ARTIFICIAL NEURAL NETWORK SURROGATES FOR IMPACT PROBLEM SIMULATIONS

Yuyi Zhang
Faculty of Mathematics and Mechanics
Saint Petersburg State University
Russia
lesliezhang0825@gmail.com

Andrey Logachov
Faculty of Mathematics and Mechanics
Saint Petersburg State University
Russia
andre.log@bk.ru

Aleksandr Smirnov
Faculty of Mathematics and Mechanics
Saint Petersburg State University
Russia

Nikita Kazarinov*
Saint Petersburg State University
Institute of Problems of Mechanical Engineering, Russia
nkazarinov@gmail.com

Article history:
Received 05.04.2025, Accepted 05.05.2025

Abstract

This study investigates the use of artificial neural network (ANN)-based surrogate models for predicting residual velocities in impact problems under varying conditions. Building upon the combination of FEM simulations and incubation time fracture criteria, this paper extends the methodology by exploring multiple ANN architectures tailored to capture complex impact dynamics, including cases involving composite targets and variable impactor velocities. Special attention is given to identifying failure modes of the FEM model. The results demonstrate that appropriately configured ANN surrogates can effectively reduce computational costs and achieve accurate predictions in the case of high-velocity impacts.

Key words

Impact, composite, FEM, neural network, incubation time

1 Introduction

In recent years, neural networks—particularly artificial neural networks (ANN) and convolutional neural networks (CNN)—have garnered considerable attention for their ability to address a broad spectrum of problems in engineering [Knyazev et al., 2023, Kamalov et al., 2023, Istomin and Pavlov, 2024, Gromov et al., 2024, Le Viet et al., 2024, Tarasova and Moseiko, 2025] and mechanics [Istomin et al., 2023, Eischens et al., 2025, Li and Mao, 2024, Zhao et al., 2024a, Zhang et al., 2024, Kazarinov and Khvorov, 2024]. Their applications have been especially prominent in fluid dynamics [Rana et al., 2024, Sharma et al., 2023, Souayeh et al., 2021, Brown et al., 2022, Portal-Porras et al., 2021] and solid mechanics [Singh et al., 2024, Ishtiaq et al., 2025, Mianroodi et al., 2021, Gao et al., 2025]. When combined with numerical methods, neural networks have demonstrated significant efficacy in solving complex engineering challenges. They have been successfully employed for shape optimization [Zheng et al., 2025, Leng et al., 2024], material property estimation [Ejaz et al., 2022, Tariq et al., 2024], and modeling of intricate mechanical systems [Rojek et al., 2021, Wen et al., 2022], many of which are otherwise computationally demanding. These advancements have not only broadened the role of machine learning in engineering but have also enhanced the efficiency of design and optimization processes—particularly in scenarios involving high complexity or large-scale simulations.

CNNs, in particular, have shown notable promise in solid mechanics, especially in the design and optimization of composites [Shokrollahi et al., 2023,Tabian et al., 2019], metamaterials [Zhao et al., 2024b, Hu et al., 2024], nanomaterials [Choudhary et al., 2022], and other structurally complex materials [Alli et al., 2024]. A key advantage of CNNs lies in their ability to efficiently capture and extract features from high-dimensional data [Rao and Liu, 2020]. This feature extraction capability is critical for training accurate predictive models and is instrumental in optimizing network performance. Compared to conventional ANNs, CNNs possess greater representational power, making them particularly well-suited for handling the complex data structures frequently encountered in materials science [Duran et al., 2025]. Moreover, the inherent architecture of CNNs enables them to more effectively address challenges such as cross-scale analysis [Wu et al., 2025]. This architectural advantage leads to improved accuracy and computational efficiency in material design, thereby streamlining the development of advanced mechanical systems.

Despite their advantages, CNNs exhibit certain limitations. Their performance is highly sensitive to the quality and distribution of the input data. In situations where datasets are limited or imbalanced, CNNs are susceptible to overfitting—yielding high performance on training data but poor generalization to unseen data [Santos and Papa, 2022, Bejani and Ghatee, 2021]. This issue is particularly acute in complex mechanical systems, where acquiring extensive experimental or simulation data can be challenging. To address these limitations, researchers have proposed various enhancements [Cao et al., 2022, Bacanin et al., 2022]. Among the most effective strategies are deeper network architectures and alternative designs, such as residual networks (ResNets) [He et al., 2016, Shafiq and Gu, 2022], which improve multiscale feature extraction and enhance model generalization.

The convergence of growing computational capabilities and the exponential increase in data availability has made the integration of machine learning with traditional numerical techniques—such as the Finite Element Method (FEM) and Finite Difference Method (FDM)—an increasingly vital approach in engineering mechanics [Huang et al., 2025]. Conventional numerical methods, while highly accurate, are often computationally expensive, particularly when applied to large-scale structures or problems involving dynamic loading conditions [Cheng et al., 2024]. Their high computational cost and resource demands continue to pose significant challenges [Kononenko and Kononenko, 2018]. Consequently, recent research has focused on incorporating machine learning into these frameworks to improve computational efficiency without sacrificing accuracy [Mitusch et al., 2021]. A common approach involves replacing specific components of the numerical computation process with machine learning models [Jokar and Semperlotti, 2021]. For example, in finite element analysis, tasks such as iterative meshing, equation solving, and repeated simulations can be time-consuming. By leveraging neural networks, researchers can develop surrogate models that predict structural responses directly from design parameters or initial conditions, thereby bypassing many of the iterative steps [Mendizabal et al., 2020]. This methodology can significantly reduce computation times, particularly in applications such as structural and shape optimization or material design [Cheng et al., 2024]. In such contexts, machine learning models offer rapid approximations that can subsequently be refined using traditional numerical methods, thereby achieving a balance between computational efficiency and predictive accuracy [Huang et al., 2025].

In this study, we develop and evaluate artificial neural network (ANN)-based surrogate models for predicting residual velocities in impact scenarios characterized by complex and variable conditions. By leveraging datasets generated from finite element method (FEM) simulations and guided by the incubation time fracture criterion, our approach advances previous work by systematically investigating multiple ANN architectures tailored to capture intricate impact dynamics. These include scenarios involving composite target materials and a range of impactor velocities. The findings demonstrate that well-configured ANN surrogates not only maintain high predictive accuracy but also significantly reduce computational costs, particularly in high-velocity impact conditions. The main contributions of this study are as follows:

Development of ANN-based surrogate models for impact prediction: This work introduces a suite of artificial neural network architectures specifically designed to predict residual velocities under varying impact conditions, including high-velocity and composite target scenarios.

Integration of FEM simulations and incubation time fracture criteria: By combining finite element simulation outputs with a dynamic fracture model, the study generates high-quality training data and identifies critical failure modes to inform surrogate model design.

Significant reduction in computational cost with preserved accuracy: The proposed ANN surrogates achieve accurate predictions while dramatically lowering the computational burden compared to traditional FEM approaches, making them suitable for rapid analysis and design optimization in impact engineering.

2 Materials, Methods and Problem Statement

In this section, we present the problem under investigation. Specifically, we analyze the penetration behavior of composite barriers impacted by steel projectiles, employing both the finite element method (FEM) and artificial neural networks (ANNs). The base target material is assumed to be polymethylmethacrylate (PMMA), and the residual velocity of the projectile is used as a key performance indicator of the barrier's resistance—the lower the residual velocity, the greater the protective capability of the barrier. We begin by addressing a fundamental case: the penetration of a homogeneous PMMA target by steel projectiles at various impact velocities. This scenario is simulated using FEM in conjunction with the incubation time fracture criterion (ITFC), and the simulation results are calibrated against empirical impact data. Next, we extend the analysis to composite targets

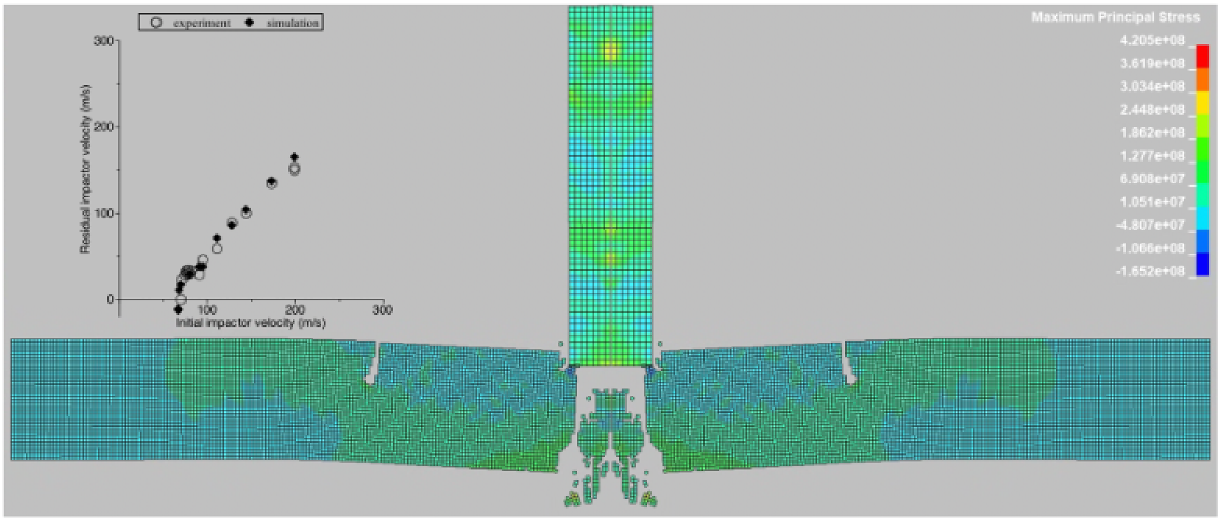

Figure 1. The FEM Model calibration – homogenous target configuration; experimental results from.

composed of segments with differing mechanical properties. A series of such simulations is conducted for targets with varied compositions, generating a dataset of solutions. This dataset is then used to train artificial neural networks capable of predicting the residual velocity for other configurations within the same class of problems. Further methodological details are provided in the subsequent subsections.

2.1 Problem Statement

The objective of this study can be formally articulated as follows: Consider a family of MM problems, where each individual problem is defined by a parameter vector $P^j = (P_1^j, P_2^j, \ldots, P_N^j)$, with $j \in [1, M]$. Each configuration P^j is mapped to an outcome R^j through the finite element method (FEM) or an equivalent numerical approach. We explicitly acknowledge the possibility of unsuccessful computations, wherein no result is obtained for certain configurations (i.e., $R^j = \emptyset$), due to the failure of the numerical scheme. Accordingly, the dataset-comprising this family of configurations and corresponding outcomes-can be represented as:

$$\left\{P^j \to R^j\right\}, j \in [1, M] \tag{1}$$

The aim is to develop an algorithm capable of mapping a new, previously unsolved configuration P^{M+1} to a corresponding result R^{M+1}, without executing the full numerical solution. This algorithm should also be able to handle cases where $R^j = \emptyset$, i.e., when the numerical solution fails. These problematic cases are addressed using knowledge derived from configurations that have been successfully solved. The parameter vector P^j may represent various aspects of the system, such as sample geometry, material properties, loading conditions, or combinations thereof. In the context of this study, P^j specifically characterizes the composite configuration. Artificial neural networks (ANNs) are employed as the predictive algorithm, trained on a dataset comprising M successfully solved problems. The dataset size M is assumed to be sufficiently large to ensure the predictive performance of the ANN has reached a stable level.

2.2 Materials

As previously noted, PMMA is used as the material for the baseline case involving the impact of a homogeneous PMMA target by a cylindrical steel projectile. The performance of the FEM model is validated against experimental data, with both the numerical and experimental results for the homogeneous target presented in Figure 1.

From a mechanical perspective, the problem can be described as follows: a 10 mm thick circular PMMA plate with a diameter of 100 mm is impacted by a cylindrical steel projectile. The projectile has a diameter of 6.9 mm and a length of 30 mm.

As the projectile interacts with the target, it decelerates, and the objective of the analysis is to determine its residual velocity after penetration. The extent of deceleration is influenced by the configuration of the target material, and therefore, an artificial neural network (ANN) is employed to predict the residual velocity of the projectile for various target configurations. To enable accurate prediction, the following material parameters are taken into account: Initial velocity of the projectile; Density of the projectile; Elastic modulus of the projectile; Poisson's ratio of the projectile; Density of the target; Elastic modulus of the target; Poisson's ratio of the target; Static strength of the target; Incubation time of the target. In this study, the parameter values are varied within ±20% and ±40% of their baseline values, while the initial velocity is selected from a specified range of 30 m/s to 300 m/s.

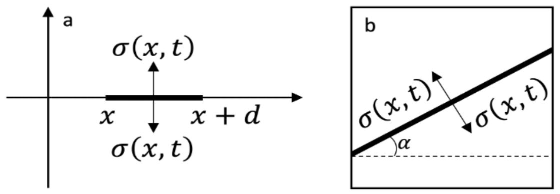

Figure 2. Spatial integration in the ITFC.

2.3 Methods

This section outlines the numerical approaches employed to generate the dataset, as well as the artificial neural network (ANN) architectures used for predicting the residual velocity.

2.3.1 Finite Element Model (FEM) The impact problem is simulated using the LS-DYNA finite element solver, which operates with an explicit time integration scheme. A custom material model is incorporated into LS-DYNA to implement a dynamic fracture mechanism based on the incubation time fracture criterion (ITFC). The computational domain is discretized using square elements with four integration points, and all target configurations are meshed with uniformly sized square elements, as required by the constraints of the fracture model. The FEM model is calibrated against established experimental data [reference], and both the experimental and numerical results are presented in Figure 1. Additionally, the stress distribution at 36 µs is illustrated for a projectile velocity of 91 m/s.

The dynamic fracture of brittle solids is a complex phenomenon that necessitates the application of appropriate fracture models to enable reliable simulation outcomes. These models must be capable of capturing key dynamic fracture effects, such as the dependence of material strength on the loading rate (e.g., see [Ravi-Chandar and Knauss, 1984, Cadoni et al., 2013]) and the phenomenon of fracture delay [Kalthoff and Shockey, 1977, Shockey et al., 1986, Mikhailova et al., 2017]. Common approaches for investigating dynamic fracture include models based on the stress intensity factor, which are widely used in studies of crack propagation [Rosakis and Ravichandran, 2000], as well as models that explicitly incorporate strain rate effects, such as the widely adopted Johnson-Holmquist (JH-2) model [Johnson and Holmquist, 1994]. In the present study, dynamic fracture is predicted using the incubation time model [Petrov and Utkin, 1989, Petrov, 1991], which has demonstrated effectiveness across various scenarios, including dynamic crack propagation [Smirnov et al., 2019, Kazarinov et al., 2021], spallation [Mikhailova et al., 2017], impact events [Kazarinov et al., 2020], and solid particle erosion [Evstifeev et al., 2018]. This model is particularly attractive due to its relative simplicity and robust performance. Accordingly, the incubation time approach has been employed in this work. The model is described as follows:

$$\frac{1}{\tau}\int_{t-\tau}^{t}\frac{1}{d}\int_{x}^{x+d}\sigma(x',t')dx'dt' \geq \sigma_c \qquad (2)$$

In equation 2, σ_c represents the ultimate stress of the material under investigation, and d denotes the characteristic minimal size of the fractured zone. This parameter defines the scale below which damaged areas are not classified as fractures. The parameter d is calculated using the expression:

$$d = \frac{2K_{Ic}^2}{\pi\sigma_c^2}$$

where K_{Ic} is the critical stress intensity factor. Notably, fracture is an inherently non-local phenomenon, implying that models relying on a single point cannot adequately capture fracture behavior. However, at the macro-scale, fracture is typically viewed as the culmination of preliminary damage processes, including the coalescence of microdefects, microcracks, and voids. These processes are influenced by the material's damage history and a time-dependent parameter known as the incubation time τ. Criterion (2) is commonly employed to predict brittle fracture, and significant success has been reported in dynamic plasticity modeling using the incubation time approach [Selyutina and Petrov, 2022].

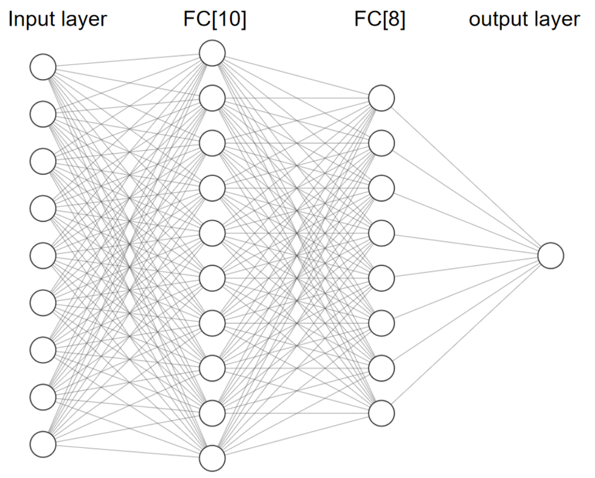

Figure 3. Architecture diagram of the artificial neural network. The dimension of the input to the network structure shown as an example is 9. The first hidden layer is a fully connected network containing 10 neurons, denoted using FC[10], and the second hidden layer is a fully connected network containing 8 neurons, denoted using FC[8]. This network structure can be abbreviated as FC[10, 8], and we still extend this naming convention in the following.

The model is implemented in LS-DYNA using user-defined material subroutines (UMAT41). Stress histories

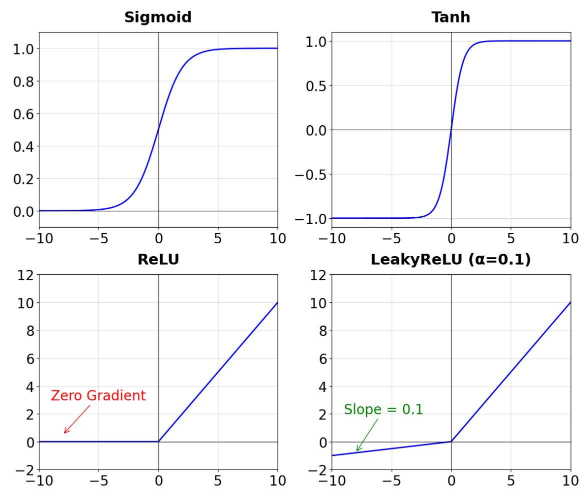

Figure 4. Visualisation of common activation functions.

are recorded in designated arrays, and the time integration in equation (2) is performed using the trapezoidal rule. A regular mesh with square elements is used in all simulations, with stresses calculated at four Gauss integration points per element. These stresses are then averaged to compute the inner spatial integral in equation (2). If criterion (2) predicts fracture at any integration point within an element, the element is removed from the mesh. The following angular areas are evaluated (refer to Figure 2b): $\alpha = 0, \frac{\pi}{2}, \pm\frac{\pi}{3}, \pm\frac{\pi}{4}, \pm\frac{\pi}{6}$.

3 ANN models

3.1 Network structure

The Artificial Neural Network (ANN) is a class of mathematical models used for information processing and learning, which is inspired by the connections between neurons in biological systems. An ANN consists of an input layer, one or more hidden layers, and an output layer, with neurons in each layer interconnected through fully connected (FC) connections (Figure 3).

The input layer receives the data, with each node corresponding to an individual input variable. The hidden layer, comprising multiple neurons, processes these inputs by applying weights through the fully connected structure. Finally, the output layer generates the network's prediction. In the forward propagation phase, the input data is passed through the network's fully connected layers to compute the output. Let the inputs be denoted as $\mathbf{x} = [x_1, x_2, \ldots, x_n]$. Each neuron within the network performs a linear combination of these inputs, which is subsequently transformed through a nonlinear activation function. Specifically, the output of a given neuron is given by:

$$\begin{aligned} z_j &= \sum_{i=1}^{n} \omega_{ij} x_i + b_j \\ y_j &= f(z_j) \end{aligned} \tag{3}$$

where w_{ij} is the weight of input x_i to neuron j, b_j is the bias, and $f(z_j)$ is the activation function.

In the output layer, the final predicted value is calculated by weighted combination with the formula:

$$\hat{y} = f(\sum_{j=1}^{m} w_{oj} y_j + b_o) \tag{4}$$

where w_{oj} is the weight from the hidden layer neuron j to the output layer, b_o is the bias of the output layer, and f is the activation function of the output layer.

Through training, the ANN continuously adjusts the weights w_{ij} and bias b_j to optimise the model for achieving improved prediction performance on specific tasks.

3.2 Activation function

Activation functions play a critical role in enabling neural networks to model nonlinear relationships. In the absence of an activation function (as defined in Equation 3), a neural network reduces to a purely linear transformation of Equation 4.

Consequently, regardless of the number of layers, the network remains a linear model, rendering it incapable of addressing complex nonlinear tasks. Common activation functions are shown in Figure 4, including Sigmoid, Tanh, ReLU, and Leaky ReLU. The Sigmoid function

$$Sigmoid(x) = \frac{1}{1 + e^{-x}}$$

outputs values in the range of 0 to 1, making it well-suited for the output layer in binary classification tasks. However, its tendency to produce very small gradients for large positive or negative inputs often leads to the vanishing gradient problem, thereby hindering the network's learning capability. The Tanh function,

$$Tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

which maps inputs to the range of -1 to 1, offers improved gradient flow compared to Sigmoid, yet still suffers from vanishing gradients in deep neural architectures.

The ReLU (Rectified Linear Unit) activation function is currently the most commonly used activation function and is defined as:

$$ReLU = max(0, x)$$

The ReLU offers the advantages of computational simplicity and effective mitigation of the vanishing gradient problem, thereby facilitating the training of deep neural networks. However, ReLU is susceptible to the issue of ”neuron death”, wherein neurons become inactive by outputting zero for all negative input values. This inactivation prevents the affected neurons from contributing to learning, thereby impairing the overall representational capacity of the network. To address this limitation, the LeakyReLU activation function was introduced. Its formulation is given as follows:

$$LeakyReLU(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha x & \text{if } x \leq 0 \end{cases}$$

where α is a small positive constant (e.g., 0.1), the LeakyReLU function permits a non-zero gradient when the input is negative, thereby mitigating the problem of neuron death. This feature enhances the network's ability to extrapolate and improves learning efficiency. Compared to ReLU, LeakyReLU is better suited for deep networks, as it maintains gradient flow in the negative input domain, contributing to improved training stability and generalization performance.

In this study, LeakyReLU is selected as the activation function primarily due to its ability to effectively mitigate the neuron death issue associated with ReLU and to enhance the network's extrapolation capacity. When applied to complex deep neural architectures, LeakyReLU contributes to maintaining network stability and prevents the occurrence of inactive neurons during training, thereby improving both the model's performance and its generalization capability.

3.3 Normalisation techniques

Normalization techniques are widely employed in deep learning to improve training stability and convergence speed by addressing internal covariate shift. Two prominent methods are Batch Normalization (BN) and Layer Normalization (LayerNorm).

Batch Normalization operates across the mini-batch dimension, normalizing each feature independently over a batch of samples. For a feature vector $\mathbf{x} = \{x_1, x_2, \ldots, x_m\}$ across m samples, the mean and variance are computed as:

$$\begin{aligned} \mu_B &= \frac{1}{m}\sum_{i=1}^{m} x_i \\ \sigma_B^2 &= \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2 \end{aligned} \tag{5}$$

The normalized value and affine transformation are given by:

$$\begin{aligned} \hat{x}_i &= \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \\ y_i &= \gamma \hat{x}_i + \beta \end{aligned} \tag{6}$$

where, γ and β are learnable parameters, and ϵ is a small constant added for numerical stability.

In contrast, Layer Normalization normalizes across the features of each individual sample. Given an input vector $\mathbf{x} = (x_1, x_2, \ldots, x_H)$ with H features, the normalization statistics are calculated as:

$$\begin{aligned} \mu &= \frac{1}{H}\sum_{j=1}^{H} x_j \\ \sigma^2 &= \frac{1}{H}\sum_{j=1}^{H}(x_j - \mu)^2 \end{aligned} \tag{7}$$

The normalized feature and affine transformation are then:

$$\begin{aligned} \hat{x}_j &= \frac{x_j - \mu}{\sqrt{\sigma^2 + \epsilon}} \\ y_j &= \gamma \hat{x}_j + \beta \end{aligned} \tag{8}$$

While Batch Normalization is highly effective in convolutional and feedforward networks with sufficiently large batch sizes, it becomes less reliable in sequence models or when batch sizes are small or variable. Layer Normalization, by computing statistics per individual sample, is better suited for recurrent networks and transformer-based architectures. The primary distinction lies in the normalization axis: BN normalizes across *samples per feature*, whereas LayerNorm normalizes across *features per sample*.

In this work, the neural network adopts a fully connected architecture. While Batch Normalization (BN) has proven effective in deep networks, its performance can deteriorate when the batch size is small or inconsistent, due to the reliance on accurate batch-wise statistics. This limitation is particularly critical in fully connected settings where batch sizes may not be large enough to ensure stable estimates of the mean and variance. To address this issue, we employ Layer Normalization (LayerNorm), which computes normalization statistics across the feature dimension of each individual sample, making it independent of batch size. This property ensures consistent behavior during both training and inference, especially when deploying the model in settings where batch-wise variation is present. By normalizing activations per sample, LayerNorm effectively stabilizes the hidden state dynamics in fully connected layers, leading to improved convergence and generalization performance.

4 Results

4.1 Architecture of the model

The architecture illustrated by figure 5, employed in this study is a fully connected feedforward neural network composed of an input layer, two hidden layers, and an output layer. This structure is designed to model nonlinear dependencies in the input data while promoting numerical stability and efficient convergence.

The input layer accepts a vector of dimensionality d, which is fully connected to the first hidden layer via equation (3) of $n = N_1$. The output of this layer is passed through the LeakyReLU activation function. The

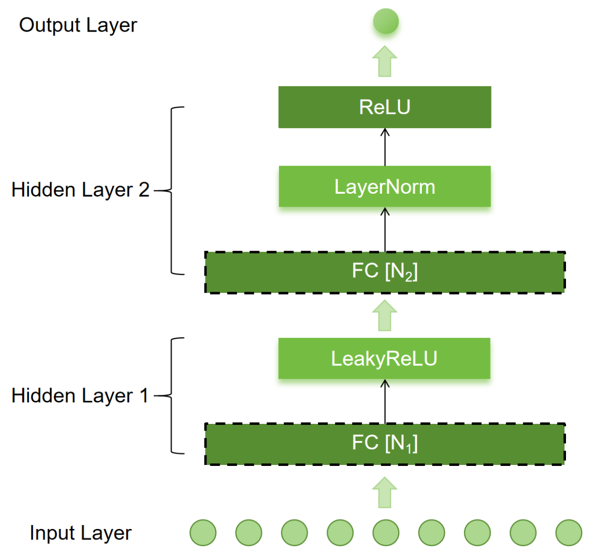

Figure 5. Two-layer neural network model architecture. The network architecture that contains two hidden layers is abbreviated as FC[N_1, N_2], and for the three-layer network architecture, the hidden layer 1 is directly copied as hidden layer 2, and the original hidden layer 2 becomes hidden layer 3, which is abbreviated as FC[N_1, N_2, N_3].

result is then propagated through the second fully connected layer of $n = N_2$. To enhance training dynamics and mitigate internal covariate shift, Layer Normalization is applied to the output of this layer. Following normalization, a ReLU activation function is employed. The final transformed representation is forwarded to the output layer, which consists of a single neuron for predicting the residual velocity of a projectile.

4.2 Evaluation of the ANN efficiency

In this work, two common metrics for evaluating regression performance were considered: the coefficient of determination (R²) and the mean squared error (MSE). Their respective mathematical definitions are as follows:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$
$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \quad (9)$$

where y_i and $\hat{y}_i$ denote the true and predicted values, $\bar{y}$ is the mean of the true values, and n is the number of samples.

The coefficient of determination (R²) was initially considered as a potential metric for evaluating model performance. However, it was observed that R² values consistently exceeded 0.99 across all tested ANN architectures and training conditions, regardless of input complexity or data size. While such high R² scores indicate strong overall correlations between predicted and target values, they offer limited sensitivity in distinguishing the performance differences among various model configurations. This insensitivity arises because R² is a relative measure of variance explained by the model and becomes saturated when residual errors are small in comparison to the total variance of the target variable.

In contrast, the mean squared error (MSE) provides an absolute measure of prediction error and is more sensitive to fine-grained differences in model performance. It directly reflects the magnitude of deviation between predicted and actual values, making it better suited for comparative analysis, especially in high-accuracy regimes where R² tends to plateau. Therefore, in this study, MSE is adopted as the primary evaluation metric for assessing the impact of model architecture and dataset size on prediction accuracy. The choice of MSE enables more nuanced assessment of model robustness, particularly in extrapolation tasks and scenarios involving increased input variability.

4.3 Performance of different model architectures

This subsection provides a comprehensive evaluation of different ANN model architectures, focusing on the effects of network depth, activation functions, normalization techniques, and training data size. By comparing multiple configurations under varying input conditions, the analysis aims to identify structural components that significantly influence model accuracy, efficiency, and generalization ability.

Table 1 presents a comprehensive performance comparison of various artificial neural network (ANN) architectures that incorporate both LeakyReLU activation and Layer Normalization (denoted as FC*) under different input parameter perturbation configurations: 20%, 20%-geo, 40%, and 40%-geo. Each architecture is defined by its specific layer width configuration, ranging from shallow models such as FC*[64,128] to deeper structures like FC*[256,512,256]. All models are evaluated on a fixed test set consisting of 1000 samples, with performance quantified using mean squared error (MSE) and coefficient of determination (R²).

The results show that while R² remains consistently high across all configurations—typically above 0.99 and therefore less discriminative—MSE provides more granular insight into performance differences. In particular, the architecture FC*[256,256] achieves the lowest MSE for the 20% and 40% settings, with values of 47.803 and 34.207, respectively. For the 20%-geo and 40%-geo configurations, the best-performing models are FC*[256,512], reaching MSE values of 44.921 and 34.177. These optimal models are highlighted in the table 1.

The findings indicate that moderately deep networks (e.g., two hidden layers with 256 or 512 neurons) strike a favorable balance between model complexity and predictive performance. Deeper or wider networks do not necessarily yield better accuracy and may suffer from overfitting or diminished generalization. This analysis confirms that careful architectural tuning—rather than

Table 1. Performance (MSE and R^2) of various ANN architectures with *LeakyReLU* and *LayerNorm* for different configurations of the impact problem. Optimal models are highlighted. For all configurations the test data subset has 1000 problems.

	Configuration							
	20%		**20% geo**		**40%**		**40% geo**	
	MSE	R²	MSE	R²	MSE	R²	MSE	R²
FC* [64,128]	53.724	0.9921	51.704	0.9936	46.139	0.9939	47.057	0.9941
FC* [128,64]	54.536	0.9919	52.629	0.9935	41.873	0.9943	48.932	0.9938
FC* [128,256]	51.278	0.9925	50.721	0.9937	36.521	0.9952	45.123	0.9940
FC* [256,128]	49.513	0.9927	48.329	0.9940	34.782	0.9955	43.279	0.9941
FC* [256,256]	**47.803**	**0.9930**	46.359	0.9942	**34.207**	**0.9955**	35.772	0.9955
FC* [256,512]	49.812	0.9928	**44.921**	**0.9944**	35.246	0.9950	**34.177**	**0.9957**
FC* [512,256]	52.556	0.9922	47.560	0.9941	35.914	0.9950	37.955	0.9949
FC* [64,64,64]	53.028	0.9922	61.472	0.9924	37.155	0.9949	42.816	0.9942
FC* [64,128,64]	54.028	0.9919	56.440	0.9930	37.352	0.9949	49.501	0.9935
FC* [128,64,128]	50.653	0.9925	55.249	0.9932	39.008	0.9946	38.704	0.9948
FC* [128,128,128]	57.067	0.9915	49.155	0.9939	44.998	0.9935	43.279	0.9941
FC* [128,256,128]	54.062	0.9920	50.802	0.9937	43.665	0.9940	40.888	0.9945
FC* [256,256,256]	59.342	0.9911	53.621	0.9933	47.112	0.9937	36.284	0.9953
FC* [256,512,256]	58.618	0.9912	53.760	0.9933	49.327	0.9932	39.461	0.9947

Table 2. Comparison experiments (MSE). The model structure is FC[256,256] for the 20% and 40% tests, and FC[256, 512] for the 20%-geo and 40%-geo tests, where FC denotes the absence of the LayerNorm and using only ReLU as the activation function. FC* is from the optimal model in Table 1, and denotes the addition of the LayerNorm with the LeakyReLU.

	Configuration			
	20%	**20% geo**	**40%**	**40% geo**
FC	59.90	61.78	45.98	54.09
FC+LeakyReLU	60.63	56.71	46.10	54.55
FC+LayerNorm	49.30	46.80	60.98	37.14
FC*	**47.80**	**44.92**	**34.20**	**34.17**

simply increasing model depth—is essential for optimizing performance in ANN-based surrogates, especially under complex or high-dimensional input conditions.

Table 2 presents a comprehensive set of comparison experiments designed to evaluate the impact of integrating LeakyReLU activation and Layer Normalization (LayerNorm) into a fully connected (FC) neural network architecture. The baseline model, referred to as FC, utilizes the standard ReLU activation function without any normalization layers. To systematically assess the contribution of each component, three extended configurations are considered: (1) FC+LeakyReLU, in which ReLU is replaced with LeakyReLU to alleviate the issue of neuron inactivation and improve gradient flow; (2) FC+LayerNorm, which incorporates layer normalization to enhance training stability and convergence behavior; and (3) FC*, the final proposed architecture that combines both LeakyReLU and LayerNorm. All models were tested under four input settings: 20%, 40%, 20%-geo, and 40%-geo. The performance of each configuration is evaluated using the mean squared error (MSE) metric. The results clearly indicate that the FC* model consistently achieves the lowest MSE across all scenarios, with especially notable reductions observed in the

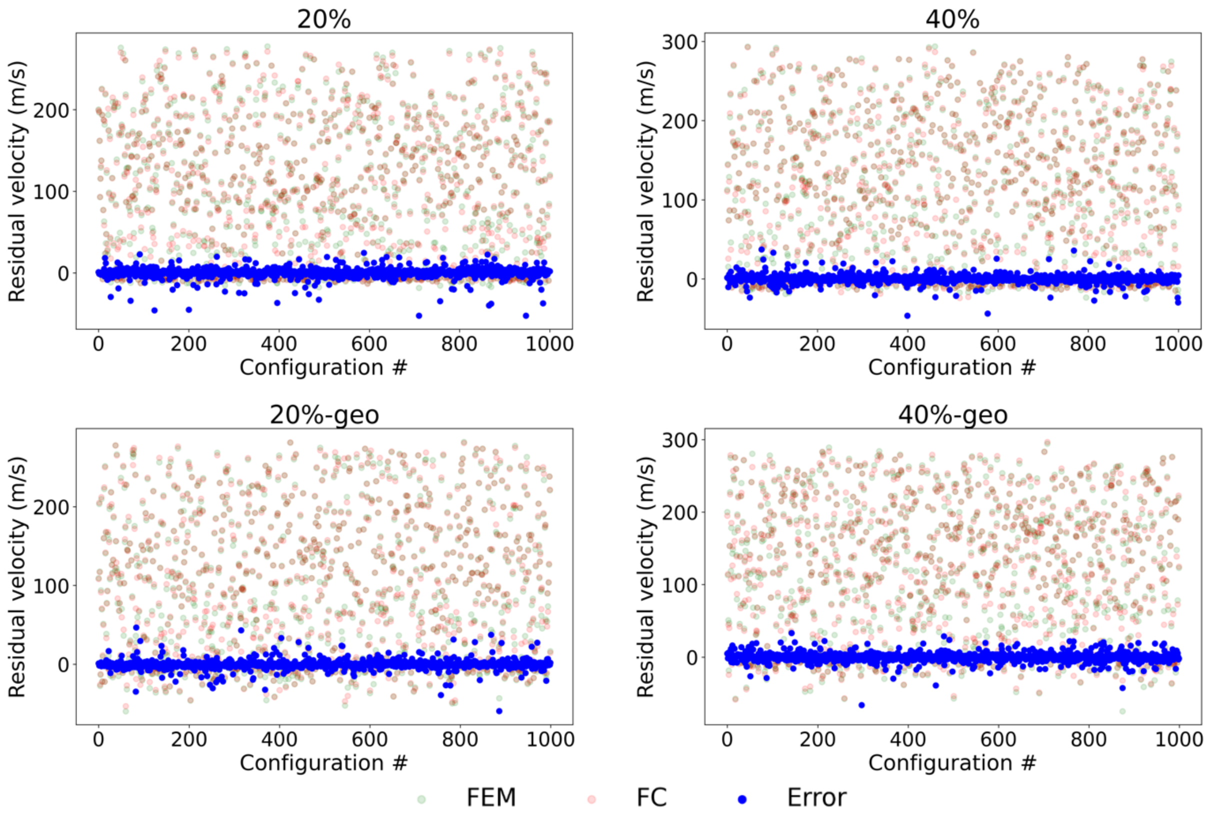

Figure 6. Accuracy of the ANN models developed for the impact problem, each dot indicates one problem from the test configuration set.

40% and 40%-geo configurations, where MSE values dropped to 34.20 and 34.17, respectively. These improvements highlight the synergistic effect of combining activation function refinement with normalization techniques, underscoring their critical role in improving model accuracy and generalization—particularly when handling high-dimensional input variability or extrapolation beyond the training domain.

Figure 6 illustrates the predictive accuracy of artificial neural network (ANN) surrogate models developed for the impact problem under varying degrees of input parameter perturbations. Each subplot represents a specific testing condition, including 20% and 40% variations in baseline parameters, as well as settings incorporating geometric perturbations (denoted as ”20%-geo” and ”40%-geo”). In each case, the residual velocities obtained from finite element method (FEM) simulations (green dots) are compared with the predictions of a fully connected ANN model (FC, red dots). The prediction errors, calculated as the difference between FEM and ANN outputs, are marked in blue. The ANN models show a strong ability to reproduce the general distribution of residual velocities across diverse configuration spaces. Notably, the prediction errors remain relatively small and are mostly concentrated in lower velocity regimes, indicating the model's stability even under complex, high-variability input conditions. These results highlight the potential of ANN-based surrogates for efficient and reliable approximations of computationally expensive impact simulations.

Figure 7 further evaluates the impact of training dataset size on the performance of the ANN models. For the 20% and 40% test settings, a network architecture (FC[256, 256]) is used, whereas a deeper structure (FC[256, 512]) is adopted for the 20%-geo and 40%-geo cases to account for the increased complexity due to geometric variability. The results show a clear trend: as the amount of training data decreases, the mean squared error (MSE) increases in all settings. This degradation is particularly noticeable in the geo-perturbed cases, indicating that more training samples are required to generalize effectively when the input space includes structural or geometric variations. These findings highlight the sensitivity of ANN performance to both dataset size and input complexity, reinforcing the need for careful model scaling in data-limited applications.

4.4 Extrapolation capability of ANN models

The figure 8 depicts a finite element method (FEM) simulation result illustrating the failure occurring at high impactor velocities (382 m/s). The model consists of two distinct parts: the red-colored impactor and the blue-colored target, both represented with mesh elements. The enlarged view emphasizes the region experiencing failure, highlighting unstable contact conditions and ex-

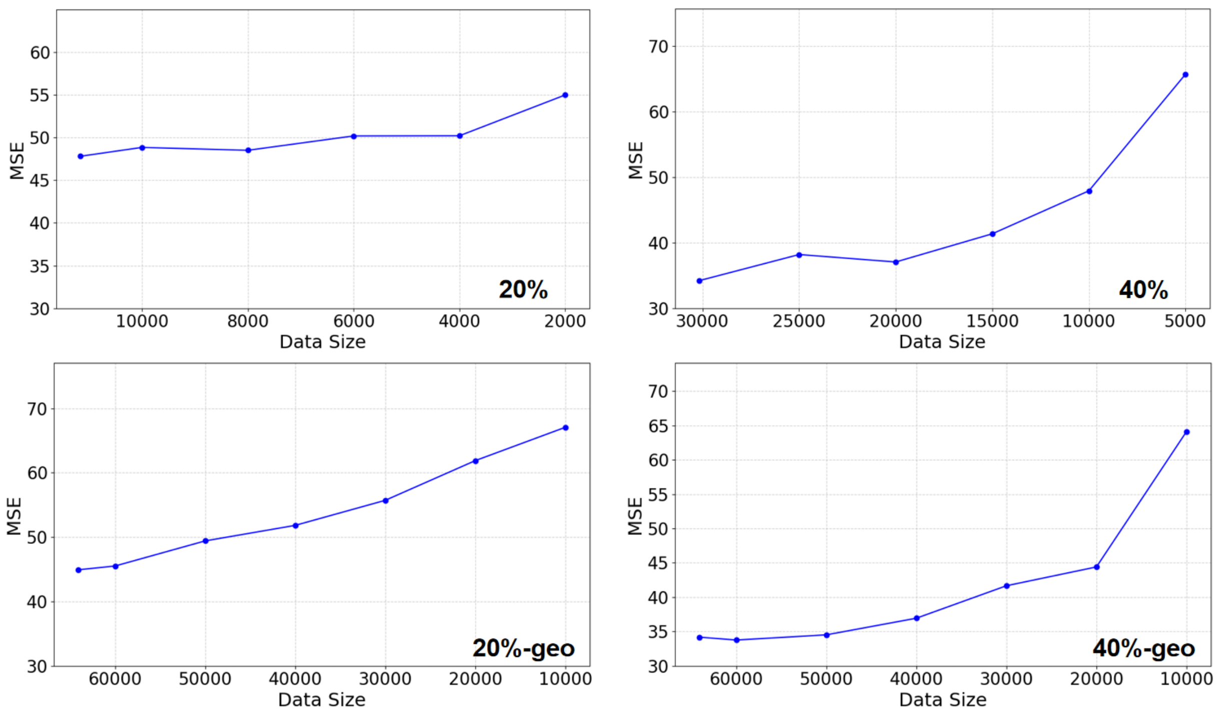

Figure 7. Efficiency (MSE) of ANN model depending on the dataset size for the impact problem.

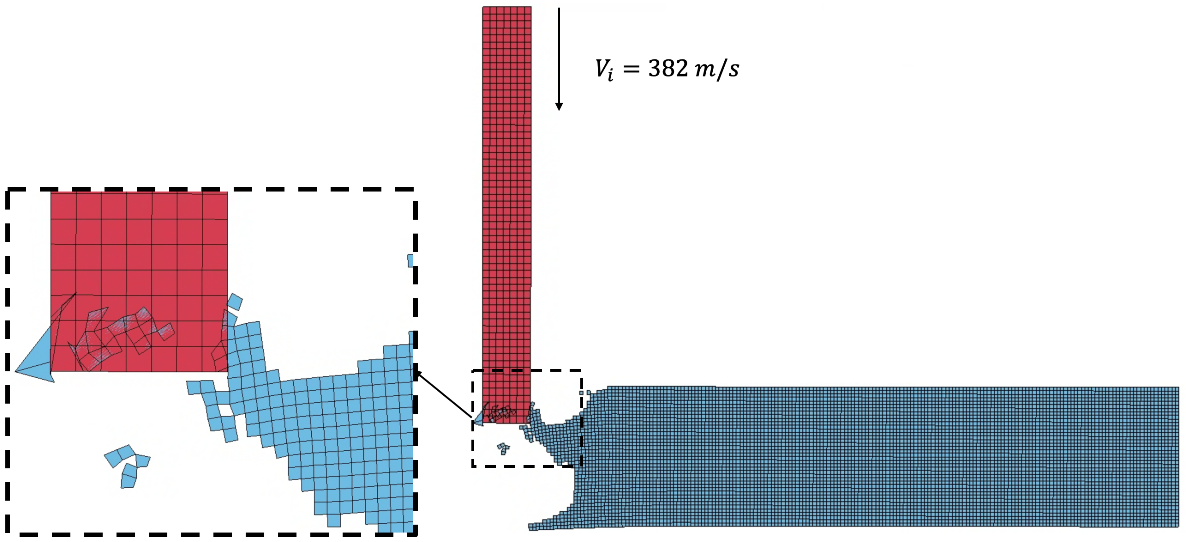

Figure 8. FEM failure for higher impactor velocities: unstable contact and excessive element deformation.

cessive deformation, as evidenced by significantly distorted and fragmented mesh elements. This figure visually illustrates the limitation of FEM under high-velocity impacts, specifically demonstrating the instability and numerical errors resulting from extreme element distortion.

ANN can serve as efficient alternative models for predicting residual velocities in high-speed impact scenarios, significantly reducing computational expense compared to traditional finite element methods (FEM). By training the NN on a comprehensive dataset generated from detailed FEM simulations or experimental data, the model can learn complex nonlinear relationships inherent in the impact mechanics. Once adequately trained, neural networks rapidly approximate residual velocities for new scenarios without performing computationally expensive iterative calculations.

Figure 9 demonstrates the extrapolation capability of the trained artificial neural network (ANN) models in predicting residual impactor velocities beyond the train-

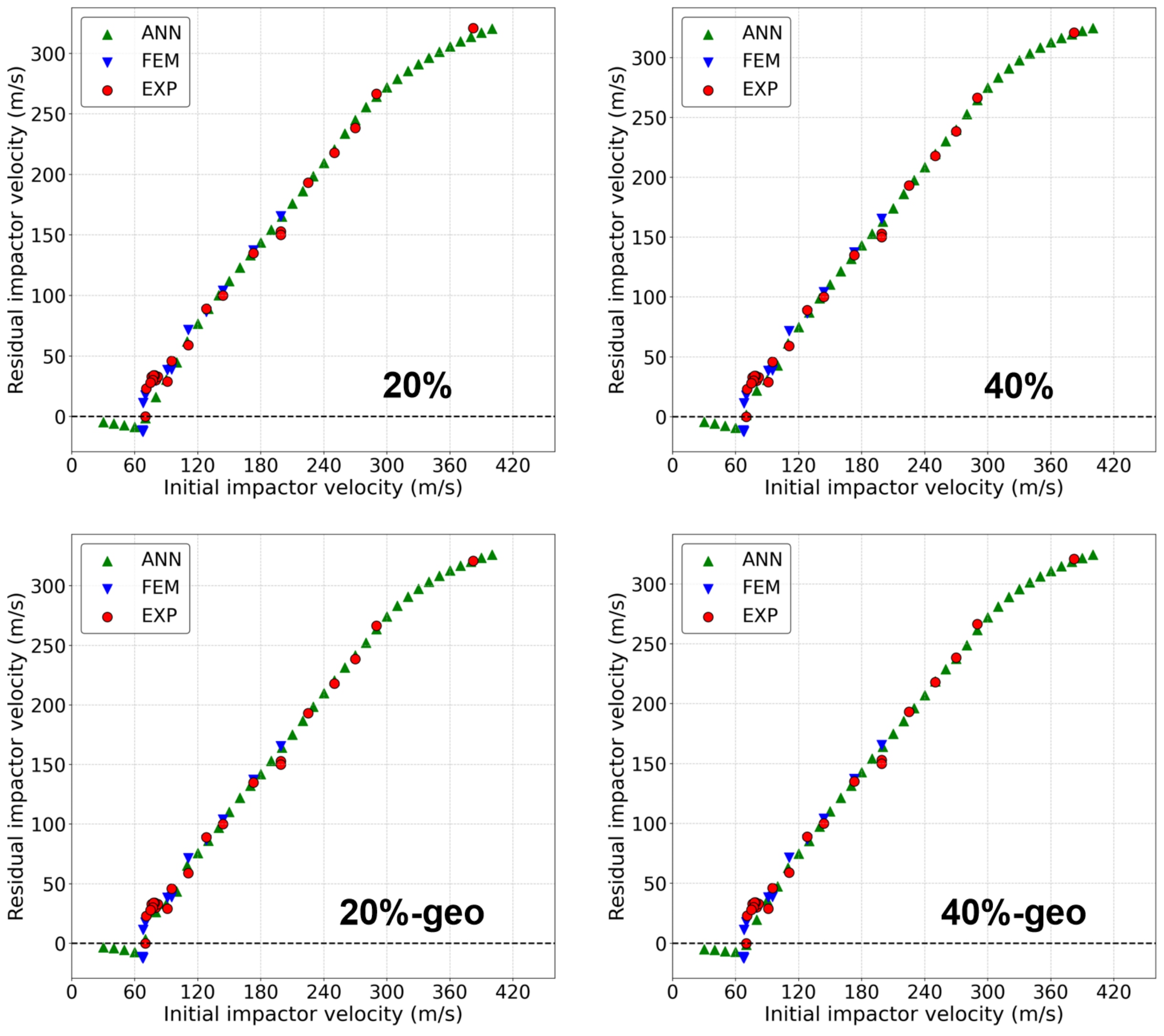

Figure 9. Demonstration of the extrapolation capability of the ANN model. The topmost red dot is for the problem with an initial velocity of 382 m/s and its experimental residual velocity of 321 m/s. This initial velocity lies outside the data distribution in the training set.

ing data range. In this test, the ANN models are evaluated on input cases with initial impactor velocities extending up to 400 m/s, while the training dataset only covers the range [30, 300] m/s. For the 20% and 40% test cases, the network architecture FC[256, 256] is employed, and for the 20%-geo and 40%-geo cases, a deeper structure FC[256, 512] is used to accommodate geometric variability. The figure compares the ANN predictions (green triangles) with corresponding finite element method (FEM) results (blue triangles) and experimental data (red circles).

Of particular interest is the prediction at an initial impactor velocity of 382 m/s, which lies well beyond the training distribution. The ANN model successfully predicts a residual velocity that closely matches the experimental measurement of 321 m/s, demonstrating not only numerical accuracy but also physical consistency. This result underscores the model's ability to generalize to unseen high-velocity impact conditions and suggests that, when appropriately trained, ANN surrogates can serve as reliable predictive tools even outside the interpolation regime.

5 Conclusion

In summary, this study demonstrates that carefully designed artificial neural network (ANN) architectures can serve as effective surrogate models for complex impact problems. The results reveal that while common performance metrics such as R^2 may become saturated in high-accuracy regimes, MSE provides a more sensitive and informative measure for comparing different model structures. Experiments show that the incorporation of LeakyReLU and Layer Normalization significantly enhances model performance, especially in high-variability and geometrically complex settings. Moderate-depth architectures (e.g., FC*[256, 256] and FC*[256, 512]) achieve optimal accuracy across most configurations, while further increases in depth or width offer diminishing returns. Furthermore, the models exhibit strong generalization capability, including accurate extrapolation beyond the training domain—successfully predicting outcomes for unseen high-velocity scenarios. These findings highlight the importance of architectural tuning and regularization in developing robust and data-efficient ANN surrogates for high-dimensional, nonlinear physical systems.

Acknowledgements

The work has been supported by RSF (grant 22-71-10019).

References

Alli, Y. A., Anuar, H., Manshor, M. R., Okafor, C. E., Kamarulzaman, A. F., Akçakale, N., Nazeri, F. N. M., Bodaghi, M., Suhr, J., and Nasir, N. A. M. (2024). Optimization of 4d/3d printing via machine learning: A systematic review. *Hybrid Advances*, p. 100242.

Bacanin, N., Zivkovic, M., Al-Turjman, F., Venkatachalam, K., Trojovskỳ, P., Strumberger, I., and Bezdan, T. (2022). Hybridized sine cosine algorithm with convolutional neural networks dropout regularization application. *Scientific Reports*, **12** (1), pp. 6302.

Bejani, M. M. and Ghatee, M. (2021). A systematic review on overfitting control in shallow and deep neural networks. *Artificial Intelligence Review*, **54** (8), pp. 6391–6438.

Brown, J., Zimny, J., and Radko, T. (2022). Identifying the origin of turbulence using convolutional neural networks. *Fluids*, **7** (7), pp. 239.

Cadoni, E., Dotta, M., Forni, D., Tesio, N., and Albertini, C. (2013). Mechanical behaviour of quenched and self-tempered reinforcing steel in tension under high strain rate. *Materials & Design*, **49**, pp. 657–666.

Cao, Y., Chen, Z., Belkin, M., and Gu, Q. (2022). Benign overfitting in two-layer convolutional neural networks. *Advances in neural information processing systems*, **35**, pp. 25237–25250.

Cheng, L., Guo, H., Sun, L., Yang, C., Sun, F., and Li, J. (2024). Real-time simulation of tube hydroforming by integrating finite-element method and machine learning. *Journal of Manufacturing and Materials Processing*, **8** (4), pp. 175.

Choudhary, K., DeCost, B., Chen, C., Jain, A., Tavazza, F., Cohn, R., Park, C. W., Choudhary, A., Agrawal, A., Billinge, S. J., et al. (2022). Recent advances and applications of deep learning methods in materials science. *npj Computational Materials*, **8** (1), pp. 59.

Duran, B., Emory, D., Azam, Y. E., and Linzell, D. G. (2025). A novel cnn architecture for robust structural damage identification via strain measurements and its validation via full-scale experiments. *Measurement*, **239**, pp. 115393.

Eischens, R., Li, T., Vogl, G. W., Cai, Y., and Qu, Y. (2025). State space neural network with nonlinear physics for mechanical system modeling. *Reliability Engineering & System Safety*, **259**, pp. 110946.

Ejaz, F., Hwang, L. K., Son, J., Kim, J.-S., Lee, D. S., and Kwon, B. (2022). Convolutional neural networks for approximating electrical and thermal conductivities of cu-cnt composites. *Scientific reports*, **12** (1), pp. 13614.

Evstifeev, A., Kazarinov, N., Petrov, Y., Witek, L., and Bednarz, A. (2018). Experimental and theoretical analysis of solid particle erosion of a steel compressor blade based on incubation time concept. *Engineering Failure Analysis*, **87**, pp. 15–21.

Gao, Z., Zhu, C., Wang, C., Shu, Y., Liu, S., Miao, J., and Yang, L. (2025). Advanced deep learning framework for multi-scale prediction of mechanical properties from microstructural features in polycrystalline materials. *Computer Methods in Applied Mechanics and Engineering*, **438**, pp. 117844.

Gromov, N. V., Smirnov, L. A., and Levanova, T. A. (2024). Prediction of extreme events and chaotic dynamics using wavenet. *CYBERNETICS AND PHYSICS*, **13** (1), pp. 20–31.

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

Hu, H., Qi, L., and Chao, X. (2024). Physics-informed neural networks (pinn) for computational solid mechanics: Numerical frameworks and applications. *Thin-Walled Structures*, p. 112495.

Huang, Z., Xiong, X., Zheng, S., and Ma, H. (2025). Integration of finite element method and neural network for enhanced prediction of rubber buffer stiffness in light aircraft. *Aerospace*, **12** (3), pp. 253.

Ishtiaq, M., Tiwari, S., Nagamani, M., Kang, S.-G., and Reddy, N. G. S. (2025). Data-driven ann-based predictive modeling of mechanical properties of 5cr-0.5 mo steel: Impact of composition and service temperature. *Crystals*, **15** (3), pp. 213.

Istomin, V. A., Kustova, E. V., Lagutin, S. M., and Shalamov, I. Y. (2023). Evaluation of state-specific transport properties using machine learning methods. *Cybern. Phys*, **12** (1), pp. 34–41.

Istomin, V. A. and Pavlov, S. A. (2024). Suitability of different machine learning methods for high-speed flow modelling issues. *CYBERNETICS AND PHYSICS :* , **12** (4), pp. 264–274.

Johnson, G. R. and Holmquist, T. J. (1994). An improved computational constitutive model for brittle materials. In *AIP conference proceedings*, vol. 309, American Institute of Physics, pp. 981–984.

Jokar, M. and Semperlotti, F. (2021). Finite element network analysis: A machine learning based computational framework for the simulation of physical systems. *Computers & Structures*, **247**, pp. 106484.

Kalthoff, J. and Shockey, D. (1977). Instability of cracks under impulse loads. *Journal of Applied Physics*, **48** (3), pp. 986–993.

Kamalov, F., Sayari, Z., Gheisari, M., Lee, C.-C., and Moussa, S. (2023). Routing algorithms in internet of things complex network with the role of machine learning. *Cybernetics and Physics*, **12** (3), pp. 182–193.

Kazarinov, N., Bratov, V., Morozov, N., Petrov, Y., Balandin, V., Iqbal, M. A., and Gupta, N. (2020). Experimental and numerical analysis of pmma impact fracture. *International Journal of Impact Engineering*, **143**, pp. 103597.

Kazarinov, N. and Khvorov, A. (2024). Predicting impact strength of perforated targets using artificial neural networks trained on fem-generated datasets. *Defence Technology*, **32**, pp. 32–44.

Kazarinov, N., Petrov, Y., and Cherkasov, A. (2021). Instability effects of the dynamic crack propagation process. *Engineering Fracture Mechanics*, **242**, pp. 107438.

Knyazev, N., Pershin, A., Golovkina, A., and Kozynchenko, V. (2023). Forecasting the state of complex network systems using machine learning methods. *Cybern. Phys*, **12** (2), pp. 129–135.

Kononenko, O. and Kononenko, I. (2018). Machine learning and finite element method for physical systems modeling. *arXiv preprint arXiv:1801.07337*.

Le Viet, H., Ngoc, H. L. H., Minh, K. T. D., and Van Hong, S. T. (2024). A deep learning framework for gym-gesture recognition using the combination of transformer and 3d pose estimation. *Cybern. Phys*, **13** (2), pp. 161–167.

Leng, J.-x., Wang, Z.-g., Huang, W., Shen, Y., and An, K. (2024). Multidisciplinary design optimization processes for efficiency improvement of aircraft: State-of-the-art review. *International Journal of Aeronautical and Space Sciences*, pp. 1–23.

Li, S. and Mao, X. (2024). Training all-mechanical neural networks for task learning through in situ backpropagation. *Nature Communications*, **15** (1), pp. 1–12.

Mendizabal, A., Márquez-Neila, P., and Cotin, S. (2020). Simulation of hyperelastic materials in real-time using deep learning. *Medical image analysis*, **59**, pp. 101569.

Mianroodi, J. R., H. Siboni, N., and Raabe, D. (2021). Teaching solid mechanics to artificial intelligence—a fast solver for heterogeneous materials. *Npj Computational Materials*, **7** (1), pp. 99.

Mikhailova, N. V., Volkov, G. A., Meshcheryakov, Y. I., Petrov, Y. V., and Utkin, A. A. (2017). Failure-delay effect in destruction of steel samples under spalling conditions. *Technical Physics*, **62**, pp. 547–552.

Mitusch, S. K., Funke, S. W., and Kuchta, M. (2021). Hybrid fem-nn models: Combining artificial neural networks with the finite element method. *Journal of Computational Physics*, **446**, pp. 110651.

Petrov, Y. V. (1991). On “quantum” nature of dynamic failure of brittle media. In *Doklady Akademii Nauk*, vol. 321, Russian Academy of Sciences, pp. 66–68.

Petrov, Y. V. and Utkin, A. (1989). Dependence of the dynamic strength on loading rate. *Soviet materials science: a transl. of Fiziko-khimicheskaya mekhanika materialov/Academy of Sciences of the Ukrainian SSR*, **25** (2), pp. 153–156.

Portal-Porras, K., Fernandez-Gamiz, U., Ugarte-Anero, A., Zulueta, E., and Zulueta, A. (2021). Alternative artificial neural network structures for turbulent flow velocity field prediction. *Mathematics*, **9** (16), pp. 1939.

Rana, P., Weigand, T. M., Pilkiewicz, K. R., and Mayo, M. L. (2024). A scalable convolutional neural network approach to fluid flow prediction in complex environments. *Scientific Reports*, **14** (1), pp. 23080.

Rao, C. and Liu, Y. (2020). Three-dimensional convolutional neural network (3d-cnn) for heterogeneous material homogenization. *Computational Materials Science*, **184**, pp. 109850.

Ravi-Chandar, K. and Knauss, W. G. (1984). An experimental investigation into dynamic fracture: I. crack initiation and arrest. *International Journal of Fracture*, **25**, pp. 247–262.

Rojek, I., Mikołajewski, D., Kotlarz, P., Tyburek, K., Kopowski, J., and Dostatni, E. (2021). Traditional artificial neural networks versus deep learning in optimization of material aspects of 3d printing. *Materials*, **14** (24), pp. 7625.

Rosakis, A. J. and Ravichandran, G. (2000). Dynamic failure mechanics. *International journal of solids and structures*, **37** (1-2), pp. 331–348.

Santos, C. F. G. D. and Papa, J. P. (2022). Avoiding overfitting: A survey on regularization methods for convolutional neural networks. *ACM Computing Surveys (Csur)*, **54** (10s), pp. 1–25.

Selyutina, N. and Petrov, Y. (2022). Instabilities of dynamic strain diagrams predicted by the relaxation model of plasticity. *Journal of Dynamic Behavior of Materials*, **8** (2), pp. 304–315.

Shafiq, M. and Gu, Z. (2022). Deep residual learning for image recognition: A survey. *Applied sciences*, **12** (18), pp. 8972.

Sharma, P., Chung, W. T., Akoush, B., and Ihme, M. (2023). A review of physics-informed machine learning in fluid mechanics. *Energies*, **16** (5), pp. 2343.

Shockey, D. A., Erlich, D., Kalthoff, J., and Homma, H. (1986). Short-pulse fracture mechanics. *Engineering Fracture Mechanics*, **23** (1), pp. 311–319.

Shokrollahi, Y., Nikahd, M. M., Gholami, K., and Azamirad, G. (2023). Deep learning techniques for predicting stress fields in composite materials: A superior alternative to finite element analysis. *Journal of Composites Science*, **7** (8), pp. 311.

Singh, V., Harursampath, D., Dhawan, S., Sahni, M., Saxena, S., and Mallick, R. (2024). Physics-informed neural network for solving a one-dimensional solid mechanics problem. *Modelling*, **5** (4), pp. 1532–1549.

Smirnov, I., Kazarinov, N., and Petrov, Y. (2019). Experimental observation and numerical modelling of unstable behaviour of a fast crack velocity. *Theoretical and Applied Fracture Mechanics*, **101**, pp. 53–58.

Souayeh, B., Bhattacharyya, S., Hdhiri, N., and Waqas Alam, M. (2021). Heat and fluid flow analysis and ann-based prediction of a novel spring corrugated tape. *Sustainability*, **13** (6), pp. 3023.

Tabian, I., Fu, H., and Sharif Khodaei, Z. (2019). A convolutional neural network for impact detection and characterization of complex composite structures. *Sensors*, **19** (22), pp. 4933.

Tarasova, E. and Moseiko, E. (2025). Decentralized spsa-based correction of task time predictions in adaptive multi-agent systems. *Cybern. Phys*, **14** (1), pp. 67–73.

Tariq, R., Abatal, M., Vargas, J., and Vázquez-Sánchez, A. Y. (2024). Deep learning artificial neural network framework to optimize the adsorption capacity of 3-nitrophenol using carbonaceous material obtained from biomass waste. *Scientific Reports*, **14** (1), pp. 20250.

Wen, L., Wang, Y., and Li, X. (2022). A new automatic convolutional neural network based on deep reinforcement learning for fault diagnosis. *Frontiers of Mechanical Engineering*, **17** (2), pp. 17.

Wu, J., Yang, C., Zhang, H., and Wang, L. (2025). The cross-scale interactive simulation and artificial neural network failure prediction of temperature field on wet friction interface. *IEEE Transactions on Instrumentation and Measurement*.

Zhang, Y., Zhao, S., Kazarinov, N., and Petrov, Y. V. (2024). Neural networks with iterative parameter generation for determining parameters of constitutive models. *Cybernetics and physics*, **13** (4), pp. 334–345.

Zhao, S., Petrov, Y. V., Zhang, Y., Volkov, G., Xu, Z., and Huang, F. (2024a). Modeling of the thermal softening of metals under impact loads and their temperature–time correspondence. *International Journal of Engineering Science*, **194**, pp. 103969.

Zhao, Y., Li, H., Zhou, H., Attar, H. R., Pfaff, T., and Li, N. (2024b). A review of graph neural network applications in mechanics-related domains. *Artificial Intelligence Review*, **57** (11), pp. 315.

Zheng, B., Moni, A., Yao, W., and Xu, M. (2025). Manifold learning for aerodynamic shape design optimization. *Aerospace*, **12** (3), pp. 258.