Digital Engineering and Digital Twin

Tech Science Press

DOI: 10.32604/dedt.2024.044545

ARTICLE

Supervised Learning for Finite Element Analysis of Holes under Biaxial Load

Wai Tuck Chow* **and Jia Tai Lau**

School of Mechanical and Aerospace Engineering, Nanyang Technological University, 639798, Singapore

*Corresponding Author: Wai Tuck Chow. Email: wtchow@ntu.edu.sg

Received: 02 August 2023 Accepted: 07 March 2024 Published: 06 May 2024

ABSTRACT

This paper presents a novel approach to using supervised learning with a shallow neural network to increase the efficiency of the finite element analysis of holes under biaxial load. With this approach, the number of elements in the finite element analysis can be reduced while maintaining good accuracy. The neural network will be used to predict the maximum stress for holes of different configurations such as holes in a finite-width plate (2D), multiple holes (2D), staggered holes (2D), and holes in an infinite plate (3D). The predictions are based on their respective coarse mesh with only 2 elements along the whole quarter perimeter. The result shows the prediction errors are under 5% for all the listed hole configurations. In contrast, the conventional FEM with the respective coarse mesh has errors above 20%. To achieve similar accuracy, the conventional FEM would require finer mesh with at least 6 elements along the perimeter. Furthermore, this setup is also effective in predicting the maximum stress for the 3D problem. With the aid of supervised learning, the FEM analysis of a coarse mesh with only 2 elements along the quarter perimeter can attain prediction errors of less than 2%. For the same coarse mesh, the conventional FEM has errors above 35%. To achieve similar accuracy for the 3D problem, the conventional FEM would require finer mesh with more than 8 elements along the perimeter. This result shows that supervised learning has great potential to enhance the efficiency of finite element analysis with fewer elements while attaining satisfactory results.

KEYWORDS

Supervised learning; finite element analysis; stress concentration

1 Introduction

The use of finite element analysis (FEA) to solve problems of engineering is popular in many engineering industries due to its ability to represent complex geometry and capture local stress concentration effects. Furthermore, due to the increasing computational power available and the decreasing costs of FEA software, the use of FEA has grown significantly over the years. Yet, modeling a large segment of a component can still require high computational costs. To keep the computational costs practical, coarser mesh and linear elements are often used. As a result, the accuracy of the model is poor, and it is insufficient for fatigue analysis to be carried out. To achieve the accuracy needed, the sub-modeling of the FEA model is performed in high-stress areas. Sub-modeling is the process of creating a solid model of the local geometry at the region of interest within the global model. Subsequently, the local solid model must undergo mesh refinement together with a transfer of boundary conditions from the global model. Unfortunately, sub-modeling is usually performed manually. Therefore, sub-modeling often is a labor-intensive process especially when the part has many features that require many sub-models. As a result, performing these complex 3-D design iterations with local geometry changes or topology optimizations often requires significant time and resources. Moreover, even with sub-modeling, the choice of 3D elements is often linear due to computational limitations. Hence, a huge amount of effort and resources can be saved if machine learning techniques could be applied to increase the accuracy of the finite element analysis of coarse 3D mesh and reduce the need for sub-modeling.

Machine learning is the usage of models to learn from data and make assessments without being explicitly programmed. The goal is to create a trained neural network model based on the pattern of the training data and be able to make predictions involving different sets of data with similar patterns. Improvement in machine learning techniques over the years has made it an effective tool to be used across many different industries such as finance, social research, and medicine. Improvements can be categorized into two areas, software and hardware. For instance, the algorithm using the Scaled Conjugate Gradient (SCG) is a more robust variant of the original conjugate gradient method (CG). It is less computationally complex and faster. SCG is used as a backpropagation algorithm to minimize errors in the neural network. Moller [1] demonstrated that SCG can converge to a minimum point where the CG cannot due to the Hessian matrix for global error function not being positive definite. In addition, SCG does not include any user-dependent parameters that are crucial for the success of the algorithm, unlike CG. Furthermore, he showed that the calculation complexity of one iteration of SCG can be more than 3 times less than that of the CG method. Meanwhile, the Levenberg-Marquardt method (LM) is developed to solve non-linear least squares problems. Gavin et al. [2–4] demonstrated that LM is more robust than the Gauss-Newton algorithm in minimization tasks where it can converge to the solution even when the starting point is very far off from the minimum. At the same time, it also converges to the minimum point faster than the gradient descent method. Finally, regularization techniques have been developed to curb the problem of overfitting the neural network (NN) to training data sets. One such method is the Bayesian Regularization (BR). Kayri [5] demonstrated that the BR algorithm with tangent sigmoid transfer function in the framework of a single hidden layer feed-forward neural network is effective in predicting nonlinear relations and non-continuous data such as social data. He also showed that using 2 neurons in such a framework has the highest reliability and robustness.

As for hardware advancements, neural processing units (NPU) are increasingly adopted for machine learning tasks which are traditionally performed by the central processing unit (CPU) or graphics processing units (GPU). NPUs are dedicated processing units optimized for power and area efficiency for matrix math. For instance, the 700 MHz NPU used by Google, known as a Tensor Processing Unit (TPU) has a matrix multiplication unit with over 65,000 arithmetic logic units and can process 92 trillion 8-bit operations per second [6]. In addition, Norman et al. [6] expected that rapid improvements will be made in NPU due to its design simplicity. Hence, with the rapid advancement in machine learning algorithms and NPU hardware, there seems to be a huge potential in integrating machine learning with finite element analysis.

Hashash et al. [7] utilized neural network to define the non-linear material constitutive models in the FEA analysis. Manevitz et al. [8] used neural network time series methodology to determine the location for finite element mesh refinement at optimum interval. Chow [9] demonstrated that by using supervised learning algorithms, the errors of coarse mesh can be significantly reduced for holes on a finite-width plate under a uniaxial load. However, more study is required to evaluate if supervised learning can be effectively applied to a more diverse set of problems.

The main objective of this paper is to provide the optimal approach to effectively utilize supervised learning to optimize the FEA of holes under biaxial load in a diverse set of problems. This study also evaluates the extent of reduction in errors by using supervised learning. In this paper, SCG, LM, and BR algorithms were used to train the NN. In addition, pure linear and tangent sigmoid transfer functions will be evaluated. The displacement nodal solutions of the nearest 6 nodes to the quarter-hole's edge were used as the training parameters. The input data set used to train the NN will consist of a small set of 20 different course mesh of a hole in an infinite-width plate. There will only be 2 elements along the quarter-hole perimeter in the coarse mesh. An analytical solution of a hole under biaxial load will be used as the output target data. Subsequently, the NN will be used to predict the maximum stress of holes under a more diverse range of problems such as holes on a finite-width plate, multiple holes, staggered holes, and holes on the 3D plate.

2 Neural Networks

An artificial neural network consists of neurons, weights, and biases. Using an iterative minimization procedure based on the different back-propagation techniques, the weights are adjusted to reduce the error [10,11]. In addition to weights, biases are included in a neural network to introduce flexibility and better generalization to the neural network. Bias is an extra input to neurons that may allow the neuron to be activated even when inputs are 0. The inclusion of biases is essential in neural networks to prevent overfitting to the training data set [12,13].

Many studies have been done to optimize the weight initializations to curb problems of vanishing gradients and exploding gradients [14,15]. These problems could cause weight updates that are overly small or large which stops the backpropagation from working effectively. However, they are only prevalent in multi-layered neural networks or recurrent neural networks with gradient-based propagation methods. Fortunately, this study uses only one hidden layer in a feed-forward neural network. Hence weight initialization will not be focused on mitigating these problems.

Weight initialization will instead be focused on reducing training time. The Nguyen-Widrow algorithm will be used to initialize the weights and biases for all 3 backpropagation methods. This eliminates most of the weight adjustments, hence only small adjustments are needed to be made during training. The Nguyen-Widrow algorithm has been demonstrated to be able to accelerate the training process and testing accuracy [16–19]. Since the Nguyen-Widrow method requires the layer's transfer function to have a finite input range, it will only be used when the layer has a tangent sigmoid transfer function. For layers with a pure linear transfer function, weights and biases will be initialized randomly with values between -1 and 1.

The following three sections present the three different back-propagation algorithms used in the study. These three backpropagation methods were chosen as they are the most common methods used in MATLAB machine learning.

2.1 Scaled Conjugate Gradient (SCG)

The Conjugate Gradient (CG) method is implemented as an iterative algorithm in the minimization problem. It can be regarded as being between the gradient descent algorithm and Newton's method. The gradient descent algorithm requires the calculation of first order derivative which is the gradient to find the minimum of the cost function. It has a slow convergence rate where subsequent steps often contradict previous steps. On the other hand, Newton's method requires the calculation of second order derivative which has high computational cost. CG method accelerates the convergence rate of the steepest descent by performing the search along conjugate directions such that subsequent steps never contradict previous steps. It also has lower computational costs compared to Newton's method. The Scaled Conjugate Gradient (SCG) method which is faster and more robust will be used [1].

Consider a quadratic function:

$$f(x) = \frac{1}{2}x^T Ax - x^T b \tag{1}$$

where A is a symmetric positive-definite matrix. Eq. (1) can model the cost function at a given point expressed by Taylor expression, where x will be the weight vector. The minimization is an iterative process in which a local minimum in the neighborhood of the initial point x_0 is found through subsequent steps of step size a_k and step direction p_k. The residual r_k is the negative gradient of $f(x)$ at point x_k. The iteration only stops when the corresponding residual is sufficiently small, signifying the minimum point where the gradient is 0.

The algorithm is started by choosing the initial weight vector x_0, then setting the initial residual $r_0 = b - Ax_0$ and initial step direction $p_0 = r_0$. The step size at k^{th} iteration a_k is computed as follows:

$$a_k = \left(\frac{{r_{k-1}}^T r_{k-1}}{{p_{k-1}}^T A p_{k-1}}\right) \tag{2}$$

The weight vector and residual are then updated:

$$x_{k+1} = x_k + a_k p_k \tag{3}$$

$$r_{k+1} = b - Ax_{k+1} = b - A(x_k + a_k p_k) = r_k - a_k A p_k \tag{4}$$

The direction vector of the next step, p_{k+1} is determined by the residual r_{k+1}, then subtracting away any vector that would counter-act the previous direction:

$$p_{k+1} = r_{k+1} - \frac{{r_{k+1}}^T A p_k}{{p_k}^T A p_k} p_k \tag{5}$$

such that p_{k+1} is conjugated to p_k. Subsequently, the weight vector for the next iteration can be obtained like in Eq. (3).

CG method will fail and converge to nonstationary point if matrix A is not positive definite. This problem can be resolved by introducing a scalar λ_k in CG. This is known as SCG method where the term $A \cdot p_k$ is estimated as shown in the following equation:

$$A \cdot p_k = f''(x_k) \cdot p_k \approx \frac{f'(x_k + \sigma_k p_k) - f'(x_k)}{\sigma_k} + \lambda_k p_k \tag{6}$$

where σ_k and λ_k are adjusted at each iteration under the condition that $0 < \sigma_k \leq 10^{-4}$ and $0 \leq \lambda_k \leq 10^{-6}$. The initial value λ_0 is 0 and is adjusted at each iteration looking at the sign of Ap_k. This ensures that matrix A is always positive definite such that a unique minimizer exists. As the scaling factor, λ_k is raised to keep ${p_k}^T A p_k$ positive, the step size a_k decreases. Lundén et al. [20] demonstrated that this technique reduces the error significantly in a minimization problem.

2.2 Levenberg-Marquardt (LM)

Similar to SCG, the Levenberg-Marquardt (LM) method is an iterative process such that the final weight is obtained through iterations with step δ. LM method uses a damping factor λ to decide the step size, with large values of λ resulting in a more robust update like the gradient descent method in the first iterations and small values of λ result in a fast update like the Gauss-Newton algorithm in later stages. This avoids the drawbacks of both the Gauss-Newton algorithm and gradient descent method whereby it can converge from any initial state, and thereafter rapid convergence near the vicinity of the minimum error. Many variants of the LM method have been developed [21–23].

Consider the cost function:

$$f(\beta) = \frac{1}{n}\sum_{i=1}^{n}[\mathrm{y}_i - g(\mathrm{x}_i, \beta)]^2 \tag{7}$$

where n is the number of datum pairs $(\mathrm{y}_i, \mathrm{x}_i)$; $g(\mathrm{x}_i, \beta)$ is the hypothesis curve; β is the weight vector. The objective is to minimize $f(\beta)$. The weight vector of each successive step is given by:

$$\beta_k = \beta_{k-1} + \delta_k \tag{8}$$

where δ_k is the step at k^{th} iteration; β_k is the corresponding weight vector. To derive δ, an approximation of the value of the hypothesis function at the next step is made by linearization:

$$g(\mathrm{x}_i, \beta + \delta) \approx g(\mathrm{x}_i, \beta) + J_i\delta \tag{9}$$

where J is the gradient of $g(x, \beta)$ with respect to β. The cost function at the point $\beta + \delta$ can then be approximated as follows:

$$f(\beta + \delta) = \frac{1}{n}\sum_{i=1}^{n}[\mathrm{y}_i - g(\mathrm{x}_i, \beta) - J_i\delta]^2 \tag{10}$$

The minimum point of the cost function has a gradient of 0. By taking the derivative of $f(\beta + \delta)$ and setting the results to zero, δ can be obtained as follows:

$$\delta = \frac{J^T[y - g(x, \beta)]}{(J^T J)} \tag{11}$$

Eq. (11) is known as Gauss-Newton update. By introducing a damping factor λ into the denominator, the damped version of this step is now as follows [24,25]:

$$\delta = \frac{J^T[y - g(x, \beta)]}{(J^T J + \lambda I)} \tag{12}$$

where I is an identity matrix. The damping factor λ is adjusted at each iteration such that step size can be controlled. When any iteration results in the worst approximation $(f(\beta + \delta) > f(\beta))$, then λ is increased. Otherwise, as the approximation improves, λ is decreased. Eq. (12) shows that when λ is 0, the update follows a Gauss-Newton update. When λ is infinite, $J^T J$ loses its importance and the update follows a gradient descent update. The iteration stops when the value of either the step δ_k or the cost function $f(\beta)$ reduces to below a predefined value.

2.3 Bayesian Regularization (BR)

To reduce overfitting, this method uses regularization parameters α and β to regulate between reducing errors and weight size. Mackay [26] presented an extensive work on the BR method. He showed that the objective function to be minimized is a linear combination of the sum of squares of errors E_D and the sum of squares of network weights E_w. The objective function is as follows:

$$f(w) = \beta E_D + \alpha E_w \tag{13}$$

When $\alpha \ll \beta$, the training emphasizes on reducing errors. Alternatively, when $\beta \ll \alpha$, the training will emphasize on reducing weight size. The goal of B the R method is to have a changing $f(w)$ after every iteration depending on the regularizing parameters, such that at the end of the training, the resulting network has good generalization qualities. Foresee et al. [27] presented the detailed derivation of α and β. They contributed to the BR algorithm by applying Gauss-Newton approximation to the Hessian matrix of $f(w)$ which is required to calculate α and β. This step reduces the calculation complexity of the BR algorithm.

The application of Bayesian Regularization (BR) within the framework of the LM algorithm will be used [27]. At each iteration, one step of the LM algorithm will be used to minimize $f(w)$. Thereafter, the regularization parameters α and β can be calculated and compared. As α and β change for each successive iteration, $f(w)$ changes, and the minimum point will move. The iteration will stop when $f(w)$ does not significantly change in subsequent iterations resulting in the convergence of the minimum point.

3 Methods

This section will first present the analytical stresses and displacement of a hole in an infinite-width plate under uniaxial load. Next, it will introduce the approach to how the analytical solution can be integrated with supervised learning for both uniaxial load and biaxial load.

3.1 Stress and Displacement around a Hole under Uniaxial Load

In Fig. 1, a hole in an infinite-width plate is subjected to uniaxial tensile load where the tensile load only exists in one primary axis. The stress field solution under uniaxial load [28] can be integrated to obtain the displacement field as follows:

$$u_r = \frac{\sigma r}{2E}\left(\left(1+\nu\right)cos2\theta + \left(1-\nu\right)\right) - \frac{\sigma a^2}{2r^3 E}\left(\left(a^2\left(1+\nu\right) - 4r^2\right)cos2\theta - \left(1+\nu\right)r^2\right) \tag{14}$$

$$u_\theta = -\frac{\sigma sin2\theta}{2r^3 E}\left(\left(a^2+r^2\right)^2 + \nu\left(a^2-r^2\right)^2\right) \tag{15}$$

where σ is the applied stress, a is the radius of the hole, r and θ are the polar coordinates of a point in the element.

3.2 Approach to Apply Supervised Training for Uniaxial Load

The input data set used to train the NN will consist of a set of 20 course mesh of a hole in an infinite-width plate. An analytical solution of a hole under biaxial load will be used as the output target data. A coarse mesh of the quarter-hole is shown in Fig. 2. The infinite-width effect is approximated using half width = 20 × hole radius. Ansys 2D linear element, Plane 182 [29], is used. There are only 2 elements along the perimeter of the quarter-hole in the coarse mesh. The displacement nodal solution of the nearest 6 nodes to the quarter-hole's edge is used as the training parameters. The 6 nodes are shown in Fig. 2. The transfer function of the first layer will be either tangent sigmoid or pure linear, while the transfer function of the output layer will be set to pure linear.

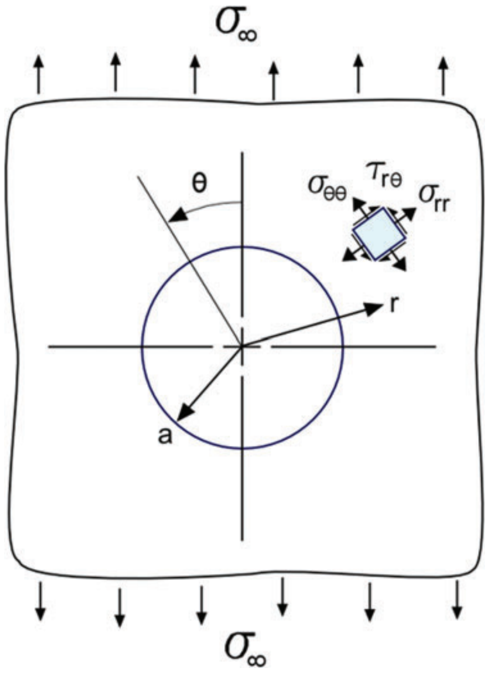


Figure 1: Figure of a hole in an infinite plate under uniaxial tensile load

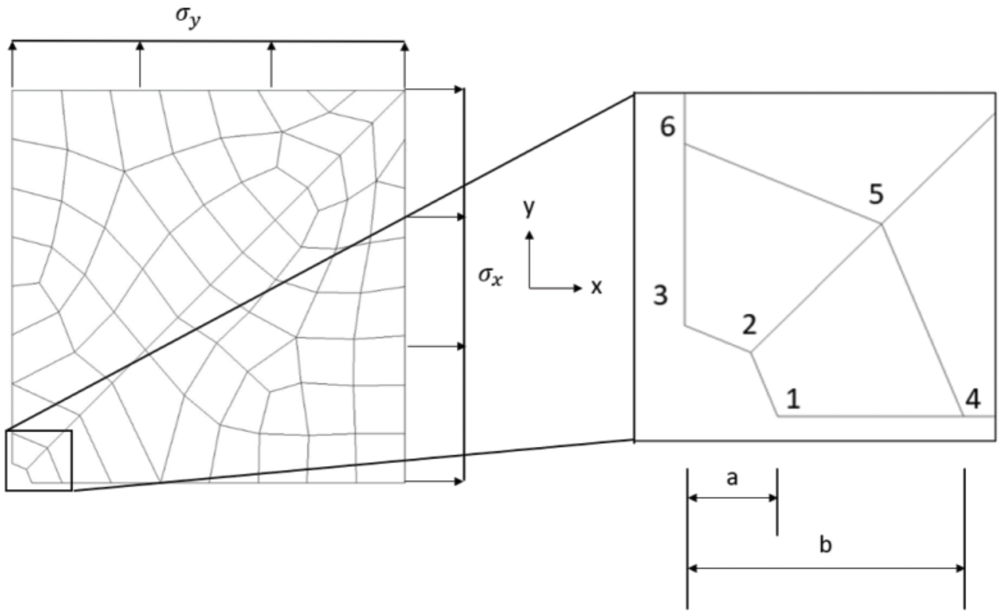


Figure 2: Coarse mesh of a quarter-hole with 2 elements along the perimeter

The material property is based on Aluminum, where $E = 10 \times 10^6\, psi$ and $\nu = 0.3$. The variations of parameters are shown in Table 1. The mesh is subjected to unit stress load, $\sigma_y = 1\ psi$ and $\sigma_x = 0$. There is a total of 20 mesh variations. Fig. 3 shows a few samples of the different coarse mesh variations. To train the model to be sensitive to the magnitude change in displacement, the training is done using a simple linear scale of 70% and 150% on the training parameters which are the displacements of the 6 nodes.

$$\overline{u_{train}} = \begin{bmatrix} 0.7\overline{u_j^{(i)}} \\ 1.5\overline{u_j^{(i)}} \end{bmatrix} \tag{16}$$

where i is node = 1 to 6 and $j = x$ or y.

Table 1: Variation of parameters for uniaxial load

Total sampling size∗	Parameters	Min	Max	Interval
40	b/a	1.5	3	0.5
	$element\ size/a$	0.6	1.4	0.2

Note: ∗The total sampling size is obtained after applying a linear scale of 70% and 150% as shown in Eq. (16), resulting in a total of 2 load variations.

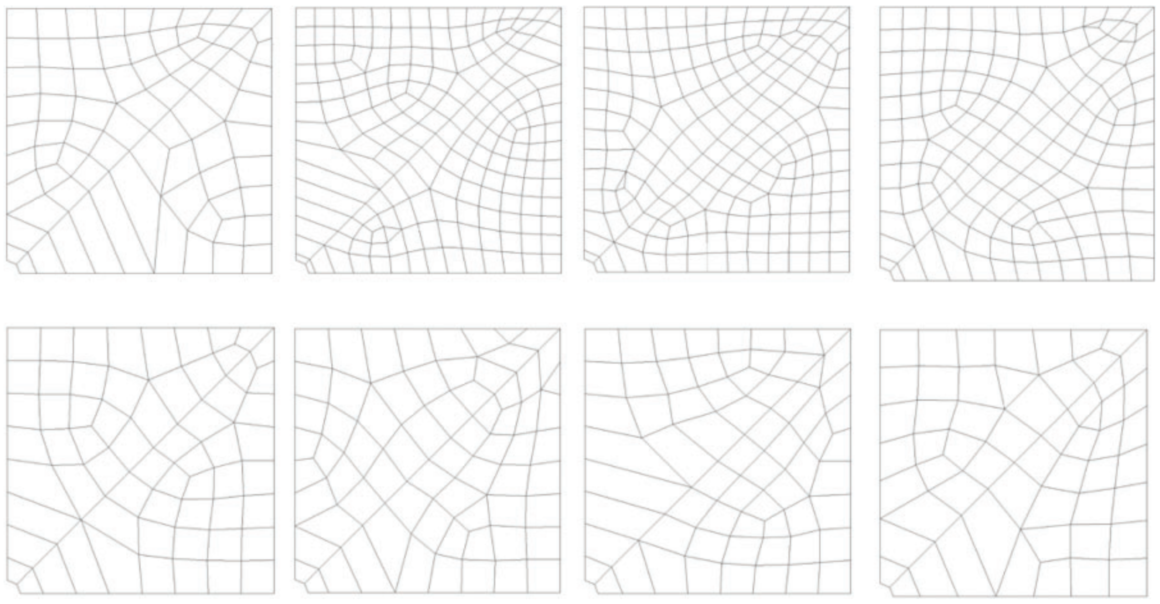


Figure 3: Samples of mesh variations

The neural network consists of a single hidden layer as shown in Fig. 4. The effectiveness of the neural network in predicting the maximum stress of a single hole in a finite-width plate will be investigated using different combinations of training parameters based on the displacements of the 6 nodes shown in Fig. 2. Two approaches to conduct supervised training will be used, one for uniaxial and the other for biaxial load.

10 neurons will be used for the SCG and LM method, while 2 neurons will be used for the BR method as the BR method requires fewer neurons to perform effectively as demonstrated by Kayri [5]. Numerical experimental was conducted and the result shows the accuracy improvement starts to taper off as the number of neurons for SCG and LM methods approaches 10. This could be because there are only 10 input parameters from the 6 nodes (2 DOF × 6 nodes with 2 less due to symmetric boundary conditions). The target consists of the analytical solution of only u_y of node 3 obtained by

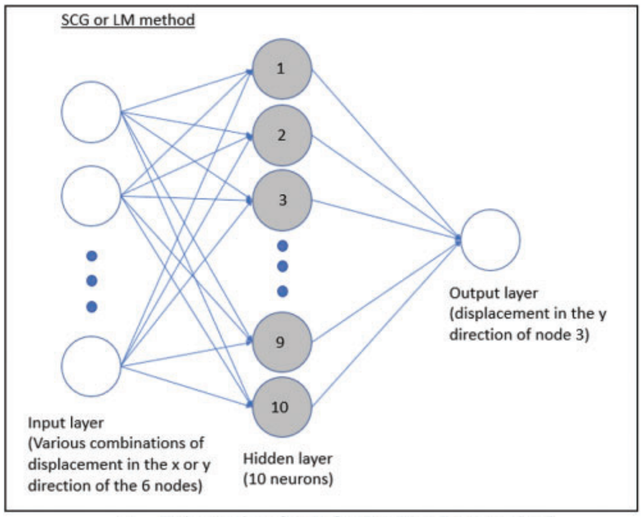
Eq. (14). For the supervised training, both the input and target data are normalized to a magnitude near 1. As a result, the displacement on the 6 nodes is normalized with the u_y of node 3 obtained from coarse mesh.

$$\overline{u_j^{(i)}} = u_j^{(i)} / u_y^{(3)} \tag{17}$$

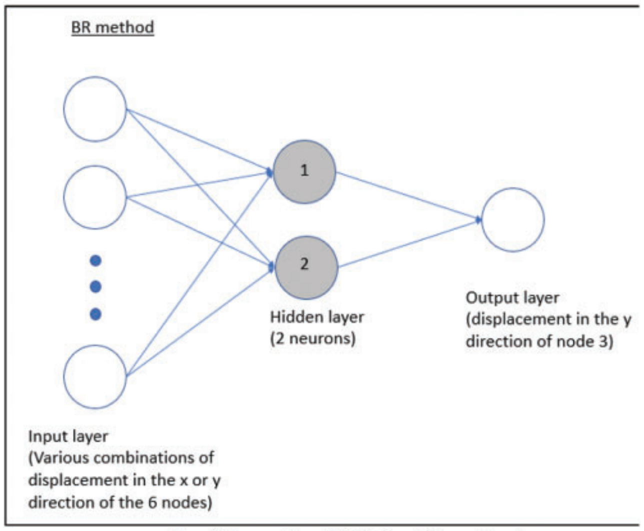
where i is node = 1 to 6 and $j = x$ or y. A normalized displacement, $\overline{u^*}$ will be predicted by the neural network. Considering the stress concentration factor of 3 (Kt of a hole in an infinite-width plate), the predicted maximum stress, σ^* can be obtained from the predicted normalized displacement as follows:

$$\sigma^* = 3\overline{u^*}\frac{u_y^{(3)}}{u_r^{(3)}} \tag{18}$$

where $u_r^{(3)}$ is the analytical radial displacement at $r = a, \theta = 0$.


SCG or LM method
1
2
3
9
10
Output layer (displacement in the y direction of node 3)
Input layer (Various combinations of displacement in the x or y direction of the 6 nodes)
Hidden layer (10 neurons)

a. Schematic of NN for SCG or LM method


BR method
1
2
Hidden layer (2 neurons)
Output layer (displacement in the y direction of node 3)
Input layer (Various combinations of displacement in the x or y direction of the 6 nodes)

b. Schematic of NN for BR method

Figure 4: Neural network with 1 output

3.3 Stress and Displacement around a Hole under Biaxial Load

Since deformation is small, assuming that the response is linear, the principle of superposition can be applied to the uniaxial load solution [28]. In Fig. 5, the analytical stress for hole under applied biaxial stress, σ_y and σ_x, with no applied shear stress will thus be as follows:

$$\sigma_{rr} = \frac{1}{2}\sigma_y\left(1-\frac{a^2}{r^2}\right)+\frac{1}{2}\sigma_y\left(1-\frac{4a^2}{r^2}+\frac{3a^4}{r^4}\right)cos2\theta_1+\frac{1}{2}\sigma_x\left(1-\frac{a^2}{r^2}\right)+\frac{1}{2}\sigma_x\left(1-\frac{4a^2}{r^2}+\frac{3a^4}{r^4}\right)cos2\theta_2 \quad (19)$$

$$\sigma_{\theta\theta} = \frac{1}{2}\sigma_y\left(1+\frac{a^2}{r^2}\right)-\frac{1}{2}\sigma_y\left(1+\frac{3a^4}{r^4}\right)cos2\theta_1+\frac{1}{2}\sigma_x\left(1+\frac{a^2}{r^2}\right)-\frac{1}{2}\sigma_x\left(1+\frac{3a^4}{r^4}\right)cos2\theta_2 \quad (20)$$

$$\tau_{r\theta} = -\frac{1}{2}\sigma_y\left(1+\frac{2a^2}{r^2}-\frac{3a^4}{r^4}\right)sin2\theta_1-\frac{1}{2}\sigma_x\left(1+\frac{2a^2}{r^2}-\frac{3a^4}{r^4}\right)sin2\theta_2 \quad (21)$$

where $\theta_2 = \theta_1 + \frac{\pi}{2}$

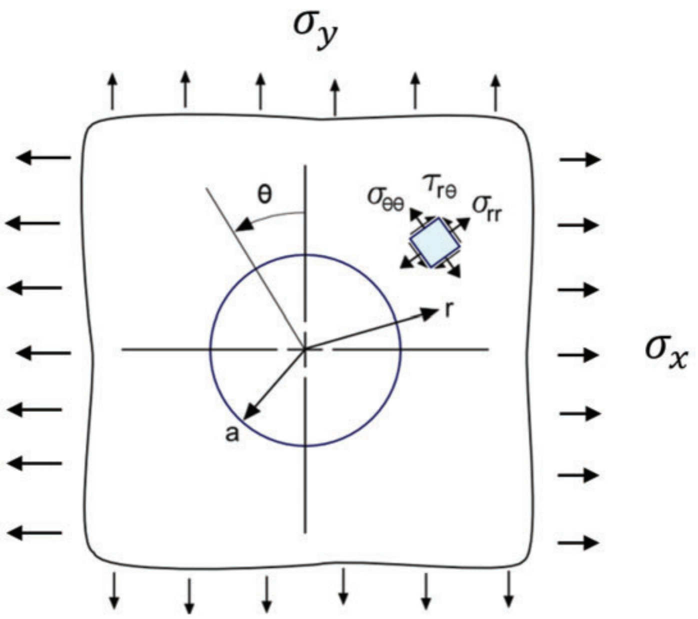


Figure 5: Figure of a hole in an infinite plate under biaxial tensile load

By applying superposition on plane stress displacements under uniaxial load given by Eqs. (14) and (15), the radial and tangential displacement of the hole under biaxial load and plane stress conditions are as follows:

$$u_r = \frac{\sigma_y r}{2E}\left((1+\nu)\,cos2\theta_1+(1-\nu)\right)-\frac{\sigma_y a^2}{2r^3E}\left(\left(a^2(1+\nu)-4r^2\right)cos2\theta_1\right.$$
$$\left.-(1+\nu)\,r^2\right)+\frac{\sigma_x r}{2E}\left((1+\nu)\,cos2\theta_2+(1-\nu)\right)-\frac{\sigma_x a^2}{2r^3E}\left(\left(a^2(1+\nu)-4r^2\right)cos2\theta_2-(1+\nu)\,r^2\right) \quad (22)$$

$$u_\theta = -\frac{\sigma_y sin2\theta_1}{2r^3E}\left(\left(a^2+r^2\right)^2+\nu\left(a^2-r^2\right)^2\right)-\frac{\sigma_x sin2\theta_2}{2r^3E}\left(\left(a^2+r^2\right)^2+\nu\left(a^2-r^2\right)^2\right) \tag{23}$$

where $\theta_2 = \theta_1 + \frac{\pi}{2}$

Aside from plane stress conditions, this paper also evaluates the effectiveness of a 2D coarse mesh training data set generated under general plane strain conditions. Under general plane strain conditions, the strain in the z-direction ε_z is not zero. Under Hooke's law with general plane strain condition, the polar strains are defined as a function of ε_z as follows:

$$\varepsilon_{rr} = \frac{1+\nu}{E}\left[(1-\nu)\sigma_{rr} - \nu\sigma_{\theta\theta} - E\left(\frac{\nu}{1+\nu}\right)\varepsilon_z\right] \tag{24}$$

$$\varepsilon_{\theta\theta} = \frac{1+\nu}{E}\left[(1-\nu)\sigma_{\theta\theta} - \nu\sigma_{rr} - E\left(\frac{\nu}{1+\nu}\right)\varepsilon_z\right] \tag{25}$$

Using Eqs. (19) and (20) to convert the polar stresses in Eqs. (24) and (25) to applied stresses in the cartesian coordinate system, then by integrating the strains, the displacements in terms of applied stresses σ_y and σ_x are derived:

$$u_r = \frac{1}{2Er^3}\left(r^2a^2\left((\sigma_x+\sigma_y+\nu\sigma_x+\nu\sigma_y)+4\cos 2\theta(\sigma_y-\sigma_x+\nu^2\sigma_x-\nu^2\sigma_y)\right)+a^4\cos 2\theta(\sigma_x-\sigma_y+\nu\sigma_x-\nu\sigma_y\right)$$
$$-\frac{r}{2E}\left(\sigma_x cos2\theta-\sigma_y cos2\theta-\sigma_y-\sigma_x+\nu\sigma_x+\nu\sigma_y+2\nu^2\sigma_x+2\nu^2\sigma_y+\nu\sigma_x cos2\theta-\nu\sigma_y cos2\theta+2E\nu\boldsymbol{\varepsilon_z}\right) \tag{26}$$

$$u_\theta = \frac{1}{2Er^3}\left(sin2\theta\left(\sigma_x-\sigma_y\right)(\nu+1)\left(a^4+r^4+2a^2r^2-4a^2r^2\nu\right)\right) \tag{27}$$

Notice that Eq. (26) is a function of the z-component strain ε_z. Hence, successful prediction of 3the D mesh by a neural network trained using a 2D coarse mesh data set (under general plane strain condition) depends on the user input parameter ε_z.

To obtain ε_z, an average will be taken from the z-component strain of the 3 nodes from a coarse 3D mesh as shown in Fig. 6. The resulting value will be used to generate the 2D coarse mesh under general plane strain conditions from APDL [29], as well as to obtain the analytical solution. With the 2D coarse mesh and the analytical solution, the neural network can then be trained to predict stresses for the case of a 3D model.

3.4 Approach to Apply Supervised Training for Biaxial Load

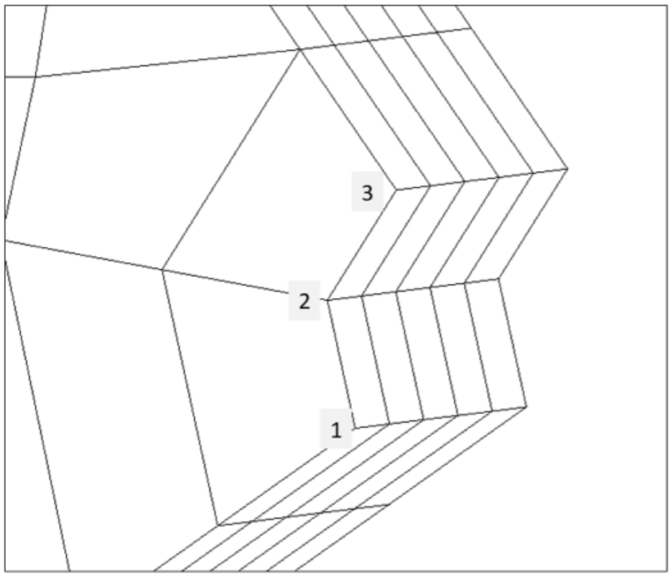
The neural network is trained with the same set of 2D coarse mesh used in the uniaxial load case. Fig. 7 shows the neural network models used for the analysis. In Fig. 2, σ_y is fixed while σ_x is varied such that σ_x/σ_y ranges from -1 to 1. Different intervals of σ_x/σ_y are computed to investigate the minimum sampling size required to obtain a reasonably good prediction. Table 2 shows the variation of parameters for each sampling size.

The target data consists of analytical solutions of u_y of node 3 and u_x of node 1 obtained by Eqs. (22) and (23) for plane stress condition or Eqs. (26) and (27) for general plane strain condition. This is indicated by the 2 outputs shown in 7. The input and target data are normalized as follows:

$$\overline{u_j^{(i)}} = u_j^{(i)}/\sqrt{(u_y^{(3)})^2+(u_x^{(1)})^2} \tag{28}$$

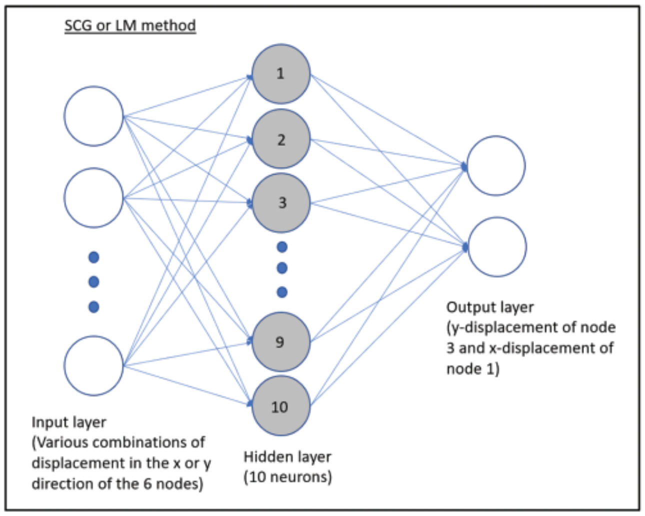
where i is node $= 1$ to 6 and $j = x$ or y. The following part shows the computation of predicted stress from predicted displacements. The displacements under uniaxial load, u_y and u_x is related to the displacements under biaxial load, u_{ye} and u_{xe} by the following:

$$\begin{bmatrix} u_y \; u_x \\ u_x \; u_y \end{bmatrix} \begin{Bmatrix} \sigma_x \\ \sigma_y \end{Bmatrix} = \begin{Bmatrix} u_{xe} \\ u_{ye} \end{Bmatrix} \tag{29}$$



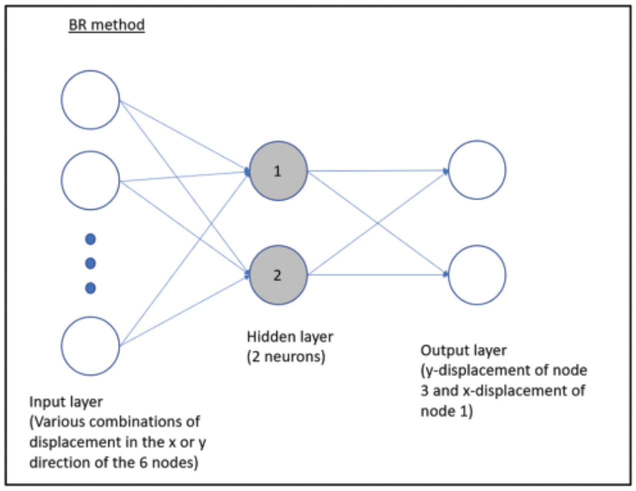
Node	ε_z
1	-0.556E-07
2	-0.375E-07
3	-0.180E-07
Average	-0.370E-07

Figure 6: ε_z from a 3D coarse mesh


SCG or LM method
1
2
3
9
10
Input layer (Various combinations of displacement in the x or y direction of the 6 nodes)
Hidden layer (10 neurons)
Output layer (y-displacement of node 3 and x-displacement of node 1)

a. Schematic of NN for SCG or LM method

Figure 7: (Continued)


BR method
1
2
Hidden layer
(2 neurons)
Output layer
(y-displacement of node 3 and x-displacement of node 1)
Input layer
(Various combinations of displacement in the x or y direction of the 6 nodes)

b. Schematic of NN for BR method

Figure 7: Neural network with 2 outputs

Table 2: Variation of parameters and biaxial loads

Total sampling size∗	Parameters	Min	Max	Interval
	b/a	1.5	3	0.5
	element size$/a$	0.6	1.4	0.2
200	σ_x/σ_y	-1	1	0.5
440	σ_x/σ_y	-1	1	0.2
840	σ_x/σ_y	-1	1	0.1

Note: ∗The total sampling size is obtained after applying a linear scale of 70% and 150% as shown in Eq. (16), resulting in the doubling of load variations for each case of σ_x/σ_y interval.

When substituting u_{ye} and u_{xe} in Eq. (29) for the predicted displacements, the stresses, σ_y and σ_x will be the predicted equivalent applied stresses for a hole in an infinite-width plate. u_y and u_x are the analytical displacements under uniaxial load. Subsequently, by applying Cramer's rule, the predicted applied stresses are:

$$\sigma_x = \frac{\begin{vmatrix} u_{xe} & u_x \\ u_{ye} & u_y \end{vmatrix}}{\begin{vmatrix} u_y & u_x \\ u_x & u_y \end{vmatrix}} \tag{30}$$

$$\sigma_y = \frac{\begin{vmatrix} u_y & u_{xe} \\ u_x & u_{ye} \end{vmatrix}}{\begin{vmatrix} u_y & u_x \\ u_x & u_y \end{vmatrix}} \tag{31}$$

Finally, by using Eq. (20), the predicted maximum stress which is the tangential stress at node 1 can be obtained.

4 Results

In this study, Matlab's library [30], *fitnet*, is used for the neural network supervised training with 'trainscg' (SCG), 'trainbr' (BR), and 'trainlm' (LM) as the backpropagation method. For SCG and LM methods, 70% of the training sample size is used for the training of neural networks, 15% for the validation of the neural network model, and the remaining 15% for the testing of the accuracy of the model. For the BR method, which is a self-validating algorithm, 85% of the training sample size is used for training and 15% for testing. At the hidden layer, either tangent sigmoid or pure linear transfer function will be used. Their effectiveness will be evaluated against each other. In addition, uniaxial and biaxial approaches are compared.

For the prediction of 2D mesh, plane stress condition is used to generate the 2D coarse mesh training data set. For the prediction of 3D mesh, the general plane strain condition is used to generate the 2D coarse mesh training data set [9].

4.1 Infinite-Width Hole: Neural Network Option

The accuracy of the test result based on the displacement at all 6 nodes as training parameters is shown in Table 3. BR method with tangent sigmoid transfer function and LM method with pure linear transfer function offer low prediction errors for both approaches.

Table 3: Root mean squared error (RMSE) of the test sample using different neural models and transfer functions. Training is done using u_x and u_y of all 6 nodes

Model	RMSE (%) under Uniaxial approach$*$		RMSE (%) under Biaxial approach$**$	
	Tangent sigmoid	Pure linear	Tangent sigmoid	Pure linear
SCG	45.87	34.52	15.54	13.08
BR	2.32	60.98	1.65	19.23
LM	6.26	2.80	2.10	1.52

Note: $*$Based on 20 mesh variations and 2 load variations, training sample size of 40; $**$Based on 20 mesh variations and 10 load variations, training sample size of 200; Maximum error with 300 trials.

To further improve the result, the study evaluates the displace field on the element attached to the maximum stress location at Node 1. In this element, Node 2 and Node 5 have non-zero u_y. The training result for this option is shown in Table 4. This step is important because using fewer training parameters reduces the tendency for the neural network model to overfit the training model, which might result in lower prediction error when applying the NN model to a separate set of problems. Tables 3 and 4 show that with the BR method, the tangent sigmoid transfer function offers lower errors than the pure linear transfer function. The opposite is true for SCG and LM methods. This is due to the synergy between the BR method and the tangent sigmoid transfer function as demonstrated by Kayri [5]. Fig. 8 demonstrates the verification of obtained neural network models for the biaxial approach. Figs. 8a and 8b show that the training of NN under biaxial load by both cases offers prediction errors that are unbiased and centered at 0, with the maximum absolute errors of less than 4%.

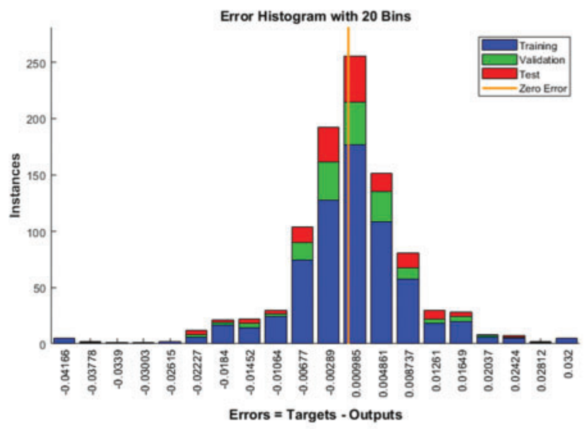
4.2 Finite-Width Single Hole

The neural network model that is trained based on the hole in an infinite-width plate problem will be used to predict the maximum stresses on a hole in a finite-width plate problem. In the infinite-width model, the side wall is far from the hole such that there is no interaction between the stress concentration of the hole with the free edge on the side. However, with the finite-width geometry, the hole is closer to the free edge on the side. As a result of the interaction between the hole and the free edge, the stress concentration of the hole would increase. The analysis of the finite-width model is to evaluate if supervised learning can be used to support the finite element method to calculate stress accurately with an alternate geometry with coarse mesh size. Tests are conducted for a range of w/a (width to hole radius) of 3, 4, 5, and 6. The mesh for this problem is shown in Fig. 9.

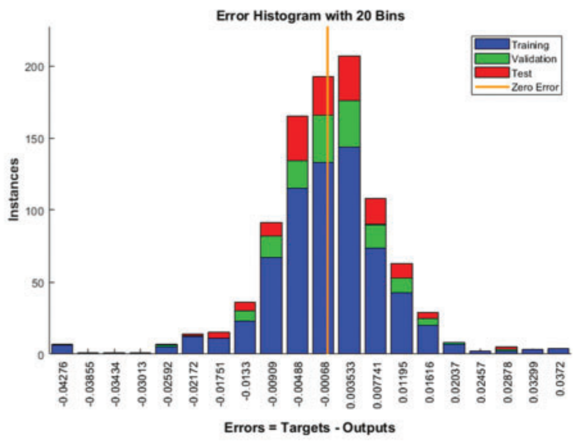
Table 4: Root mean squared error (RMSE) of the test sample using different neural models and transfer functions. Training is done using u_x and u_y of Node 2 and Node 5

Model	RMSE (%) under Uniaxial approach*		RMSE (%) under Biaxial approach**	
	Tangent sigmoid	Pure linear	Tangent sigmoid	Pure linear
SCG	22.56	19.96	14.55	7.93
BR	2.52	51.19	2.16	13.76
LM	9.47	5.42	2.28	2.23

Note: *Based on 20 mesh variations and 2 load variations, training sample size of 40; **Based on 20 mesh variations and 10 load variations, training sample size of 200; Maximum error with 300 trials.



a. u_y and u_x of all 6 nodes as training parameters



b. u_y and u_x of Node 2 and 5 only as training parameters

Figure 8: Error histogram of training using based on biaxial approach using Levenberg-Marquardt method with pure linear transfer function

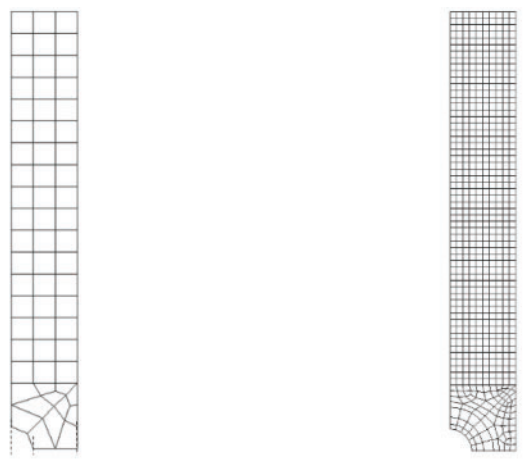
The mesh is constructed by linear elements and only has 2 elements at the quarter-hole perimeter. The result is compared against the Roark's solution [31] given as follows:

$$\sigma_{max} = K_t \sigma_{nom} \tag{32}$$

where

$$\sigma_{nom} = \frac{P}{2t\,(w - r)} \tag{33}$$

$$K_t = 3.00 - 3.13\left(\frac{r}{w}\right) + 3.66\left(\frac{r}{w}\right)^2 - 1.53\left(\frac{r}{w}\right)^3 \tag{34}$$



a. Coarse mesh with 2 elements along the perimeter

b. Fine mesh with 6 elements along the perimeter

Figure 9: Coarse and fine mesh of a quarter-hole with finite-width of $w/a = 3$

The result in Table 5 shows the prediction errors using the neural network model based on both u_x and u_y of all 6 nodes are high. In this case, the tangent sigmoid transfer function performs worse than the pure linear transfer function even with the BR method as shown in Table 5. The poor performance of the tangent sigmoid transfer function could be attributed to the fact that the tangent sigmoid transfer function has properties that are more suitable for classification problems than regression problems. This is indicated by the shape of the graph of a sigmoid function where it levels off to an asymptote at both ends. In this study, using the trained neural network to predict maximum stresses for a different set of problems can be regarded as a regression problem. Hence, the pure linear transfer function may perform better as an approximator. However, the prediction errors using the pure linear transfer function are still high. Therefore, the neural network model trained using the infinite hole mesh cannot be applied to the finite hole mesh under this setup.

Table 5: Prediction error (%) using different neural models and transfer functions. Training done using u_x and u_y of all 6 nodes

Model	Absolute error (%) under Uniaxial approach$*$		Absolute error (%) under Biaxial approach$**$	
	Tangent sigmoid	Pure linear	Tangent sigmoid	Pure linear
SCG	87.40	77.51	81.51	19.80
BR	34.97	18.39	18.27	13.62
LM	74.79	21.40	112.30	16.45

Note: $*$Based on 20 mesh variations and 2 load variations, training sample size of 40; $**$Based on 20 mesh variations and 10 load variations, training sample size of 200; Maximum error with 300 trials, conducted for a range of *w*/*a*: 3, 4, 5 and 6.

To evaluate if the trained model can be further improved by focusing the training on just the element with the highest stress, the training is based on u_y and u_x of Node 2 and Node 5. Table 6 shows that using just u_y and u_x of Node 2 and Node 5 indeed reduced the prediction error under pure linear transfer function as using fewer parameters reduces the tendency to overfit to training model. With only 20 mesh variations and a sample size of 40, the uniaxial method is unable to provide a satisfactory prediction for the finite hole. On the other hand, with the same 20 mesh variations but with a sample size of 200, the biaxial approach can offer low prediction errors under the LM method and BR method with pure linear transfer function.

Table 6: Prediction error (%) using different neural models and transfer functions. Training is done using u_x and u_y of Node 2 and Node 5

Model	Absolute error (%) under Uniaxial approach∗		Absolute error (%) under Biaxial approach∗∗	
	Tangent sigmoid	Pure linear	Tangent sigmoid	Pure linear
SCG	66.17	27.40	57.70	6.39
BR	55.81	13.96	4.26	3.98
LM	86.82	13.02	325.27	3.55

Note: ∗Based on 20 mesh variations and 2 load variations, training sample size of 40; ∗∗Based on 20 mesh variations and 10 load variations, training sample size of 200; Maximum error with 300 trials, conducted for a range of w/a: 3, 4, 5 and 6.

In Table 7, the stress concentration, K_t, has a maximum error of 26.2% for the conventional FEA analysis. However, with the aid of the neural network, the maximum error is reduced to just 3.55%. To obtain similar accuracy without supervised learning, a much finer mesh is required. A fine mesh with 6 elements along the perimeter, as shown in 9b, would be necessary.

Table 7: A sensitivity study on the mesh size where n is the number of elements on the hole perimeter

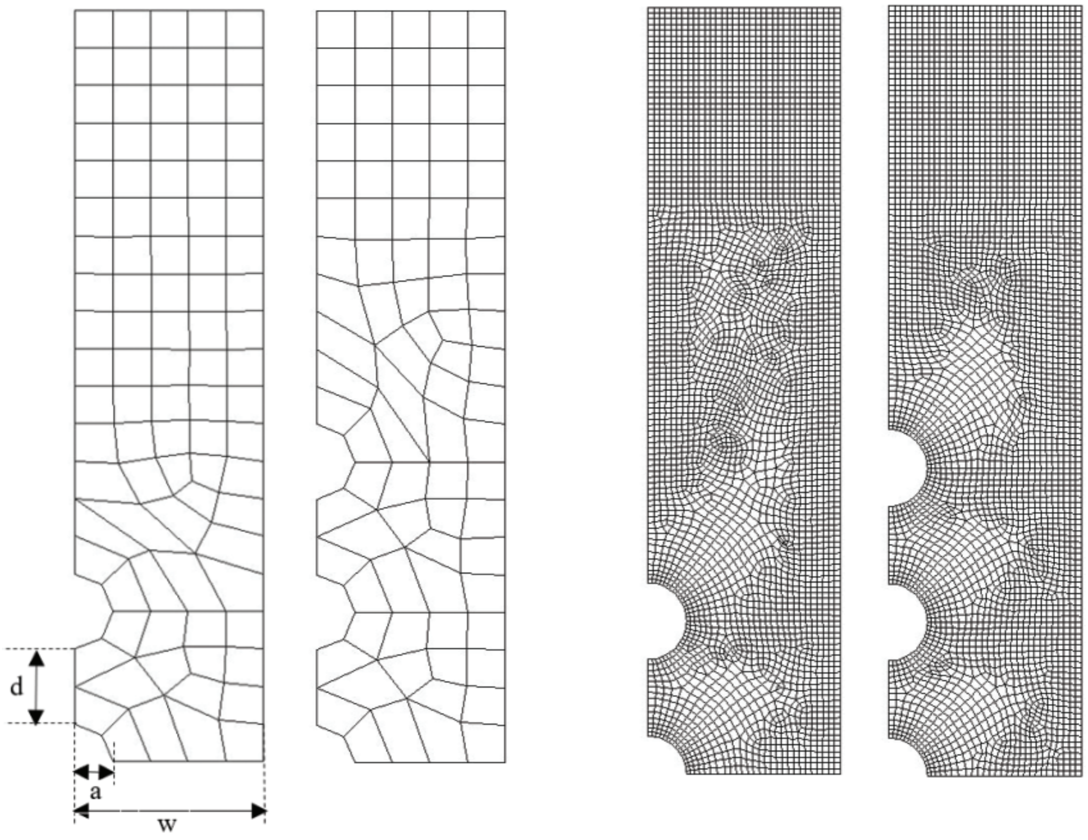
		$w/a = 3$		$w/a = 4$		$w/a = 5$		$w/a = 6$	
	n	Kt	Error	Kt	Error	Kt	Error	Kt	Error
Roark		3.460		3.230		3.135		3.088	
Biaxial approach∗	2	3.340	−3.47%	3.115	−3.55%	3.112	−0.73%	3.031	−0.85%
FEM	2	2.552	−26.2%	2.420	−25.1%	2.404	−23.3%	2.377	−23.0%
FEM	4	3.281	−5.2%	3.072	−4.9%	2.984	−4.8%	2.940	−4.8%
FEM	6	3.388	−2.08%	3.188	−1.30%	3.099	−1.15%	3.057	−1.85%

Note: ∗Using u_y and u_x of Node 2 and 5 as training parameters, the Levenberg-Marquart algorithm, pure linear function, based on the maximum error with 300 trials, using the biaxial approach.

Both the LM and BR methods offer comparable results. Since the LM method has lower computational costs, the LM method would be the preferred method. The LM method will thus be applied on mesh with very different configurations such as multiple holes and staggered holes on finite-width plates for further tests. The biaxial approach will be used since it offers lower error than the uniaxial approach and has the advantage of being sensitive to biaxial loads.

4.3 Multiple Holes on Finite-Width Plate

The holes are separated by distance, ***d***, from edge to edge of holes as shown in Fig. 10. To evaluate the accuracy of the prediction, the Kt values are obtained from the fine mesh using quadratic element Plane 183 as shown in Fig. 10b [9]. These Kt values are compared to the Kt values predicted by the neural network using displacements from coarse mesh shown in Fig. 10a. The variations in the mesh are shown in Table 8, with a fixed $w/a = 5$.



a. Coarse mesh with 2 elements along the perimeter (Plane 182), Left: 3 holes; Right: 5 holes

b. Fine mesh with 16 elements along the perimeter (Plane 183) Left: 3 holes; Right: 5 holes

Figure 10: Coarse and fine mesh of multiple holes with finite-width of $w/a = 5$, $d = 2$

Table 8: Variations in finite multiple hole mesh

	Min	Max	Interval
d/a	2	4	1

Table 9 investigates the minimum number of training sampling sizes needed to obtain good predictions for mesh with different configurations using biaxial-trained NN. To allow the NN to fit more loosely to the single hole in an infinite plate problem, u_y and u_x of Nodes 2 and 5 are used as training parameters such that the NN can effectively predict the stresses for multiple holes on finite plate problems. Table 9 shows that a sampling size of 440 lines offers significant improvement over 200 lines while the sampling size of 840 lines does not offer much improvement. The biaxial approach will be used to illustrate the effects of different training sampling sizes.

Table 9: Prediction errors (%) using different sampling sizes for the case of 3 holes, $w/a = 5$

	Prediction error (%) under Biaxial approach*		
	$d/a = 2$	$d/a = 3$	$d/a = 4$
Sampling size of 200	−13.47	−4.88	−3.02
Sampling size of 440	0.48	3.01	2.07
Sampling size of 840	0.37	2.89	1.97

Note: *Using u_y and u_x of Nodes 2 and 5 as training parameters, the Levenberg-Marquart algorithm, a pure linear function, based on a maximum absolute error with 300 trials.

To ensure that the NN is also effective in predicting similar problems, the prediction error of a quarter finite plate with 3 holes as shown in Fig. 10 is studied. Table 10 shows that errors are generally higher than those in Table 9. However, the difference is not significant.

Table 10: Prediction errors (%) for case of 5 holes, $w/a = 5$

	Prediction error (%) under Biaxial approach*		
	$d/a = 2$	$d/a = 3$	$d/a = 4$
Sampling size of 440	−0.74	3.27	2.14

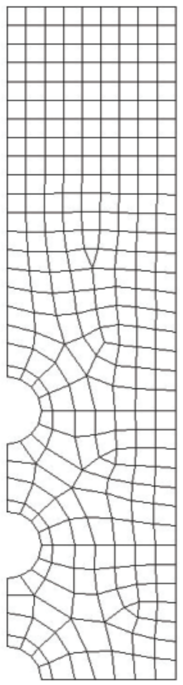
Note: *Using u_y and u_x of Nodes 2 and 5 as training parameters, the Levenberg-Marquart algorithm, a pure linear function, based on a maximum absolute error with 300 trials.

A sensitivity study will be conducted for the case of 5 holes since it has higher prediction errors. Table 11 shows that even for 5 holes, to achieve similar accuracy, a fine mesh with 6 elements along the perimeter as shown in Fig. 11 would be necessary. The prediction errors in Table 11 are low and comparable to that of the single hole in finite-width plate problem. This is because of the absence of any stagger of the holes which may introduce additional shear stresses. Therefore, the stress field is still similar to that of a single-hole problem. As a result, the trained NN model will still be able to make satisfactory predictions in this problem. Furthermore, a w/a of 5 is used in this problem. With higher w/a, the prediction error is expected to be low as it has higher similarity with the infinite-width problem that is used to train the NN.

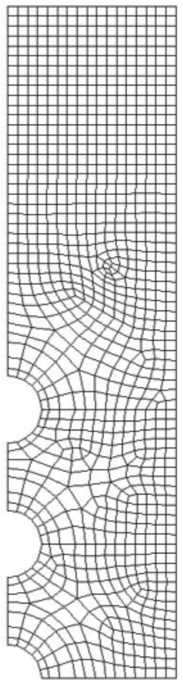
Table 11: A sensitivity study on the mesh size for 5 holes where n is the number of elements on the hole perimeter (based on fine mesh shown in Fig. 10b)

		$d/a = 2$		$d/a = 3$		$d/a = 4$	
	n	Kt	Error	Kt	Error	Kt	Error
FEM (quadratic element, Plane 183)	16	2.512		2.693		2.827	
Biaxial approach∗ (linear element, Plane 182)	2	2.493	−0.74%	2.781	3.27%	2.887	2.14%
FEM (linear element, Plane 182)	2	1.940	−22.77%	2.132	−20.85%	2.227	−21.25%
FEM (linear element, Plane 182)	4	2.344	−6.67%	2.528	−6.15%	2.661	−5.90%
FEM (linear element, Plane 182)	6	2.469	−1.68%	2.664	−1.11%	2.796	−1.11%

Note: ∗Using u_y and u_x of Node 2 and 5 as training parameters, training sampling size of 440, Levenberg-Marquart algorithm, pure linear function, based on maximum absolute error with 300 trials.



a. Coarse mesh with 4 elements along the perimeter (Plane 182)



b. Coarse mesh with 6 elements along the perimeter (Plane 182)

Figure 11: Various fine mesh of 5 holes

4.4 Staggered Holes on Finite-Width Plate

There is a stagger of $s = 2 *$ radius of the hole. From Fig. 12a, only the elements adjacent to the quarter-hole are relevant. The mesh distortions away from the quarter-hole are inconsequential. Like the previous section, the Kt values of fine mesh shown in Fig. 12b are compared to the Kt values predicted by the neural network using displacements from coarse mesh. The variations of the mesh are shown in Table 12 with a fixed $w/a = 5$.

Since Node 5 is in the vicinity of the other hole, its stresses may be affected by the stress field of the other hole. This may increase discrepancies from that of a single hole. Table 13 evaluates if prediction error can be reduced by removing Node 5 from the training parameters. Indeed, prediction errors are lowered by using u_y and u_x of only Node 2. Using fewer training parameters also reduces the tendency for NN to overfit the training setup.

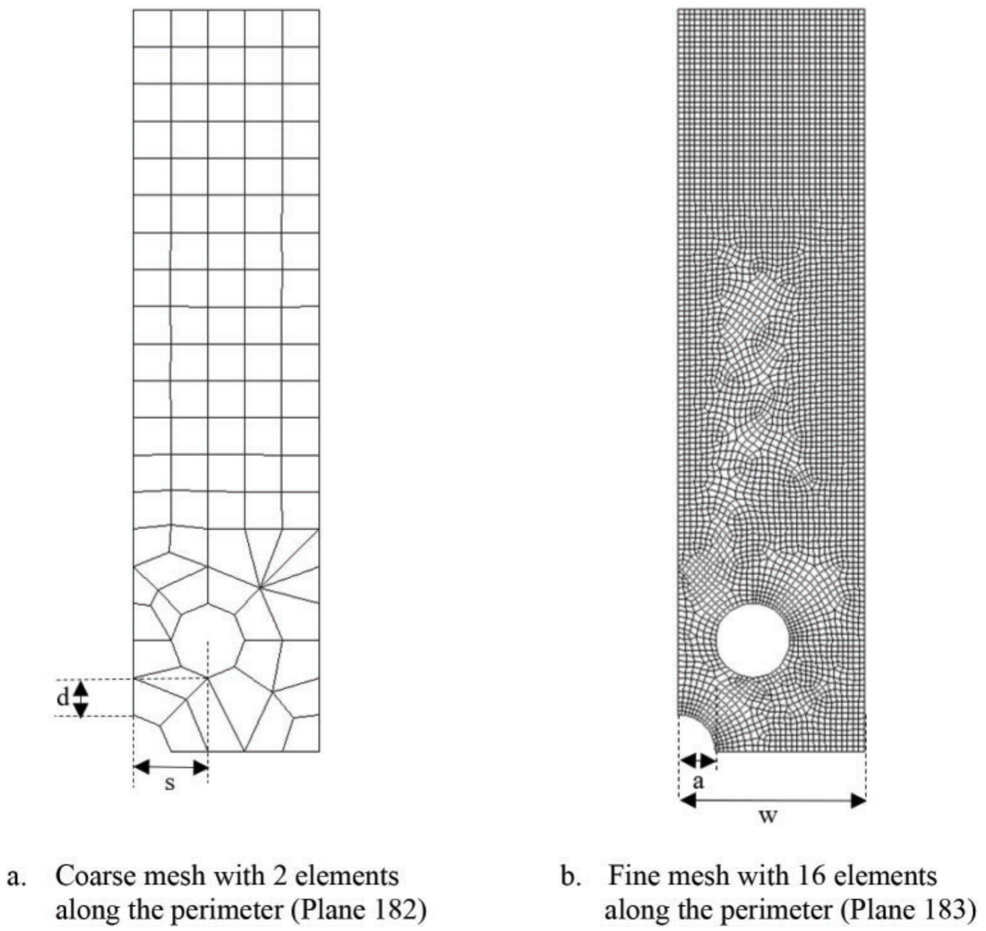

d
s
a
w
a. Coarse mesh with 2 elements along the perimeter (Plane 182)
b. Fine mesh with 16 elements along the perimeter (Plane 183)

Figure 12: Coarse and fine mesh of staggered holes with finite-width of $w/a = 5$; $d = 1$; $s/a = 2$

Table 12: Variations in finite multiple-hole mesh

	Min	Max	Interval
d/a	1	3	1

Table 13: Prediction error (%) using a different set of training parameters for the case of $s/a = 2$, $w/a = 5$

	Prediction error (%) under Biaxial approach∗		
	$d/a = 1$	$d/a = 2$	$d/a = 3$
u_y and u_x of Nodes 2 and 5	8.12	6.81	4.38
u_y and u_x of Node 2	4.61	4.07	3.11

Note: ∗Based on the training sampling size of 440, Levenberg-Marquart algorithm, pure linear function, maximum absolute error over 300 trials.

Similar to Table 9 in Section 4.5, Table 14 shows that a sampling size of 440 lines offers a significant improvement over 200 lines while 840 lines offer little improvement.

Table 14: Prediction error (%) using different training sampling size for the case of $s/a = 2$, $w/a = 5$

	Prediction error (%) under Biaxial approach∗		
	$d/a = 1$	$d/a = 2$	$d/a = 3$
Sampling size of 200	31.95	16.96	5.35
Sampling size of 440	4.61	4.07	3.11
Sampling size of 840	4.55	3.97	2.98

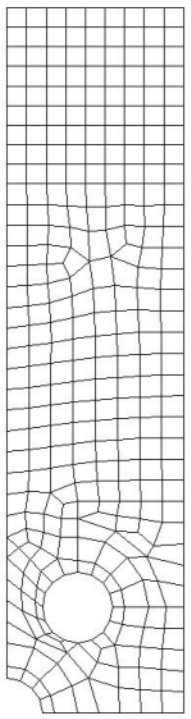
Note: ∗Using u_y and u_x of Node 2 as training parameters, Levenberg-Marquart algorithm, pure linear function, maximum absolute error over 300 trials.

Table 15 shows that for $s/a = 2$, to achieve similar accuracy, a fine mesh with 6 elements along the perimeter shown in Fig. 13 would be necessary. The prediction errors are slightly higher than those for holes without stagger (Table 11) but are still satisfactory. This is due to the extra shear effect introduced by the staggered hole.

Table 15: A sensitivity study on the mesh size for $s/a = 2$ where n is the number of elements on the hole perimeter (compared to fine mesh shown in Fig. 12b)

		$d/a = 1$		$d/a = 2$		$d/a = 3$	
	n	Kt	Error	Kt	Error	Kt	Error
FEM (quadratic element, Plane 183)	16	3.465		2.988		2.878	
Biaxial approach∗ (linear element, Plane 182)	2	3.625	4.61%	3.110	4.07%	2.968	3.11%
FEM (linear element, Plane 182)	2	2.773	−19.98%	2.386	−20.14%	2.289	−20.47%
FEM (linear element, Plane 182)	4	3.394	−2.06%	2.857	−4.40%	2.739	−4.84%
FEM (linear element, Plane 182)	6	3.512	1.36%	2.988	0.01%	2.864	−0.47%

Note: ∗ Using u_y and u_x of Node 2 as training parameters, training sampling size of 440, Levenberg-Marquart algorithm, pure linear function, based on maximum absolute error with 300 trials.


a. Coarse mesh with 4 elements along the perimeter (Plane 182)

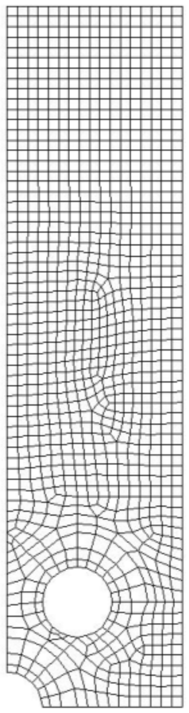

b. Coarse mesh with 6 elements along the perimeter (Plane 182)

Figure 13: Various fine mesh of staggered holes with $s/a = 2$, $w/a = 5$

4.5 3D Single Hole in the Infinite Plate Using Biaxial Load Trained NN

The 3D mesh model is a quarter-hole in an infinite plate that is bisected in half along the z-axis. The coarse mesh is constructed with linear element Solid 185. Since this is the problem of predicting the stresses of a single hole in an infinite-width plate using a biaxial load trained NN like in Section 4.3, u_y and u_x of all 6 nodes of the coarse mesh will be used as training parameters. The predicted stresses are compared to that of the fine mesh shown in Fig. 14 which is constructed with the quadratic element Solid 186 [10]. Table 16 shows the variation of thickness tested.

Table 17 shows that to achieve similar accuracy, a much finer mesh with 8 elements along the quarter-hole perimeter is required as shown in Fig. 15. Furthermore, the mesh must be constructed by quadratic element Solid 186.

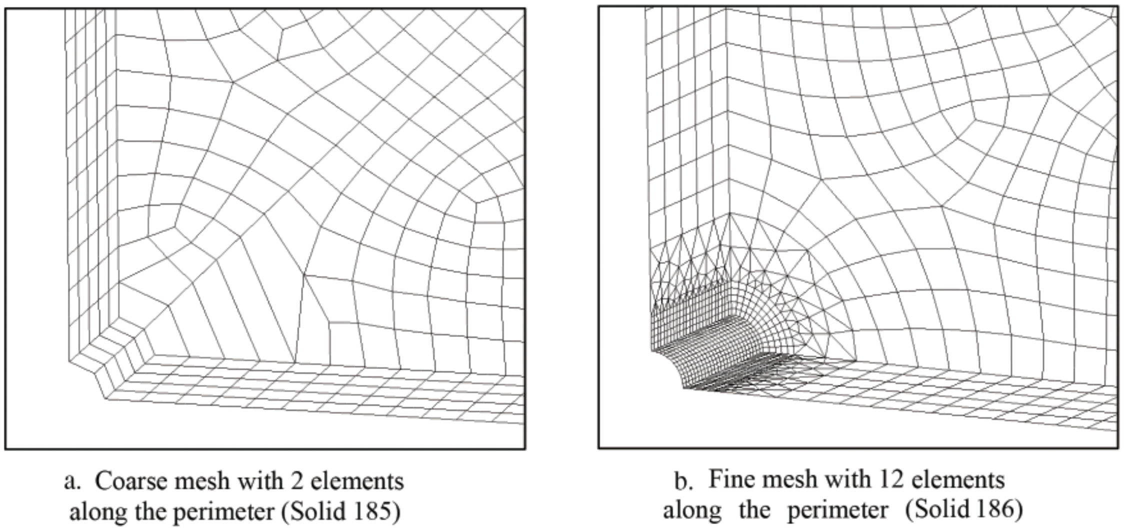

a. Coarse mesh with 2 elements along the perimeter (Solid 185)
b. Fine mesh with 12 elements along the perimeter (Solid 186)

Figure 14: Coarse and fine mesh of 3D hole in infinite plate ($w/a = 20$) with thickness, t where $t/a = 10$

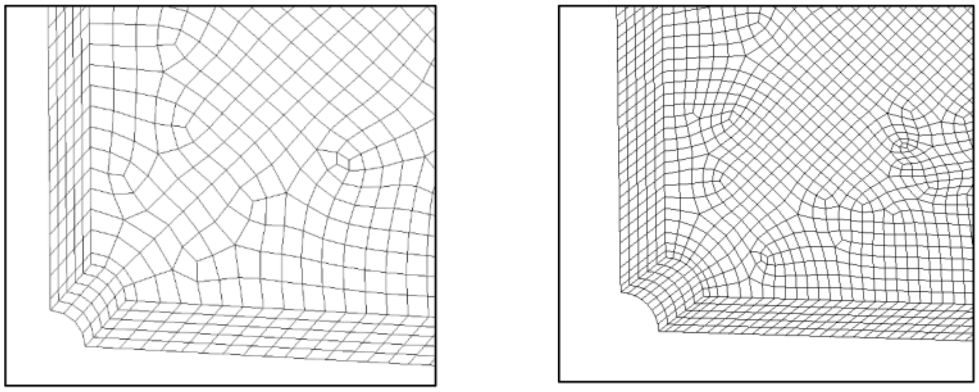
Table 16: Variations in finite multiple-hole mesh

	Min	Max	Interval
t/a	6	18	4

Table 17: A sensitivity study on the mesh size where n is the number of elements on the hole perimeter

		$t/a = 6$		$t/a = 10$		$t/a = 14$		$t/a = 18$	
	n	Kt	Error	Kt	Error	Kt	Error	Kt	Error
FEM (quadratic element, Solid 186)	12	3.053		3.038		3.033		3.032	
Biaxial approach∗ (linear element, Solid 185)	2	3.112	1.93%	3.091	1.74%	3.082	1.62%	3.079	1.55%
FEM (linear element, Solid 185)	2	1.963	−35.70%	1.958	−35.55%	1.957	−35.48%	1.956	−35.49%
FEM (linear element, Solid 185)	4	2.586	−15.30%	2.573	−15.31%	2.569	−15.30%	2.568	−15.30%
FEM (linear element, Solid 185)	8	2.844	−6.85%	2.827	−6.95%	2.825	−6.86%	2.824	−6.87%
FEM (quadratic element, Solid 186)	4	2.95	−3.37%	2.935	−3.39%	2.93	−3.40%	2.93	−3.36%
FEM (quadratic element, Solid 186)	8	3.043	−0.33%	3.028	−0.33%	3.023	−0.33%	3.022	−0.33%

Note: ∗Using u_y and u_x of all 6 Nodes as training parameters, training sampling size of 440, Levenberg-Marquart algorithm, pure linear function, based on maximum absolute error with 300 trials.



a. Coarse mesh with 4 elements along the perimeter

b. Coarse mesh with 8 elements along the perimeter

Figure 15: Various 3D fine mesh of single hole in infinite-width plate

5 Conclusion

The result shows that with the aid of supervised learning, an error of less than 5% can be obtained even for a coarse mesh with only 2 elements along the whole quarter perimeter. Training is done using the analytical solution for a coarse 2D mesh of a single hole in an infinite-width plate (2D). The prediction is successfully applied to different geometries such as holes in finite-width plates (2D), multiple holes without stagger (2D) and staggered holes (2D). A preliminary study done in this paper also demonstrates that the approach can be applied to 3D models with minor adjustments.

Levenberg-Marquardt method (LM) with pure linear transfer function and Bayesian Regularization (BR) with tangent sigmoid transfer function can achieve low prediction errors for the hole in infinite-width and finite-width problems. However, LM is faster and less calculational complex than BR. Thus, LM is extensively tested in this study. The result shows good prediction accuracy for the different problems.

Nevertheless, various factors can affect the prediction results. Firstly, to train neural networks to be sufficiently sensitive to biaxial loading, the interval of σ_x/σ_y must be sufficiently small. An interval of 0.2 for the range of -1 to 1 is necessary. Secondly, using more training parameters increases the tendency for the NN to overfit the training data set. To reduce the prediction error for other problems such as finite-width (2D) and multiple holes (2D), the number of training parameters is reduced. This is done by using only non-zero displacements of the nodes associated with the element with the highest stress. In this case, 8 non-zero displacements from the 6 nodes are reduced to 4 non-zero displacements. Under these conditions, LM with pure linear transfer function offers maximum absolute prediction errors of 1.52% in the infinite-width problem; 3.55% in the finite-width problem; and 3.27% in the multiple holes (without stagger) problem. In contrast, conventional FEM with a coarse mesh of 2 elements at the quarter-hole perimeter has an error of 26.2%. To achieve similar accuracy by FEM without the aid of supervised learning, at least 6 elements at the quarter-hole perimeter would be necessary.

For staggered holes (2D), the prediction errors are higher due to the influence of shear stress which the neural network is not trained to model. In this case, the interaction of nearby holes may affect the prediction accuracy. Nodes chosen are encouraged to be further away from the other holes to reduce the influence of the stress field of the other adjacent holes. For example, for the staggered holes problem, removing the nodes that are in proximity of the adjacent hole from the training parameters drastically reduced the prediction errors to 4.61%. In contrast, conventional FEM with a coarse mesh of 2 elements at the quarter-hole perimeter has an error of 20.47%. To achieve similar accuracy by FEM without the aid of supervised learning, at least 6 elements at the quarter-hole perimeter would be necessary.

This paper has also demonstrated that the 3D problem consisting of a hole in an infinite-width 3D plate can be accurately predicted using supervised learning. The neural network training is performed using the 2D coarse mesh training set with generalized plane strain condition. When applied to the 3D plate that has a coarse mesh with just 2 elements at the quarter-hole perimeter, the maximum prediction error is 1.93%. In contrast, using the conventional FEM without any supervised training gives an error of 35.7%. To achieve similar accuracy by conventional FEM without supervised learning, more than 8 elements at the quarter-hole perimeter would be necessary.

Hence, it is demonstrated that supervised learning can be used effectively to aid finite element analysis such that even coarse mesh can offer satisfactory accuracy. As a result of using this approach, when performing complex 3-D design iterations with many holes, the need to perform many sub-modeling analyses (for greater accuracy) is significantly reduced.

Acknowledgement: The authors thank the guidance and encouragement from Prof S.N. Atluri over the years.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: The authors confirm their contribution to the paper as follows: Study conception and design: W.T. Chow; analysis and interpretation of results: J.T. Lau, W.T. Chow; draft manuscript preparation: J.T. Lau. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: Data will be available upon request.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. M. F. Moller, “A scaled conjugate gradient algorithm for fast supervised learning,” *Neural Netw.*, vol. 6, no. 4, pp. 525–533, 1993. doi: 10.1016/S0893-6080(05)80056-5.
2. H. P. Gavin, *The Levenberg-Marquardt Algorithm for Nonlinear Least Squares Curve-Fitting Problems*. Durham: Department of Civil and Environmental Engineering, Duke University, 2019.
3. M. G. Shirangi and A. A. Emerick, “An improved TSVD-based Levenberg-Marquardt algorithm for history matching and comparison with Gauss-Newton,” *J. Petrol. Sci. Eng.*, vol. 143, no. 3, pp. 258–271, 2016. doi: 10.1016/j.petrol.2016.02.026.
4. P. S. Prerana and P. Sehgal, “Comparative study of GD, LM and SCG method of neural network for thyroid disease diagnosis,” *Int. J. Appl. Res.*, vol. 1, no. 10, pp. 34–39, 2015.
5. M. Kayri, “Predictive abilities of Bayesian regularization and Levenberg-Marquardt algorithms in artificial neural networks: A comparative empirical study on social data,” *Math. Comp. App.*, vol. 21, no. 2, pp. 20, 2016. doi: 10.3390/mca21020020.

6. N. P. Jouppi *et al.*, "In-datacenter performance analysis of a tensor processing unit," in *Proc. of the 44th Int. Symp. on Computer Architecture (ISCA)*, Toronto, Canada, 2017, pp. 1–12.
7. Y. M. A. Hashash, S. Jung, and J. Ghaboussi, "Numerical implementation of a neural network based material model in finite element analysis," *Int. J. Numer. Meth. Eng.*, vol. 59, no. 7, pp. 989–1005, 2004. doi: 10.1002/nme.905.
8. L. Manevitz, A. Bitar, and D. Givoli, "Neural network time series forecasting of finite-element mesh adaptation," *Neurocomputing*, vol. 63, no. 6, pp. 447–463, 2005. doi: 10.1016/j.neucom.2004.06.009.
9. W. T. Chow, "Supervised learning for finite element analysis of holes under tensile load," in *Int. Conf. on Comp. & Experimental Eng. and Sci.*, Switzerland, Springer International Publishing, 2019, pp. 1329–1339.
10. C. M. Bishop, *Neural Networks for Pattern Recognition*. UK: Oxford University Press, 1995.
11. S. Haykin, *Neural Networks: A Comprehensive Foundation*. USA: Prentice Hall PTR, 1998.
12. H. Almuallim and T. G. Dietterich, "Learning with many irrelevant features," *Artif. Intell.*, vol. 91, pp. 547–552, 1991.
13. D. F. Gordon and M. Desjardins, "Evaluation and selection of biases in machine learning," *Mach. Learn.*, vol. 20, no. 1–2, pp. 5–22, 1995. doi: 10.1007/BF00993472.
14. H. H. Tan and K. H. Lim, "Vanishing gradient mitigation with deep learning neural network optimization," in *2019 7th Int. Conf. on Smart Computing & Communications (ICSCC)*, Sarawak, Malaysia, 2019, pp. 1–4.
15. S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *Int. J. Uncertain., Fuzzi. Knowled.-Based Sys.*, vol. 6, no. 2, pp. 107–116, 1998. doi: 10.1142/S0218488598000094.
16. D. Nguyen and B. Widrow, "Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights," in *1990 IJCNN Int. Joint Conf. on Neural Networks*, San Diego, USA, 1990, pp. 21–26.
17. Matlab, *Nguyen-Widrow Layer Intialization Function*. USA: MathWorks, 2018.
18. A. Pavelka and A. Procházka, "Algorithms for initialization of neural network weights," in *Proc. of the 12th Annual Conf.*, MATLAB, 2004, pp. 453–459.
19. M. R. Wayahdi, M. Zarlis, and P. H. Putra, "Initialization of the Nguyen-widrow and Kohonen algorithm on the backpropagation method in the classifying process of temperature data in Medan," *J. Phy.: Conf. Series*, vol. 1235, no. 1, pp. 012031, 2019. doi: 10.1088/1742-6596/1235/1/012031.
20. J. Lundén and V. Koivunen, "Scaled conjugate gradient method for radar pulse modulation estimation," in *2007 IEEE Int. Conf. on Acoustics, Speech and Signal Processing-ICASSP'07*, Honolulu, USA, 2007, pp. II-297.
21. M. I. Lourakis, "A brief description of the Levenberg-Marquardt algorithm implemented by levmar," *Found. Res. Technol.*, vol. 4, no. 1, pp. 1–6, 2005.
22. K. Madsen, H. B. Nielsen, and O. Tingleff, *Methods for Non-Linear Least Squares Problems*. Denmark: Technical University of Denmark, 2004.
23. H. B. Nielsen, *Damping Parameter in Marquardt's Method*. Denmark: Technical University of Denmark, 1999.
24. K. Levenberg, "Method for the solution of certain problems in least squares siam," *Q. Appl. Math.*, vol. 2, no. 2, pp. 164–168, 1944. doi: 10.1090/qam/10666.
25. D. W. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters," *J. Soc. Ind. Appl. Math.*, vol. 11, no. 2, pp. 431–441, 1963. doi: 10.1137/0111030.
26. D. J. MacKay, "Bayesian interpolation," *Neural Comput.*, vol. 4, no. 3, pp. 415–447, 1992. doi: 10.1162/neco.1992.4.3.415.
27. F. D. Foresee and M. T. Hagan, "Gauss-Newton approximation to Bayesian learning," in *Proc. of Int. Conf. on Neural Networks (ICNN'97)*, Houston, USA, 1997, pp. 1930–1935.

28. W. D. Pilkey, D. F. Pilkey, and Z. Bi, *Peterson's Stress Concentration Factors*. USA: John Wiley & Sons, 2020.
29. ANSYS, Inc. *Mechanical APDL Theory Reference*. USA: ANSYS, Inc., 2018.
30. Matlab, *Statistics and Machine Learning Toolbox: User's Guide*. USA: MathWorks, 2018.
31. W. C. Young, R. G. Budynas, and A. M. Sadegh, *Roark's Formulas for Stress and Strain*. USA: McGraw-Hill Education, 2012.