf{x}_c) \right\|. \tag{19}$$

In the above relations, $\mathbf{x}_c$ is the position of collocation points over the domain Ω, N^{Ω} is the number of collocation points, and $\hat{\mathbf{u}}(\mathbf{x}_c)$, $\hat{\boldsymbol{\varepsilon}}(\mathbf{x}_c)$, $\hat{\boldsymbol{\varepsilon}}^p(\mathbf{x}_c)$, $\hat{\boldsymbol{\sigma}}(\mathbf{x}_c)$ are the target values obtained by means of the FEM simulation.

The physics-based loss function may be written as the sum of the loss terms corresponding to different governing equations in the following form:

$$L^P = L_{ec}^P + L_{cl}^P + L_{fr}^P + L_{BC}^P + L_{KT}^P, \tag{20}$$

where L_{ec}^P corresponds to the equilibrium condition, L_{cl}^P is the loss term of the constitutive law, L_{fr}^P corresponds to the flow rule, L_{BC}^P is the boundary condition terms, and L_{KT}^P stands for the loss terms expressing the Kuhn-Tucker complementarity conditions. It is possible to introduce the penalty coefficients $\lambda_1 \ \ldots \lambda_4$ as the weights associated with some loss terms to regularize the solution process of an elastoplastic problem. Accordingly, equation (20) can be rewritten as

$$L^P = \lambda_1 L_{ec}^P + \lambda_2 L_{cl}^P + \lambda_3 L_{fr}^P + \lambda_4 L_{BC}^P + L_{KT}^P. \tag{21}$$

According to the equations displayed in Table 1, the loss terms are expressed in chosen collocation points over the domain Ω using the regular L_2 norm by the following relations:

$$L_{ec}^P = \frac{1}{N^{\Omega}} \sum_{c=1}^{N^{\Omega}} \left\| \nabla \boldsymbol{\sigma}(\mathbf{x}_c) + \mathbf{f}(\mathbf{x}_c) \right\|, \tag{22}$$

$$L_{cl}^P = \frac{1}{N^{\Omega}} \sum_{c=1}^{N^{\Omega}} \left\| \boldsymbol{\sigma}(\mathbf{x}_c) - \mathbf{C} : \left[\boldsymbol{\varepsilon}(\mathbf{x}_c) - \boldsymbol{\varepsilon}^p(\mathbf{x}_c) \right] \right\|, \tag{23}$$

$$L_{fr}^P = \frac{1}{N^{\Omega}} \sum_{c=1}^{N^{\Omega}} \left\| \boldsymbol{\varepsilon}^p(\mathbf{x}_c) - \frac{3}{2} \varepsilon_{eq}^p \frac{\mathbf{S}}{\sigma_{eq}}(\mathbf{x}_c) \right\|. \tag{24}$$

The boundary loss term is expressed as

$$L_{BC}^{P} = L_{D\mathbf{u}}^{P} + L_{N\mathbf{t}}^{P} \tag{25}$$

where $L_{D\mathbf{u}}^{P}$ and $L_{N\mathbf{t}}^{P}$ represent the loss component for the Dirichlet boundary condition and the Neumann boundary condition, respectively; they are written as

$$L_{D\mathbf{u}}^{P} = \frac{1}{N^{\Gamma_u}} \sum_{c=1}^{N^{\Gamma_u}} \left\| \mathbf{u}(\mathbf{x}_c) - \bar{\mathbf{u}}(\mathbf{x}_c) \right\|, \tag{26}$$

$$L_{N\mathbf{t}}^{P} = \frac{1}{N^{\Gamma_t}} \sum_{c=1}^{N^{\Gamma_t}} \left\| \boldsymbol{\sigma}(\mathbf{x}_c)\mathbf{n} - \bar{\mathbf{t}}(\mathbf{x}_c) \right\|. \tag{27}$$

In the above two equations, $\mathbf{x}_c$ denotes the position of collocation points along the boundaries Γ_u and Γ_t, while N^{Γ_u} and N^{Γ_t} stand for the number of these points.

The loss term corresponding to the Kuhn-Tucker complementarity conditions is expressed by three parts as

$$L_{KT}^{P} = L_{\varepsilon_{eq}^{p}}^{P} + L_{F}^{P} + L_{\varepsilon_{eq}^{p}F}^{P}, \tag{28}$$

where the terms $L_{\varepsilon_{eq}^{p}}^{P}$, L_{F}^{P}, $L_{\varepsilon_{eq}^{p}F}^{P}$ associated with the inequalities $\varepsilon_{eq}^{p} \geq 0, \quad F \leq 0, \quad \varepsilon_{eq}^{p} F = 0$ presented in Table 1 are expressed according to [48] in the following form:

$$L_{\varepsilon_{eq}^{p}}^{P} = \frac{1}{N^{\Omega}} \sum_{c=1}^{N^{\Omega}} \left\| (1 - \operatorname{sign} \varepsilon_{eq}^{p}(\mathbf{x}_c)) \left| \varepsilon_{eq}^{p}(\mathbf{x}_c) \right| \right\| \tag{29}$$

$$L_{F}^{P} = \frac{1}{N^{\Omega}} \sum_{c=1}^{N^{\Omega}} \left\| (1 + \operatorname{sign} F(\mathbf{x}_c)) \left| F(\mathbf{x}_c) \right| \right\| \tag{30}$$

$$L_{\varepsilon_{eq}^{p}F}^{P} = \frac{1}{N^{\Omega}} \sum_{c=1}^{N^{\Omega}} \left\| \varepsilon_{eq}^{p}(\mathbf{x}_c) F(\mathbf{x}_c) \right\| \tag{31}$$

It is to note that $\varepsilon_{eq}^{p} \geq 0$ is incorporated in the loss as $(1 - \operatorname{sign} \varepsilon_{eq}^{p}(\mathbf{x}_c)) \left| \varepsilon_{eq}^{p}(\mathbf{x}_c) \right|$. The inequality $F \leq 0$ is analogously expressed by equation (30).

The application of the presented PINN formulation has been illustrated in [55], where the elastoplastic analysis of the perforated plate has been performed under tension. The geometry of the computational model and the uniaxial extension displacement distribution are presented in Figure 8.

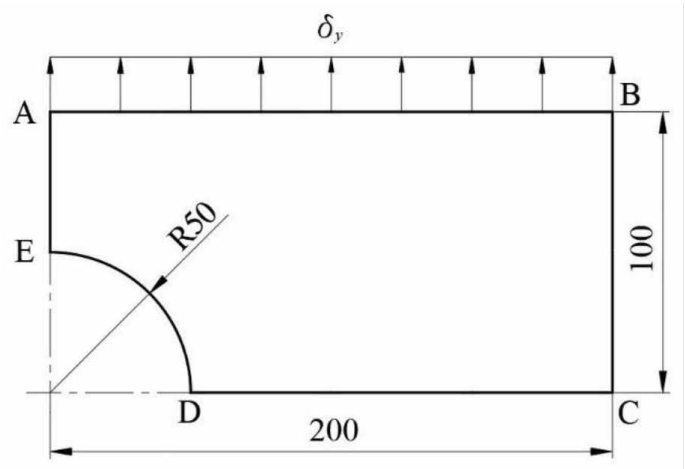


Figure 8. Quarter of perforated plate subjected to constant displacement
Slika 8. Četvrtina ploče s otvorom izložene konstantnom pomaku

As evident, one-quarter of the plate has been considered, and the symmetry has been employed along the boundaries $\overline{\mathrm{AE}}$ and $\overline{\mathrm{DC}}$, while the boundaries $\overline{\mathrm{BC}}$ and $\widehat{\mathrm{ED}}$ are stress-free. The displacement of $\delta_y = 2\ \mathrm{mm}$ is imposed along the $\overline{\mathrm{AB}}$ surface. The material data are Young's modulus $E = 7 \cdot 10^4\ \mathrm{MPa}$, Poisson's ratio $\nu = 0.2$, and $\sigma_{y0} = 250\ \mathrm{MPa}$. The plane strain condition and the linear isotropic hardening with tangent modulus of $E_t = 2171\ \mathrm{MPa}$ have been assumed. The PINN computations have been performed in 48,400 collocation points using the NN architecture of 5 hidden layers with 80 neurons each. The hyperbolic *tangent* (*tanh*) activation functions have been employed. According to [55], for optimal solutions for the elastoplastic problem considered, the penalty coefficients $\lambda_1 = 1, \lambda_2 = 3, \lambda_3 = 2, \lambda_4 = 1$ have been prescribed into the physics-based loss function expressed by equation (21). During training, the model has been run for 5,000 epochs, and the TensorFlow [56] computing system has been used.

The field variables predicted by the PINN approach have been compared with those computed by the FEM in [55]. In the FEM analysis, the problem domain has been discretized by free triangular consisting of 31,402 domain and 960 boundary elements. The refined mesh has been used to capture the stress-concentration behaviour near to the circular edge. The excellent agreement of the predicted field variables with the FEM solutions has been illustrated. The comparison of the plastic strain components approximated by the PINN to the values computed by the FEM is shown in Figure 9.

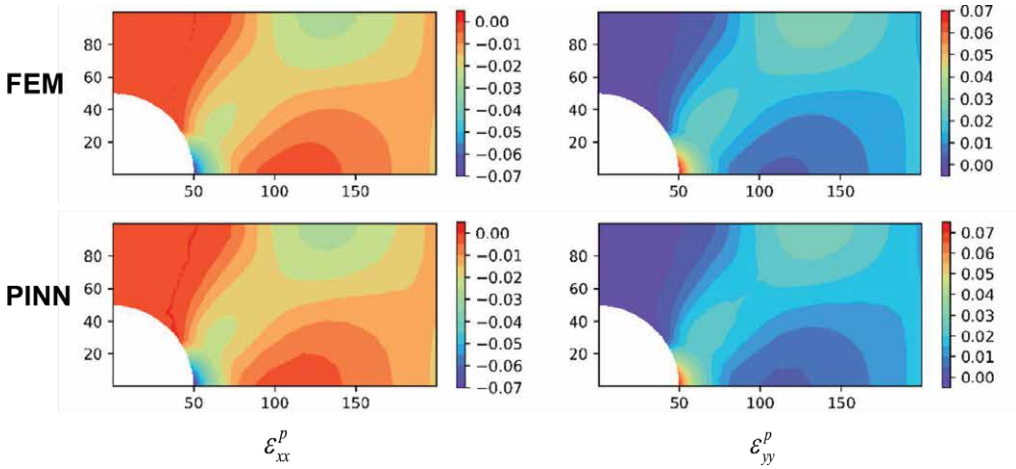


Figure 9. Contour plot of plastic strain components obtained by FEM and PINN formulation [55].
Slika 9. Grafički prikaz raspodjele komponenata plastične deformacije dobivenih metodom konačnih elemenata i PINN formulacijom [55].

It may be concluded that the results obtained from the PINNs predictions match well with the reference FEM solution both qualitatively and quantitatively for all field variables, as illustrated in more detail in [55]. These results show that the PINN model has computational efficiency in comparison with the conventional FE model. The non-linear iterations for the plastic correction schemes during the stress-integration in the framework of the FEM are avoided in the proposed NN formulation. Furthermore, the presented formulation builds a new promising avenue concerning future work in solving complex elastoplastic problems, which have so far not been solved by the conventional numerical algorithms based on continuum mechanics.

3.2 Energy based PINN application to phase-field brittle crack modelling

3.2.1 Phase-field approach to brittle crack modelling

Over the last decade, the phase-field approach has gained great popularity in the modelling of crack evolution in engineering structural components. The phase-field computational strategy is classified as a smeared approach, in which the sharp discontinuity caused by crack is distributed over a small, but finite width defined by the length-scale parameter governing the regularization. The phase-field variable is introduced,

which separates the broken and unbroken material states through a smooth transition. The formulation for the brittle crack modelling follows the Griffith theory of fracture. This approach can be efficiently used for solving the complex fracture processes such as the crack nucleation, propagation, merging, kinking and branching. More on the phase-field formulations for the quasi-static brittle fracture can be found in [57, 58, 59].

Following the variational approach applied, the fracture process is governed by the minimization of the free energy functional $\Psi(\mathbf{u},\phi)$ [59] given by

$$\Psi(\mathbf{u},\phi)=\int_{\Omega}\left[(1-\phi)^2\psi_e^+(\boldsymbol{\varepsilon})+\psi_e^-(\boldsymbol{\varepsilon})\right]\mathrm{d}\Omega+\int_{\Omega}\left\{\frac{G_c}{4l}\left[\phi^2+4l^2(\nabla\phi)^2\right]+(1-\phi)^2H(t)\right\}\mathrm{d}\Omega\,, \quad , \tag{32}$$

where $\phi=\phi(x)$ is the phase-field scalar parameter $\phi\in[0,1]$, expressed as

$$\phi(x)=\exp\left(\frac{-|x|}{l}\right). \tag{33}$$

The strain tensor $\boldsymbol{\varepsilon}$ is decomposed into the tensile $\boldsymbol{\varepsilon}^+$ and compressive parts $\boldsymbol{\varepsilon}^-$ via spectral decomposition:

$$\boldsymbol{\varepsilon}^+=\sum_{i=1}^{3}\langle\varepsilon_i\rangle^+\mathbf{n}_i\otimes\mathbf{n}_i\,, \tag{34}$$

$$\boldsymbol{\varepsilon}^-=\sum_{i=1}^{3}\langle\varepsilon_i\rangle^-\mathbf{n}_i\otimes\mathbf{n}_i\,, \tag{35}$$

where $\langle x\rangle_{\pm}:=\frac{1}{2}\left(x\pm|x|\right)$ are the Macaulay brackets, ε_i are the principal strains and $\mathbf{n}_i$ their respective directions. $\psi_e^+(\boldsymbol{\varepsilon})$ and $\psi_e^-(\boldsymbol{\varepsilon})$ denote the tensile and compressive elastic strain energy written as

$$\psi_e^+(\boldsymbol{\varepsilon})=\frac{\lambda}{2}\langle\mathrm{tr}\boldsymbol{\varepsilon}\rangle^2+\mu\,\mathrm{tr}\left[(\boldsymbol{\varepsilon}^+)^2\right], \tag{36}$$

$$\psi_e^-(\boldsymbol{\varepsilon})=\frac{\lambda}{2}\langle\mathrm{tr}\boldsymbol{\varepsilon}\rangle^2+\mu\,\mathrm{tr}\left[(\boldsymbol{\varepsilon}^-)^2\right] \tag{37}$$

with λ and μ as the Lamé constants. In above equation (32) G_c is the Griffit's fracture energy density, l denotes length-scale parameter, and $H(t)$ stands for the strain energy history function to restrict the crack healing in unloading. It contains maximum positive tensile energy:

$$H(t)=\max_{\tau\in[0,t]}\ \psi_e^+(\tau)\,. \tag{38}$$

The phase-field parameter continuously varies between the fractured $(\phi=1)$ and intact $(\phi=0)$ material states, as shown in Figure 10.

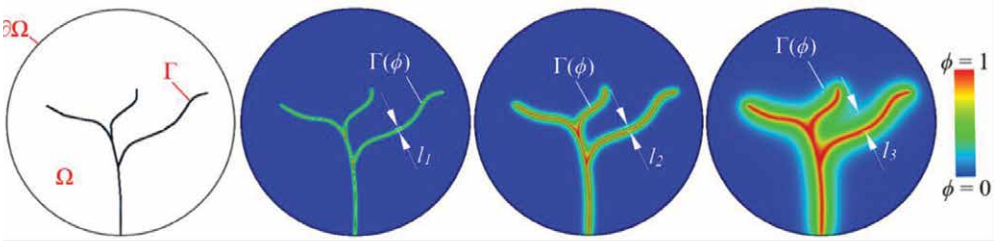


Figure 10. Schematic representation body Ω with discrete crack surface Γ and its diffusive approximations $\Gamma(\phi)$ governed by length-scale parameter *l* where $l_1 < l_2 < l_3$ [59].
Slika 10. Shematski prikaz tijela Ω s diskretnom pukotinom Γ i njenom difuzijskom aproksimacijom $\Gamma(\phi)$ upravljanom duljinskim parametrom l pri čemu je $l_1 < l_2 < l_3$ [59].

The minimization of the energy functional (32) leads to the following strong-form Eulerian equations representing the phase field governing equations:

$$\begin{aligned}
&\nabla\boldsymbol{\sigma} + \bar{\mathbf{b}} = 0 \quad \text{in } \Omega,\\
&\boldsymbol{\sigma}\cdot\mathbf{n} = \bar{\mathbf{t}} \quad \text{on } \Gamma_t,\\
&\mathbf{u} = \bar{\mathbf{u}} \quad \text{on } \Gamma_u,\\
&-4G_c l^2\Delta\phi + \left[4l\,H(t) + G_c\right]\phi = 4l\,H(t) \quad \text{in } \Omega,\\
&\nabla\phi\cdot\mathbf{n} = 0 \quad \text{on } \partial\Omega.
\end{aligned} \tag{39}$$

Herein, $\bar{\mathbf{b}}$ represents the volume forces, $\bar{\mathbf{u}}$ is the prescribed displacement on the boundary Γ_u, and $\bar{\mathbf{t}}$ is the prescribed traction on the boundary Γ_t, respectively. The vector $\mathbf{n}$ represents the unit normal to the boundary Γ_t. The stress tensor $\boldsymbol{\sigma}$ is defined as

$$\boldsymbol{\sigma} = (1-\phi)^2\frac{\partial\psi^+}{\partial\boldsymbol{\varepsilon}} + \frac{\partial\psi^-}{\partial\boldsymbol{\varepsilon}}. \tag{40}$$

3.2.2 Energy-based PINN

In the energy-based PINN, the physical information is incorporated in the NN architecture by means of the energy functional. Unlike most of the PINN algorithms, where the residual of the governing equations is minimized, the energy functional is minimized in the proposed PINN formulation. Accordingly, the loss function is expressed by

the energy functional. Since the Neumann boundary conditions have been included in the variational energy formulation in a natural way, only the Dirichlet boundary conditions have to be imposed [60, 61, 62]. A further advantage of the variational formulation is concerned with the order of derivatives present in the functional form which is of lower order than in the residual form used in the conventional PINN, which accelerates network training.

Here, the functional (32) is recalled and rewritten in the following form:

$$\Psi\left(\mathbf{u},\phi\right)=\int_{\Omega}\psi_e(\mathbf{x})\,\mathrm{d}\Omega+\int_{\Omega}\psi_c(\mathbf{x})\,\mathrm{d}\Omega\,, \tag{41}$$

where $\psi_e(\mathbf{x})$ and $\psi_c(\mathbf{x})$ are the elastic strain energy density and the fracture energy density, respectively:

$$\psi_e(\mathbf{x})=\left(1-\phi\right)^2\psi_e^{+}(\boldsymbol{\varepsilon})+\psi_e^{-}(\boldsymbol{\varepsilon})\,, \tag{42}$$

$$\psi_c(\mathbf{x})=\frac{G_c}{4l}\left[\phi^2+4l^2\left(\nabla\phi\right)^2\right]+\left(1-\phi\right)^2 H(t)\,. \tag{43}$$

Now the loss function can be defined as follows:

$$L\left(\mathbf{u},\phi\right)=\int_{\Omega}\psi_e(\mathbf{x})\,\mathrm{d}\Omega+\int_{\Omega}\psi_c(\mathbf{x})\,\mathrm{d}\Omega+\lambda_u\frac{1}{N^{\Gamma_u}}\sum_{i=1}^{N^{\Gamma_u}}\left|\mathbf{u}(\mathbf{x}_i)-\overline{\mathbf{u}}(\mathbf{x}_i)\right|^2. \tag{44}$$

The last term in (44) expresses the Dirichlet boundary condition $\mathbf{u}=\overline{\mathbf{u}}$ on Γ_u. $\mathbf{x}_i$ denotes the position of collocation points along the boundary, N^{Γ_u} is the number of points, and λ_u stands for the penalty parameter. Alternatively, in case that the Dirichlet boundary conditions are imposed strongly, the last term can be avoided, which simplifies the computation [62, 61]. According to the numerical integration strategy, the energy of the body is approximated by a weighted sum of the energy density at integration points. Then, relation (44) is transformed in the following form:

$$L\left(\mathbf{u},\phi\right)\approx\sum_{i=1}^{N_\Omega}\psi_e(\mathbf{x}_i)\,\omega(\mathbf{x}_i)+\sum_{i=1}^{N_\Omega}\psi_c(\mathbf{x}_i)\,\omega(\mathbf{x}_i)+\lambda_u\frac{1}{N^{\Gamma_u}}\sum_{i=1}^{N^{\Gamma_u}}\left|\mathbf{u}(\mathbf{x}_i)-\overline{\mathbf{u}}(\mathbf{x}_i)\right|^2, \tag{45}$$

where $\psi_e(\mathbf{x}_i)$ and $\psi_c(\mathbf{x}_i)$ are the energy approximation functions evaluated at integration points $\mathbf{x}_i$, and $\omega(\mathbf{x}_i)$ are their corresponding weights according to the Gauss quadrature rule. N_Ω is the number of integration points over domain Ω.

The PINN architecture applied is analogous to the network presented in Figure 7 for elastoplastic analysis. Instead of the elastoplastic field variables, here the output layer contains the predicted displacement $\mathbf{u}$ and the phase-field parameter ϕ. The loss function is expressed by the energy functional. The application of the presented energy

based PINN formulation has been illustrated in [61], where the two-dimensional examples dealing with the brittle crack evolution have been considered and compared to the FEM solutions. Here, the results of the single-edge notched shear test performed in [61] is presented.

A square plate with a horizontal notch presented in Figure 11 has been considered. A horizontal displacement with the increment of $\Delta u = 1 \cdot 10^{-6}$ mm is imposed at the upper boundary. The low end is clamped, and the vertical boundaries are free. The material parameters used are: Young's modulus $E = 210\,\text{GPa}$, Poisson's ratio $\nu = 0.3$, critical fracture energy density G_c = 2.7 N/mm, and length scale $l = 0.01\,\text{mm}$.

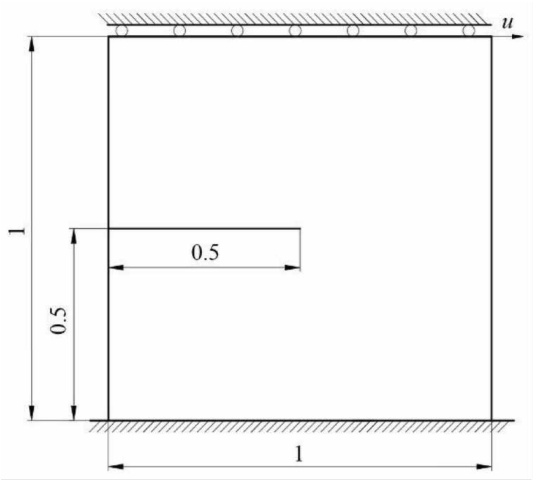
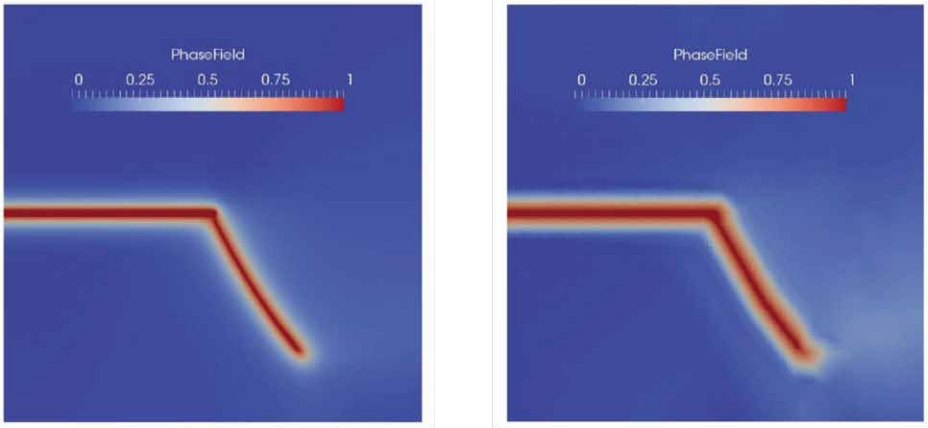


Figure 11. Geometry and boundary conditions of single-edge notched plate subjected to shear loading
Slika 11. Geometrija i rubni uvjeti zarezane ploče s narinutim smičnim opterećenjem

The PINN computations have been performed in 95,686 collocation points using the NN architecture of 4 hidden layers of 50 neurons each. Different activation functions such as the *ReLU*, *softplus* and *tanh* have been employed, however the *ReLU* has given the best prediction in terms of the crack propagation speed. During training, the Adam optimizer has been used. To compare the PINN predictions with the FE solutions, the FE analysis with 60,572 linear triangle elements has also been performed. The comparison of the PINN learning prediction to the FE crack pattern solutions is displayed in Figure 12. As evident, the FE results have been correctly reproduced.



a) b)

Figure 12. Single-edge notched shear test: a) FE crack pattern b) PINN learning prediction [61]
Slika 12. Test posmičnog opterećenja zarezane ploče: a) pukotina dobivena konačnim elementima b) predviđanje PINN učenjem [61]

4. CONCLUSION

A short review of the artificial neural network methods most common used in solid mechanics as an alternative to the conventional finite element application is given. The feedforward neural network (FFNN), and the physics-informed neural network (PINN), employing both the governing partial differential equations (PDEs) and the energy functional, are presented and discussed in more detail. A linear elastic boundary value problem is solved by means of the FFNN, and the results are compared to the finite element (FE) solutions. The excellent agreement of the state variables predicted by both approaches is displayed.

In the PINN framework governed by the PDEs, an elastoplastic problem is considered. The multi-objective loss function consisting of the data-driven loss in terms of the field variables and the physics-based loss employing the governing equations is presented. Their application is demonstrated by the modelling of the elastoplastic response of the perforated plate under tension, and the results obtained are again compared to the FE solutions. An agreement of the field variables has been achieved. The PINN approach has shown an advantage in computational efficiency.

A brittle crack modelling by means of the energy based PINN has been performed by using the phase-field theory. The governing phase-field equations, as well as the energy functional expressions, are displayed. The PINN formulation has been tested by

solving a single-edge notched shear problem, where the crack evolution has been considered. Therein, different activation functions have been used, and the results have been compared with those obtained by the FE formulation. It has been found that the crack response of the sample analyzed depends on the chosen activation function. Among the activation functions applied, only the *ReLU* function reproduces the finite element results correctly. The formulation of an efficient loss function is still an open issue and remains a great challenge in further research.

The artificial neural networks enable accelerating numerical simulations and improving accuracies. The design of their architecture is currently an extremely active research area, especially in the field of solid mechanics. The improvement of NN procedures, as well as algorithms applied, will in the near future contribute to their application in the numerical modelling of real structural responses, together with progress in computer power.

References

1. Herrmann L, Kollmannsberger S. Deep learning in computational mechanics: a review. Computational Mechanics, 2024; 74;281–331.
2. Butler KT, Davies DW, Cartwright H, Isayev O, Walsh A. Machine learning for molecular and materials science. Nature. 2018;559(7715):547–55.
3. Goodfellow I, Bengio Y, Courville A. Deep learning. MIT Press. The MIT Press, Massachusetts Institute of Technology; 2021.
4. Haykin S. Neural Networks. A Comprehensive Foundation. Prentice Hall; 1999.
5. Hagan MT, Demuth HB, Beale MH. Neural Network Design (2nd Edition). 2nd ed. Martin Hagan; 2014.
6. Yagawa G, Oishi A. Computational mechanics with neural networks. 1st ed. Cham, Switzerland: Springer Nature; 2021.
7. Aldakheel F, Satari R, Wriggers P. Feed-forward neural networks for failure mechanics problems. Applied Sciences 2021;11(14):1-22.
8. Dornheim J, Morand L, Nallani HJ, et al. Neural Networks for Constitutive Modeling: From Universal Function Approximators to Advanced Models and the Integration of Physics. Arch Computat Methods Eng. 2024; 31: 1097–1127. https://doi.org/10.1007/s11831-023-10009-y.
9. Li H, Kafka OL, Gao J, Yu C, Nie Y, Zhang L, et al. Clustering discretization methods for generation of material performance databases in machine learning and design optimization. Computational Mechanics. 2019 May 22;64(2):281–305.

10. Shen Y, Chandrashekhara K, Breig WF, Oliver LR. Neural Network Based Constitutive Model for Rubber Material. Rubber Chemistry and Technology. 2004;77(2):257–77.
11. Kushvaha V, Kumar SA, Madhushri P, Sharma A. Artificial neural network technique to predict dynamic fracture of particulate composite. Journal of Composite Materials. 2020;54(22):3099–108.
12. LeCun Y, Bengio Y, Hinton G. Deep Learning. Nature. 2015;521(7553):436–44.
13. Schmidhuber J. Deep learning in neural networks: An overview. Neural Networks. 2015;61:85–117.
14. Liu Z, Wu CT, Koishi M. A deep material network for multiscale topology learning and accelerated nonlinear modeling of heterogeneous materials. Computer Methods in Applied Mechanics and Engineering. 2019;345:1138–68.
15. Lew AJ, Yu CH, Hsu YC, Buehler MJ. Deep learning model to predict fracture mechanisms of graphene. 2D Materials and Applications. 2021;5(1).
16. Betancourt D, Freitag S, Muhanna RL. Damage detection for structural health monitoring under uncertainty using deep interval neural networks. In 2021: Proceedings of International Workshop on Reliable Engineering Computing (REC2021) 2021;.9:1-16.
17. Kim Y, Kim Y, Yang C, Park K, Gu GX, Ryu S. Deep learning framework for material design space exploration using active transfer learning and data augmentation. Computational Materials. 2021;7(1).
18. Agrawal A, Choudhary A. Deep materials informatics: Applications of deep learning in materials science. MRS Communications. 2019;1–14.
19. Linka K, Hillgärtner M, Abdolazizi KP, Aydin RC, Itskov M, Cyron CJ. Constitutive artificial neural networks: A fast and general approach to predictive data-driven constitutive modeling by deep learning. Journal of Computational Physics. 2021;429:110010.
20. Wang Y, Oyen D, Guo W (Grace), Mehta A, Scott CB, Panda N, et al. StressNet - Deep learning to predict stress with fracture propagation in brittle materials. Materials Degradation. 2021;5(1).
21. Oishi A, Yagawa G. Computational mechanics enhanced by deep learning. Computer Methods in Applied Mechanics and Engineering. 2017;327:327–51.
22. Saha S, Gan Z, Cheng L, Gao J, Kafka OL, Xie X, et al. Hierarchical Deep Learning Neural Network (HiDeNN): An artificial intelligence (AI) framework for computational science and engineering. Computer Methods in Applied Mechanics and Engineering. 2021;373:113452.

23. Zhang L, Cheng L, Li H, Gao J, Yu C, Domel R, et al. Hierarchical deep-learning neural networks: finite elements and beyond. Computational Mechanics. 2021;67:207–30.
24. Wang Y, Soutis C, Ando D, Sutou Y, Narita F. Application of deep neural network learning in composites design. European Journal of Materials. 2022;2(1):117–70.
25. Saxena A. An Introduction to Convolutional Neural Networks. International Journal for Research in Applied Science & Engineering Technology 2022;10:943-947.
26. Indolia S, Goswami AK, Mishra SP, Asopa P. Conceptual Understanding of Convolutional Neural Network- A Deep Learning Approach. Procedia Computer Science 2018;132:679–88.
27. Abueidda DW, Almasri M, Ammourah R, Ravaioli U, Jasiuk IM, Sobh NA. Prediction and optimization of mechanical properties of composites using convolutional neural networks. Composite Structures 2019;227:111264.
28. Cang R, Li H, Yao H, Jiao Y, Ren Y. Improving direct physical properties prediction of heterogeneous materials from imaging data via convolutional neural network and a morphology-aware generative model. Computational Materials Science. 2017;150:212.
29. Li X, Liu Z, Cui S, Luo C, Li C, Zhuang Z. Predicting the effective mechanical property of heterogeneous materials by image based modeling and deep learning. Computer Methods in Applied Mechanics and Engineering. 2019;347:735–53.
30. Rao C, Liu Y. Three-dimensional convolutional neural network (3D-CNN) for heterogeneous material homogenization. Computational Materials Science. 2020;184:109850.
31. Kelleher JD. Deep learning. MIT Press. The MIT Press, Massachusetts Institute of Technology; 2021.
32. Lipton ZC, Berkowitz J, Elkan C. A Critical Review of Recurrent Neural Networks for Sequence Learning. arXiv.org. 2015;1506.00019.
33. Hochreiter S, Schmidhuber J. Long Short-Term Memory. Neural Computation. 1997;9(1):1–42.
34. Cho K, van Merrienboer B, Bahdanau D, Bengio Y. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. arXiv.org. 2014;1409.1259.
35. Mozaffar M, Bostanabad R, Chen W, Ehmann K, Cao J, Bessa MA. Deep learning predicts path-dependent plasticity. Proceedings of the National Academy of Sciences. 2019;116(52):26414–20.
36. Wu L, Nguyen VD, Kilingar NG, Noels L. A recurrent neural network-accelerated multi-scale model for elasto-plastic heterogeneous materials subjected to random cyclic and non-proportional loading paths. Comput Methods Appl Mech Eng. 2020 ;(369) :113234.

37. Logarzo HJ, Capuano G, Rimoli JJ. Smart constitutive laws: Inelastic homogenization through machine learning. Comput Methods Appl Mech Eng. 2021;373(113482):113482.
38. Bonatti C, Mohr D. On the importance of self-consistency in recurrent neural network models representing elasto-plastic solids. Journal of the Mechanics and Physics of Solids 2022;158:104697.
39. Ghavamian F, Simone A. Accelerating multiscale finite element simulations of history-dependent materials using a recurrent neural network. Comput Methods Appl Mech Eng 2019;357(112594):112594.
40. Schwarzer M, Rogan B, Ruan Y, Song Z, Lee DY, Percus AG, et al. Learning to fail: Predicting fracture evolution in brittle material models using recurrent graph convolutional neural networks. Computational Materials Science. 2019;162:322–32.
41. Koeppe A, Bamer F, Hernandez Padilla CA, Markert B. Neural network representation of a phase-field model for brittle fracture. PAMM. 2017;17(1):253–4.
42. Vaswani A, Shazeer N, Parmar N, et al. "Attention is all you need," in Advances in Neural Information Processing Systems, I. Guyon et al., eds. 2017; (30). arXiv: 1706.03762.
43. Islam S, Hanae Elmekki H, Elsebai A, et al. A comprehensive survey on applications of transformers for deep learning tasks, Expert Systems with Applications Volume. 2024; (241).122666.
44. Pitz, E., Pochiraju, K.: A neural network transformer model for composite microstructure homogenization. Engineering Applications of Artificial Intelligence 2024; (134).108622.
45. Li Q-J, Cinbizb MN, Zhanga Y. et al Robust deep learning framework for constitutive relations modeling, Acta Materialia. 2023;(254).118959.
46. Karniadakis GE, Kevrekidis IG, Lu L, Perdikaris P, Wang S, Yang L. Physics-informed machine learning. Nature Reviews Physics. 2021;3(6):422–40.
47. Das S, Tesfamariam S. State-of-the-Art Review of Design of Experiments for Physics-Informed Deep Learning. arxiv.org 2022;2202.06416.
48. Haghighat E, Raissi M, Moure A, Gomez H, Juanes R. A deep learning framework for solution and discovery in solid mechanics. arXiv.org 2020;2003.02751.
49. Arora R, Kakkar P, Dey B, Chakraborty A. Physics-informed neural networks for modeling rate- and temperature-dependent plasticity. arXiv.org 2022;2201.08363.
50. Bai J, Rabczuk T, Gupta A, Alzubaidi L, Gu Y. A physics-informed neural network technique based on a modified loss function for computational 2D and 3D solid mechanics. Comput Mech. 2023;71(3):543–62.

51. Li W, Bazant MZ, Zhu J. A physics-guided neural network framework for elastic plates: Comparison of governing equations-based and energy-based approaches. Computer Methods in Applied Mechanics and Engineering. 2021;383:113933.
52. Goswami S, Anitescu C, Chakraborty S, Rabczuk T. Transfer learning enhanced physics informed neural network for phase-field modeling of fracture. Theoretical and Applied Fracture Mechanics 2020;106:102447.
53. Güneş Baydin A, Pearlmutter B, Siskind J, Baydin G, Radul A, Mark J. Automatic Differentiation in Machine Learning: a Survey. Journal of Machine Learning Research. 2018;18:1–43.
54. Sorić J. Finite Element Method, Linear and Nonlinear Analysis of Structures. Golden marketing-Tehnička knjiga. Zagreb 2021 (in Croatian).
55. Roy, A.M and Guha, S. Elastoplastic Physics-Informed Deep Learning Approach for J2 Plasticity. 2022. Available at SSRN: https://ssrn.com/abstract=4332254.
56. Abadi M, Barham P, Chen J, et al. (2016). Tensorflow: A system for large-scale machine learning. In 12th (USENIX) symposium on operating systems design and implementation (OSDI16). 2016: 265-283.
57. Miehe C., Welschinger F. and Hofacker M. Comput Methods Appl Mech Eng. 2010; (199): 2765-2778.
58. Seleš K., Lesičar T, Tonković Z, Sorić J. A Residual Control Staggered Solution Scheme for the Phase-Field Modeling of Brittle Fracture. Engineering fracture mechanics. 2018; (205): 370-386.
59. Seleš K, Jurčević A, Tonković Z, Sorić J. Crack propagation prediction in heterogeneous microstructure using an efficient phase-field algorithm. Theoretical and Applied Fracture Mechanics. 2019; (100):289-297.
60. Goswami S, Anitescu C, Chakraborty S, Rabczuk T. Transfer learning enhanced physics informed neural network for phase-field modeling of fracture. Theor Appl Fract Mech. 2020;(106):102447.
61. Motlagh Y G, Jimack PK, de Borst R. Deep learning phase-field model for brittle fractures, Int J Numer Methods Eng. 2023; (124): 620–638.
62. Samaniego E, Anitescu C, Goswami S. et al. An Energy Approach to the Solution of Partial Differential Equations in Computational Mechanics via Machine Learning: Concepts, Implementation and Applications. Comput Methods Appl Mech Eng. 2020; (362): 112790.

O PRIMJENI UMJETNIH NEURONSKIH MREŽA U MEHANICI ČVRSTOG TIJELA KAO ALTERNATIVI KONVENCIONALNOM MODELIRANJU KONAČNIM ELEMENTIMA

Sažetak

Zbog složenosti inženjerskih problema i povećanih mogućnosti računala, nova računalna strategija koja koristi umjetne neuronske mreže nedavno se pojavila kao alternativa numeričkom modeliranju konvencionalnom primjenom konačnih elemenata. Neuronske mreže temeljna su tehnologija u okviru strojnog učenja, koje je potpodručje umjetne inteligencije, a primjenjuju se za rješavanje problema računalne mehanike, posebno u području mehanike čvrstog tijela. U ovom radu dan je kratak pregled neuronskih mreža, a detaljnije su prikazane i razmatrane unaprijedna neuronska mreža i fizikalno informirana neuronska mreža. U okviru formulacija neuronskih mreža temeljenih na fizici, u funkcijama gubitka koriste se osnovne parcijalne diferencijalne jednadžbe i energijski funkcional. Postupak unaprijednih neuronskih mreža testiran je linearnom elastičnom analizom, dok je učinkovitost fizikalno informirane neuronske mreže prikazana modeliranjem elastoplastičnih konstrukcijskih odziva i dvodimenzijskog širenja pukotine primjenom teorije faznog polja. Svi rezultati uspoređeni su s rješenjima konačnih elemenata. Pokazano je da algoritmi neuronskih mreža ispravno reproduciraju rezultate konačnih elemenata i imaju veću računalnu učinkovitost.

Ključne riječi: umjetne neuronske mreže; unaprijedna neuronska mreža; fizikalno informirana neuronska mreža; linearno elastična analiza; elastoplastična analiza; širenje pukotine.

Jurica Sorić
University of Zagreb
Faculty of Mechanical Engineering and Naval Architecture
Ivana Lučića 5, HR 10000 Zagreb, Croatia
E-mail: jurica.soric@fsb.unizg.hr