International Journal of Engineering Sciences 2024 16(1) 31 - 40

INTERNATIONAL JOURNAL
OF
ENGINEERING SCIENCES

Integration of FEA and GNN for Structural Analysis in Discontinuous Geometries

Shailesh S. Kadre,[1] S. K. Vemuri,[2] P.K. Mukka[3], R.G. Paspula [4] and Vipin K. Tripathi[5]
[1, 2, 3, 4]Analysis and Simulation Department, Cyient Ltd, India
[5]National Institute of Technical Teachers' Training & Research, Bhopal, India

(Received 12 March 2024; Accepted 20 September 2024)

https://doi.org/10.36224/ijes.160104

Abstract
A Finite Element Analysis (FEA) code was developed in Python to solve shell problems under plane stress conditions using constant strain triangles. The workflow involves mesh generation, computation of stiffness and assembly matrices, and application of loads and boundary conditions. The analysis provides stresses and displacements for different configurations. To enhance predictive accuracy, a Graph Neural Network (GNN) model was implemented using a Python library. Node features, including coordinates and boundary conditions, were incorporated, while edge indices were computed. The GNN utilized these inputs, with nodal stresses and displacements from FEA serving as target variables in a supervised learning setup. A diverse dataset was generated to capture various configurations. The results showed a high correlation between FEA-derived displacements and stresses and the corresponding GNN predictions for test and validation data. This method has been successfully generalized for different engineering applications, demonstrating its effectiveness in structural analysis. Furthermore, this study extends GNN-based predictions to cases involving different discontinuity shapes, such as circular, elliptical, semicircular, and bend radius geometries, further validating its robustness and applicability across a range of structural problems.

Keywords: Graph Neural Network - GNN, FEM, Surrogate Models, Predictive Modeling.

1. Introduction

- The accurate analysis and prediction of stress and displacement in shell structures is a fundamental aspect of engineering design and analysis. Traditional methods, such as Finite Element Analysis (FEA), have been extensively used for this purpose, offering robust and reliable results. However, the complexity and computational cost of these methods, particularly for large and intricate structures, necessitate the exploration of more efficient and advanced techniques.
- In this study, we present an innovative approach that integrates FEA with Graph Neural Networks (GNNs) to enhance the modelling and prediction capabilities for shell problems under plain stress conditions. Our FEA code, developed in Python, utilizes constant strain triangles to create a mesh, calculate stiffness and assembly matrices, and apply loads and boundary conditions. This traditional FEA setup provides the groundwork for obtaining accurate stresses and displacements across various configurations. To further improve the efficiency and predictive accuracy, we

*Corresponding author
Email address: shaileshkadre@gmail.com (Shailesh Kadre)
ISSN 0976 – 6693.

implemented a GNN model. By leveraging the Python library for GNNs, we formulated node features that include coordinates and boundary conditions, and computed edge indices.

- These features, along with nodal stresses and displacements from the FEA, were used to train the GNN in a supervised learning framework. The creation of a comprehensive dataset, encompassing a wide range of configurations, ensured that the GNN could generalize well across different scenarios.

- The results of our study demonstrate a strong correlation between the FEA-derived stresses and displacements and the predictions made by the GNN model, affirming the potential of this integrated approach. The combination of FEA's accuracy with GNN's predictive capabilities offers a powerful tool for structural analysis, capable of handling complex engineering applications with improved efficiency and reduced computational demands.

- In addition to these advancements, the integration of FEA and GNN opens new avenues for multi-objective optimization in structural engineering. By incorporating this hybrid approach into optimization frameworks, engineers can efficiently explore trade-offs between multiple conflicting objectives, such as minimizing weight while maximizing strength or reducing material cost while ensuring safety and durability. The GNN's rapid prediction capabilities can significantly accelerate the optimization process by quickly evaluating numerous design alternatives, allowing for a more comprehensive exploration of the design space.

- This paper details the methodology, implementation, and validation of our approach, highlighting its application across various engineering scenarios. Our findings suggest that the integration of FEA and GNN not only maintains the robustness of traditional methods but also significantly enhances predictive performance and optimization efficiency, paving the way for more advanced and efficient structural analysis and multi-objective optimization techniques.

2.Literature review

The integration of Graph Neural Networks (GNNs) with Finite Element Analysis (FEA) has emerged as a promising method for enhancing predictive accuracy and computational efficiency in engineering applications. This review synthesizes recent advancements in GNN-based surrogate models for various engineering simulations and optimizations.

Kadre and Tripathi have contributed significantly to surrogate-based optimization and machine learning in engineering design. Their work spans dynamic cluster-based frameworks, mixture surrogate models, and advanced surrogate-assisted multi-objective optimization (RASAMO). Applications include vehicle crashworthiness optimization and aerospace system predictions. Their 2023 study enhances Remaining Useful Life (RUL) predictions for aero-engines using optimized LSTM models. These studies emphasize hybrid surrogate modelling, AI integration, and computational efficiency in optimization. Their contributions pave the way for future advancements in engineering optimization and machine learning-driven predictive modelling.

Bleeg (2020) presents a GNN model for predicting turbine interaction loss, highlighting significant computational time reductions while maintaining high accuracy in fluid dynamics. Feng et al. (2024) extend GNN use to thermal engineering, developing a surrogate model for natural convection in annuli, demonstrating GNNs' capability in complex heat transfer problems. Fu et al. (2023) introduce a boundary-oriented graph embedding approach for FEA surrogate modelling, emphasizing rapid design iterations and accurate FEA result approximations. Jin et al. (2023) combine GNNs with neural operator techniques for high-fidelity mesh-based physics simulations, showcasing enhanced precision and speed. Lee and Lee (2023) apply Graph Convolutional Networks (GCNs) to predict structure-borne noise, demonstrating effective noise pattern predictions. Nourian et al. (2023) use a GNN-based surrogate model for truss structure optimization, enabling efficient design space exploration for optimal configurations. Roznowicz et al. (2024) develop large-scale graph-machine-learning models for 3D flow field prediction in external aerodynamics, underscoring GNNs' scalability with large datasets. Shivaditya et al. (2022) enhance FEA efficiency using GNN-based surrogate models, confirming practical benefits of this integration. Whalen and Mueller (2022) explore reusable surrogate models through graph-based transfer learning on trusses, enhancing model reusability and adaptability across different engineering problems.

Collectively, these studies highlight GNNs' transformative impact on traditional computational methods.

GNNs provide rapid and accurate predictions, improving computational efficiency and opening new pathways for multi-objective optimization and advanced design exploration. The promising results affirm the potential of GNN-based surrogate models to revolutionize engineering simulations and optimizations, offering robust tools for tackling complex problems efficiently.

3. Method of analysis (Experimental procedure/ Numerical experiments)

Following steps summarises the integration of finite element analysis with GNN for prediction.

Finite Element Model based displacement and stress calculations

1. Mesh Generation: Generate triangular mesh for a rectangular plate with a hole using meshpy.triangle library.
2. Plate Stiffness Matrix Calculation:
   - Define material properties (Young's modulus, Poisson's ratio, thickness).
   - Build stiffness matrix K using finite element analysis (FEA) principles.
3. Boundary Conditions and Loading Application:
   - Apply fixed boundary conditions (displacement constraints) to nodes on one side of the plate.
   - Apply uniform load on another side of the plate
4. Displacement and Stress Calculation:
   - Solve for displacements using sparse matrix solver (spla.spsolve) on the reduced stiffness matrix.
   - Calculate stresses based on displacements and geometry of the plate.

Stepwise Process for GNN Model Creation

1. Theoretical Background

Graph Neural Networks (GNNs) are a class of deep learning models designed to operate on graph-structured data. Graphs consist of nodes (representing entities) and edges (representing relationships between entities). GNNs learn to encode both node features and graph structure, making them powerful tools for tasks such as node classification, link prediction, and graph-level prediction.

Key concepts in GNNs include:

- **Node Features**: Attributes associated with each node in the graph, such as coordinates, attributes, or embeddings.
- **Edge Features**: Optional attributes associated with each edge in the graph, such as weights or distances.
- **Message Passing**: A fundamental operation in GNNs where nodes exchange information with their neighbors to update their own features iteratively.
- **Graph Convolutional Layers**: Building blocks of GNNs that aggregate information from neighboring nodes to compute new node representations.
- **Pooling and Aggregation**: Mechanisms used to combine information from multiple nodes or graphs for higher-level predictions.

2. Model Architecture

Graph Convolutional Network (GCN) is a popular architecture for GNNs, suitable for tasks involving node-level predictions on graphs:

- **Input Representation**: Nodes are represented by initial feature vectors, often including spatial coordinates, attributes, or embeddings.
- **Graph Structure**: Edges in the graph are defined by adjacency matrices or edge index tensors, indicating connections between nodes.
- **Layer Stacking**: Multiple GCNConv layers are stacked to allow nodes to aggregate information from their neighbors iteratively.
- **Activation Functions**: ReLU (Rectified Linear Unit) is commonly used after each layer to introduce non-linearity.
- **Output Layer**: The final layer produces predictions (e.g., node classifications or regression outputs) based on the final node representations.

3. Implementation Steps

Here's a step-by-step guide to implement a GCN model for predicting displacements and stresses in a plate with a hole:

Step 1: Data Preparation

- Convert the mesh of the plate into a graph representation using networkx or other graph libraries.
- Define node features (e.g., coordinates) and edge features (if applicable).
- Split the dataset into training, validation, and test sets.

Step 2: Model Definition

- Define a GCN class inheriting from torch.nn.Module.
- Initialize GCNConv layers (torch_geometric.nn.GCNConv) for message passing between nodes.
- Define forward method to specify the data flow through the model. Below code snippet shows the code for GCN model generation.

```
import torch
import torch.nn as nn
from torch_geometric.nn import GCNConv
class GCN(nn.Module):
    def __init__(self, num_node_features, hidden_channels, num_output_features):
        super(GCN, self).__init__()
        self.conv1 = GCNConv(num_node_features, hidden_channels)
        self.conv2 = GCNConv(hidden_channels, hidden_channels)
        self.conv3 = GCNConv(hidden_channels, num_output_features)

    def forward(self, data):
        x, edge_index = data.x, data.edge_index
        x = self.conv1(x, edge_index)
        x = torch.relu(x)
        x = self.conv2(x, edge_index)
        x = torch.relu(x)
        x = self.conv3(x, edge_index)
        return x
```

Step 3: Training and Evaluation

- Define loss function (e.g., Mean Squared Error for regression tasks) and optimizer (e.g., Adam).
- Implement training loop to iterate over batches from DataLoader.
- Evaluate model performance on validation set during training to monitor convergence.
- Evaluate final model performance on test set to report accuracy or loss metrics.

Step 4: Visualization and Analysis

- Visualize displacement and stress contours using matplotlib or other visualization tools.
- Compare predicted results from GCN model with ground truth obtained from FEA for validation.
- Analyze and interpret results to draw conclusions about the effectiveness of the GCN model for the given task.

4. Results and discussion

In this study, following three plate configurations are considered.

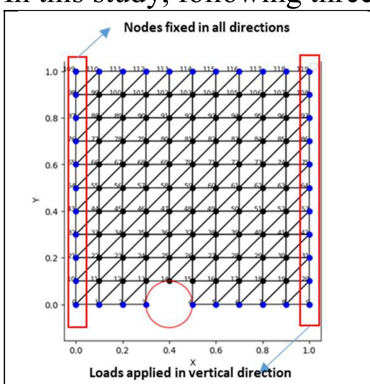


Figure 1 : A Squares plate fixed at left edge and UDL applied on Right edge in **Y Direction**

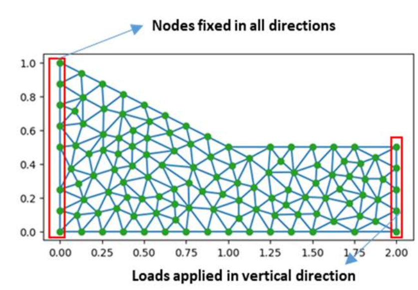


Figure 2: A Trapezoidal plate fixed at left edge and UDL applied on Right edge in **Y Direction**

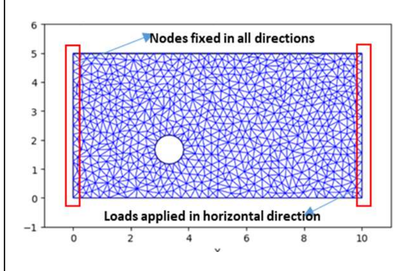


Figure 3 : A Rectangular plate fixed at left edge and UDL applied on Right edge in **X Direction**

As shown in the above Figures 1 to 3, left hand end of each plate is fixed in all direction. Three different shapes of plates are considered: one square plate with a hole, another trapezoidal plate and third one is rectangular plate with a very small hole.

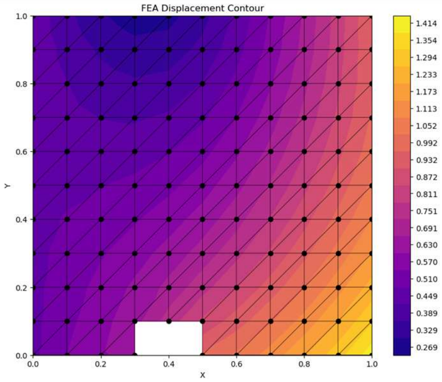


Figure 4 : Calculated displacements from FEM

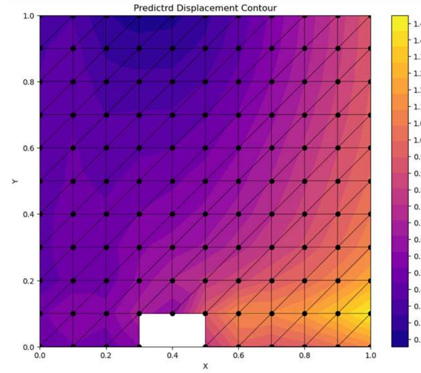


Figure 5 : Predicted displacements by GNN Model

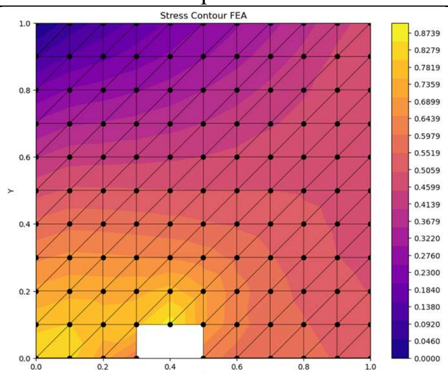


Figure 6: Calculated stresses by FEM

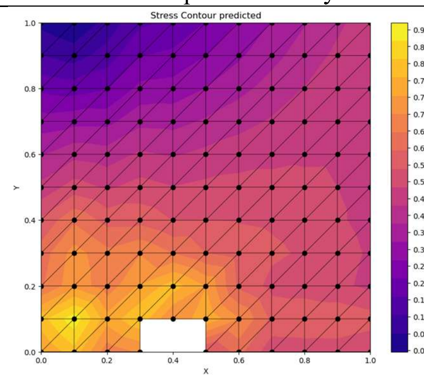


Figure 7: Predicted stresses by GNN Model

From the study of Figures 4 and 5, it can be observed that there is a good agreement between the FEM displacement and GNN predicted displacements. Similarly for the stresses also good correlation is observed for FEM stresses and GNN predicted stresses (Figures 6 and 7)

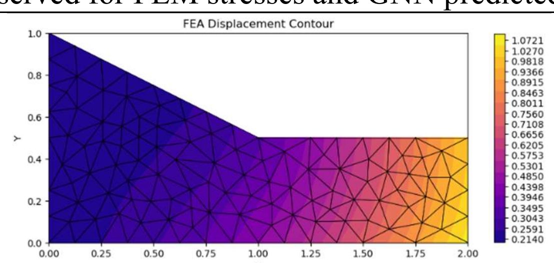


Figure 8 : Calculated displacements from FEM

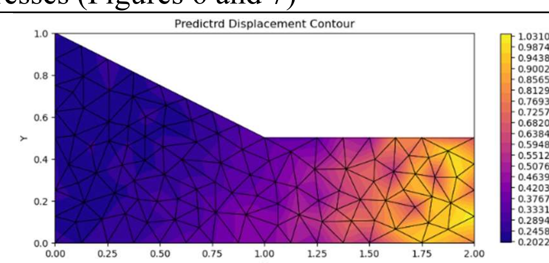


Figure 9 : Predicted displacements by GNN Model

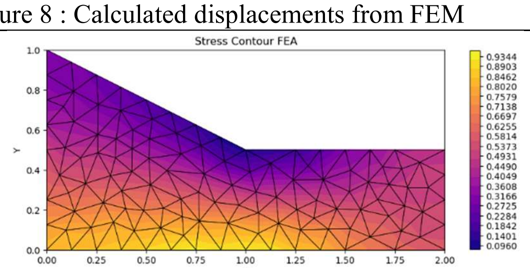


Figure10: Calculated stresses by FEM

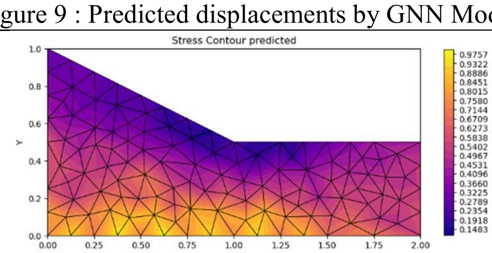


Figure 11: Predicted stresses by GNN Model

From the study of Figures 8 and 9, it can be observed that there is a good agreement between the FEM displacement and GNN predicted displacements. Similarly for the stresses also good correlation is observed for FEM stresses and GNN predicted stresses (Figures 10 and 11)

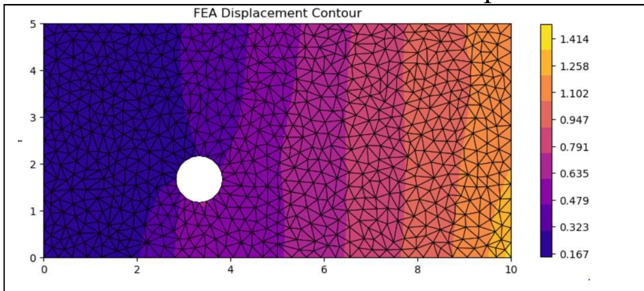


Figure 12 : Calculated displacements from FEM

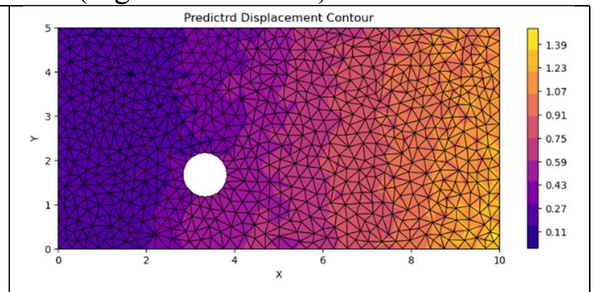


Figure 13 : Predicted displacements by GNN Model

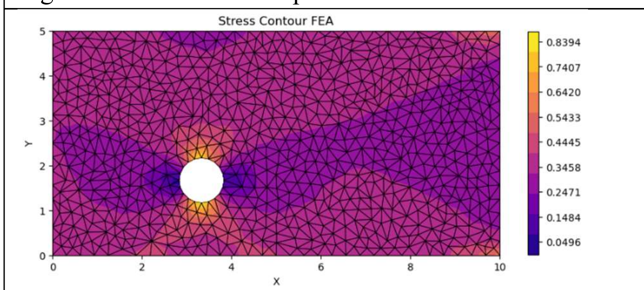


Figure 14: Calculated stresses by FEM

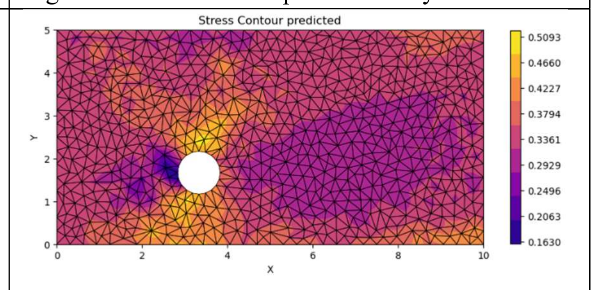
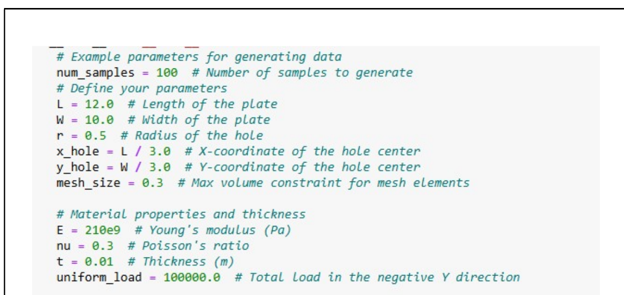


Figure 15: Predicted stresses by GNN Model

From the study of Figures 12 and 13, it can be observed that there is a good agreement between the FEM displacement and GNN predicted displacements. Similarly, for the stresses also good correlation is observed for FEM stresses and GNN predicted stresses (Figures 14 and 15), there is some difference in the stress patterns and magnitude. This may be due to a smaller number of samples in the dataset. The study is extended for the elliptical hole in the plate. Following Figure 16 shows the details of various parameters used for training datasets. The details of Training and Validation loss are depicted in Figure 17 which clearly indicates that the convergence is reached after 50 epochs.



```
# Example parameters for generating data
num_samples = 100  # Number of samples to generate
# Define your parameters
L = 12.0  # Length of the plate
W = 10.0  # Width of the plate
r = 0.5  # Radius of the hole
x_hole = L / 3.0  # X-coordinate of the hole center
y_hole = W / 3.0  # Y-coordinate of the hole center
mesh_size = 0.3  # Max volume constraint for mesh elements

# Material properties and thickness
E = 210e9  # Young's modulus (Pa)
nu = 0.3  # Poisson's ratio
t = 0.01  # Thickness (m)
uniform_load = 100000.0  # Total load in the negative Y direction
```

Figure 16: Details of various parameters used for training datasets

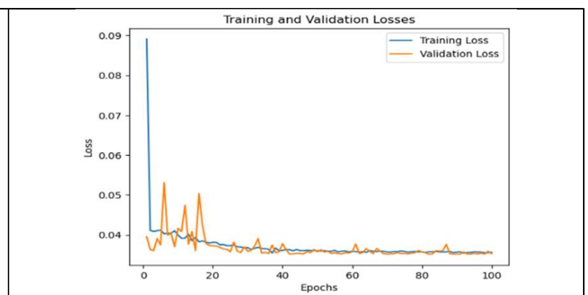


Figure 17: Training and validation loss curve

From the study of Figures 16 and 17, it can be observed that there is a good agreement between the FEM displacement and GNN predicted displacements for elliptical holes too. Similarly, for the stress “patterns” also good correlation is observed for FEM stresses and GNN predicted stresses (Figures 14 and 15), there is some difference in the stress magnitude. This may be due to a smaller number of samples in the dataset.

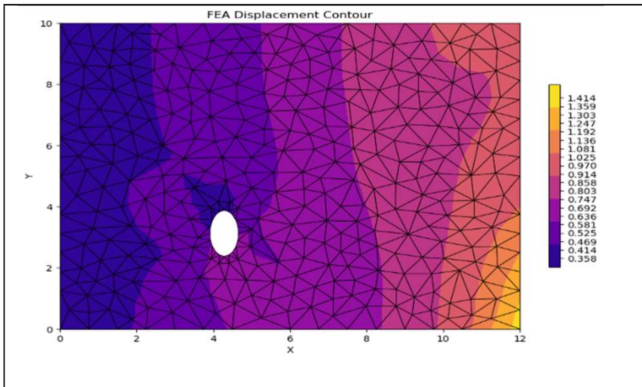


Figure 18: Calculated displacements from FEM

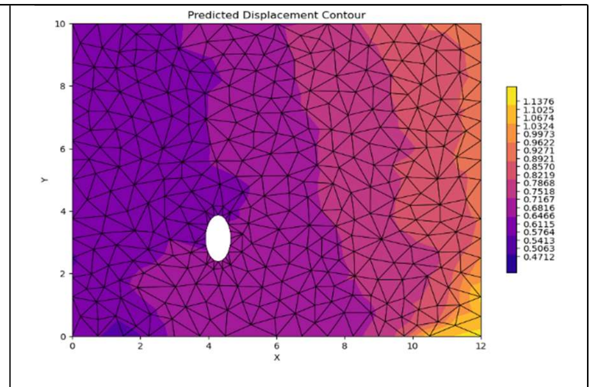


Figure 19: Predicted displacements by GNN

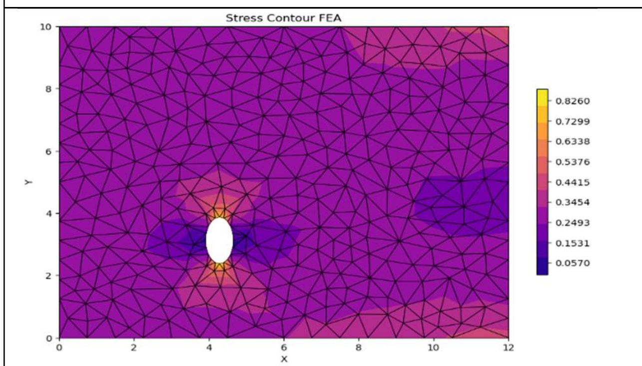


Figure 20: Calculated stresses by FEM

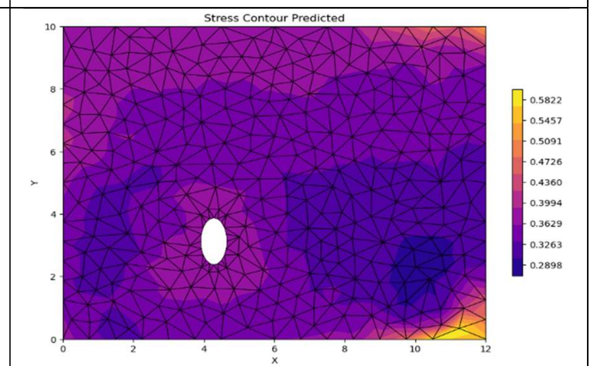


Figure 21: Predicted stresses by GNN Model

This study successfully applied Graph Neural Networks (GNNs) to predict structural responses, specifically displacements and stresses, in a plate with a hole. A strong correlation was observed between Finite Element Analysis (FEA) results and GNN predictions, demonstrating the model's effectiveness in capturing complex spatial dependencies and structural behaviors.

Key Findings:

- **High Prediction Accuracy:** The GNN effectively learned from graph-based mesh representations, accurately predicting displacements and stresses for various hole configurations.
- **Robust Validation:** Predictions were extensively validated against FEA results, confirming the model's reliability across different test cases.
- **Generalization Capability:** The model successfully predicted structural responses for unseen configurations, showcasing its ability to generalize beyond training data.
- **Adaptive Learning:** The GNN framework dynamically adapted to different mesh densities and boundary conditions, improving flexibility in real-world applications.
- **Scalability:** The approach is scalable to complex geometries, making it suitable for more intricate engineering problems involving discontinuities.
- **Practical Engineering Utility:** By leveraging data-driven learning, GNNs offer a viable alternative to traditional FEA, enabling faster and more efficient structural analysis for design and optimization.

This study highlights the potential of GNNs in structural mechanics, paving the way for their broader adoption in engineering applications where computational efficiency and predictive accuracy are crucial.

I. 5. Conclusion and Future Scope

This study underscores the transformative potential of integrating Graph Neural Networks (GNNs) with Finite Element Analysis (FEA) for predictive modeling and structural optimization. By achieving a high correlation with FEA-calculated responses, GNNs have proven to be highly accurate in capturing complex spatial dependencies and nonlinear structural behaviors. Their ability to serve as efficient surrogate models significantly reduces computational costs while maintaining precision, making them a powerful tool for accelerating engineering workflows.

The integration of GNNs into multi-objective optimization frameworks represents a major advancement in structural design. As surrogate models, GNNs can effectively replace computationally expensive FEA simulations, enabling rapid design iterations while optimizing multiple conflicting objectives such as minimizing weight, maximizing structural integrity, and reducing stress concentrations. This shift not only enhances computational efficiency but also fosters innovative approaches to structural analysis and design.

Future research should focus on further improving the robustness and scalability of GNNs, particularly in handling complex geometries and diverse loading conditions. Advancing their application in multi-objective optimization can revolutionize how engineers approach design challenges, bridging the gap between traditional simulation methods and modern machine learning-driven solutions. The integration of GNNs with FEM paves the way for a new era of efficient, data-driven structural engineering.

6. Conclusión

It is concluded that the significant effect of the problem-solving method on mathematical competence in students was achieved with a 5% probability of error and a 95% confidence level.
The significant effect of the problem-solving method on the achievement of mathematical competence was demonstrated, with a considerable increase of 236.13% in the output evaluation of the experimental group with a 5% probability of error and a 95% confidence level, due to the use of the 4 phases of the problem-solving method, which are: understanding the problem, conceiving a plan, executing the plan and examining the solution obtained.
The significant effect of the problem-solving method in the achievement of the mathematical competence of number and relations was demonstrated, with a considerable increase of 230.77% in the output evaluation of the experimental group with a 5% probability of error and a 95% confidence level, this was due to the use of the 4 phases of the problem-solving method, which are: understanding the problem, conceiving a plan, executing the plan and examining the solution obtained.

References

1. Bleeg, J., (2020) A Graph Neural Network Surrogate Model for the Prediction of Turbine Interaction Loss. *Journal of Physics: Conference Series*, 16186, p.062054.
2. Feng, F., Li, Y.-B., Chen, Z.-H., Wu, W.-T., Peng, J.-Z. and Mei, M., (2024) Graph convolution network-based surrogate model for natural convection in annuli. *Case Studies in Thermal Engineering*, 57, p.104330.

3. Fu, X., Zhou, F., Peddireddy, D., Kang, Z., Jun, M.B.-G. and Aggarwal, V., (2023) An finite element analysis surrogate model with boundary oriented graph embedding approach for rapid design. *Journal of Computational Design and Engineering*, 103, pp.1026–1046.
4. Jin, Z., Zheng, B., Kim, C. and Gu, G.X., (2023) Leveraging graph neural networks and neural operator techniques for high-fidelity mesh-based physics simulations. *APL Machine Learning*, 14.
5. Lee, S.-Y. and Lee, S.-K., (2023) Graph Convolutional Network Surrogate Model for Mesh-Based Structure-Borne Noise Simulation. *Applied Sciences*, 1316, p.9079.
6. Nourian, N., El-Badry, M. and Jamshidi, M., (2023) Design Optimization of Truss Structures Using a Graph Neural Network-Based Surrogate Model. *Algorithms*, 168, p.380.
7. Roznowicz, D., Stabile, G., Demo, N., Fransos, D. and Rozza, G., (2024) Large-scale graph-machine-learning surrogate models for 3D-flowfield prediction in external aerodynamics. *Advanced Modeling and Simulation in Engineering Sciences*, 111, p.6.
8. S. S. Kadre and V. K. Tripathi, "Engineering design optimization by the dynamic cluster-based framework using mixture surrogates", International Journal of Innovative Technology and Exploring Engineering, ISSN: 2278-3075,Volume-8, Issue-11S, September 2019, Page 979- 988, 2019.
9. S. S. Kadre and V. K. Tripathi, "Mixture Surrogate Models for Multi-Objective Optimization", International Journal of Engineering Sciences (ISSN 0976 – 6693), 13(1) 21 – 28, 2020.
10. S. S. Kadre and V. K. Tripathi, "Advanced Surrogate Models for Design Optimization," International Journal of Engineering Sciences (ISSN 0976 – 6693), 9(2), 66 – 73, 2016.
11. S. S. Kadre and V. K. Tripathi,. "Multi-Objective Optimization of Vehicle Crashworthiness by SAMO: Advanced Surrogate Model-based Algorithm", Journal of Advanced Research in Dynamical and Control Systems, ISSN: 0193-4120, 2020, Volume 12, issue- 6, Pp 2278- 2288
12. S. S. Kadre, V. K. Tripathi,, Hoi Huynh Tan. "Refined Advanced Surrogate Assisted Multi-Objective Optimization Algorithm- RASAMO.& quot; Elementary Education Online 20 (2021), 2374-2388. doi:10.17051/ilkonline.2021.05.259
13. S. S. Kadre, R.G. Paspula, Improved RUL Predictions of Aero- Engines by Hyper-Parameter Optimization of LSTM Neural Network Models. International Journal for Research in Applied Science & Engineering Technology (IJRASET) Volume 11 Issue III, Mar 2023.
14. Shivaditya, M.V., Alves, J., Bugiotti, F. and Magoules, F., (2022) Graph Neural Network-based Surrogate Models for Finite Element Analysis. In: *2022 21st International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES)*. IEEE, pp.54–57.
15. Whalen, E. and Mueller, C., (2022) Toward Reusable Surrogate Models: Graph-Based Transfer Learning on Trusses. *Journal of Mechanical Design*, 1442.
16. S. S. Kadre and V. K. Tripathi, "Engineering design optimization by the dynamic cluster-based framework using mixture surrogates", International Journal of Innovative Technology and Exploring Engineering, ISSN: 2278-3075,Volume-8, Issue-11S, September 2019, Page 979- 988, 2019.