Grenze International Journal of Engineering and Technology, January Issue

ML Visualizer

Surekha M[1], Pranjali Jain[2], Shushila Vishwakarma[3] and Tathagat Tripathi[4]
[1-4]Department of Computer Science and Engineering, JSS Academy of Technical Education, Noida, Uttar Pradesh, India
Email: surekha@jssaten.ac.in, pranjalijain2004@gmail.com, shushila257@gmail.com, tathagat7374@gmail.com

***Abstract*— Through an interactive graphical user interface that facilitates understandable, transparent, and effective model training for practical applications like counterfeit banknote detection, this project significantly contributes to closing the gap between sophisticated machine learning algorithms and user accessibility. The study presents a Machine Learning Graphical User Interface (ML GUI) frame- work developed for both counterfeit currency recognition and general-purpose classification tasks using statistical parameters including variance, skewness, kurtosis, and entropy. Without the need for complex coding skills, the system offers an intuitive environment for data preparation, feature visualization, model training, and performance evaluation. It incorporates multiple algorithms—Support Vector Machine (SVM), Decision Tree (DT), Random Forest (RF), K-Nearest Neighbors (KNN), Linear Regression (LR), Logistic Regression (LogR), and Multilayer Perceptron (MLP) to ensure diverse and reliable classification. Model effectiveness is evaluated using accuracy, precision, recall, and F1-score, while visual tools such as confusion matrices enhance interpretability. Experimental analysis shows that Decision Trees and Random Forests deliver high accuracy and transparency, SVM performs well on linearly separable data, and MLP effectively models complex non-linear relationships. Developed using Scikit-learn and TensorFlow, the GUI framework is scalable, educationally valuable, and adaptable for real-time fraud detection, making machine learning more accessible for both learners and professionals.**

***Keywords*— Machine Learning, GUI, Data Visualization, Skewness, Kurtosis, Entropy.**

I. INTRODUCTION

One of the 21st century's most revolutionary technologies is machine learning (ML), which propels improvements in automation, predictive analytics, and intelligent decision-making across a range of sectors. However, its adoption is often limited by the technical expertise required to develop and train models using complex frameworks such as TensorFlow, PyTorch, and Scikit-learn. To address this challenge, Graphical User Interface (GUI)-based ML tools have emerged as an accessible solution, providing visual, drag-and-drop interfaces for data upload, algorithm selection, model training, and evaluation

A well-designed GUI-based ML system typically includes:

- Data input and preprocessing (CSV upload, missing value handling)
- Statistical visualization (scatter plots, histograms, skewness, kurtosis)
- Algorithm selection (classification and regression)
- Model training and evaluation (accuracy, confusion matrix, precision, recall)
- User interactivity (parameter tuning, visual feedback)

Such interfaces bridge the gap between theoretical data science concepts and practical model implementation enabling users to intuitively construct, train, and evaluate ML models.

Grenze ID: 01.GIJET.12.1.357

This review paper focuses on GUI-driven ML platforms that embed the entire machine learning workflow within a visual environment, incorporating algorithms such as Support Vector Machine (SVM), Logistic Regression (LogR), Linear Regression (LR), K-Nearest Neighbors (KNN), Random Forest (RF), Decision Tree (DT), and Multilayer Perceptron (MLP). These algorithms collectively support preprocessing, visualization, and performance evaluation using metrics such as precision, accuracy, recall, and F1-score. Moreover, interpretability tools such as confusion matrices and feature importance plots enhance transparency and explainability, enabling data-driven decision-making.

GUI-based ML systems have demonstrated practical relevance in diverse domains. Financial institutions can employ them for real-time fraud detection and counter- feit banknote authentication, while educators and researchers can leverage them for interactive experimentation and teaching. Frameworks built using libraries such as Scikit-learn and TensorFlow further ensure scalability and robustness.

In summary, GUI-based machine learning environments democratize artificial intel- ligence by combining the power of advanced algorithms with user-centric design. The aim of this review is to evaluate and put into perspective the research and develop- ment initiatives in this field, including the Bank Note Authentication ML-GUI using PyQt5, and to highlight how such innovations simplify and strengthen the process of intelligent decision-making.

II. Literature Review

Interactive visualization has become a crucial component in enhancing machine learning (ML) workflows, improving both interpretability and user engagement.

- Li et al. [1] explored this integration by developing interactive machine learning systems that allow users to iteratively refine models through visualization. Their approach focuses on real-time visual feedback, enabling users to understand model behavior and make adjustments dynamically, which is particularly valuable when dealing with complex datasets or models.
- Spinner et al. [2] expanded on this by creating explAIner, an extensive visual analytics platform for explainable machine learning. This system facilitates users in interacting with ML models by providing clear visual representations of model predictions, feature importance, and decision boundaries. By making the model's inner workings more transparent, explAIner helps users trust and validate complex machine learning models more effectively.
- For non-expert users, the process of model training and hyperparameter tuning can be daunting. Shakeel et al. [3] addressed this by proposing an Automated Model Training (AMT) GUI, which automates these steps via an easy-to-use interface. This tool not only simplifies the model-building process but also integrates visualization components to communicate model performance and optimization progress, making machine learning more accessible.
- Wang et al. [4] provided a broad overview of how recent machine learning advances can be leveraged to improve data visualization itself. Their survey highlights tech- niques for automated feature extraction and pattern recognition within visual data, enabling more intelligent and adaptive visualization systems that assist in exploratory data analysis and decision-making.
- Feature engineering is a critical yet often time-consuming part of machine learning. FeatureEnVi, a visual analytics system that assists users in choosing and extracting significant features, was proposed by Chatzimparmpas et al. [5]. Their approach combines stepwise selection techniques with semi-automatic extraction methods, helping users interactively discover and evaluate features that improve model performance.
- Wang et al. [6] introduced HypoML, a framework that integrates hypothesis-driven evaluation within ML model analysis. By allowing domain experts to test specific hypotheses about model behavior visually, HypoML bridges the gap between tech- nical ML evaluation metrics and domain-specific understanding, facilitating more targeted model assessments.
- Comparing different deep learning architectures can be complex due to their high dimensionality and layered structures. Xuan et al. [7] addressed this with a visual analytics tool called VAC-CNN was designed for comparing and contrasting convolu- tional neural networks (CNNs). VAC-CNN enables detailed, layer-wise inspection of CNN features and activations, aiding researchers in understanding model differences and optimizing architectures.
- Misclassification is a common challenge in ML model development. Nguyen et al. [8] developed VisMCA, a system for visually analyzing and correcting misclassi- fied instances. Their approach integrates human-in-the-loop corrections with visual analytics to iteratively improve model accuracy and reduce errors.

- In the specific application domain of banknote authentication, machine learning methods have been used extensively to detect counterfeit currency. Sonje and Sonje [9] evaluated various advanced classification algorithms tailored for this task, demonstrating the effectiveness of machine learning in practical authentication scenarios.
- SWang et al. [10] examined the effectiveness of artificial neural networks and logistic regression for banknote verification, offering information on the trade-offs between neural network-based and conventional techniques.
- WRajitha et al. [11] extended this research by utilizing hybrid and ensemble machine learning models, which integrate several algorithms to enhance classification precision and robustness against counterfeit notes.
- Addressing usability in machine learning tools, Yamani et al. [12] proposed a set of heuristics aimed at enhancing GUI-based machine learning tools for inexperienced users. Their guidelines help in creating more intuitive and effective interfaces, reducing the learning curve for users new to machine learning.
- On the front of interpretability, Doshi-Velez and Kim [13] emphasized the importance of developing a rigorous science around interpretable machine learning. They argued for standardized evaluation metrics and formal methodologies to ensure that interpretability claims are meaningful and reproducible.
- Jim´enez-Luna et al. [14] applied explainable AI in the context of drug discovery, showcasing how transparent machine learning models can accelerate pharmaceutical research by providing insights into molecular mechanisms and guiding experimental design.
- More recent contributions include ExperimentLens by Maroulis et al. [15], a system designed for managing ML experiments with a strong focus on visual analytics and explainability. This tool helps researchers track, compare, and understand different ML experiments interactively.
- Chatzimparmpas et al. [16] contributed techniques for visual analytics aimed at explainable and trustworthy AI, focusing on methods that enhance users' trust by making AI decisions more transparent and understandable.
- Kondaveeti et al. [17] evaluated deep learning models using explainable AI methods, improving the assessment of model reliability and identifying weaknesses that might not be apparent through traditional accuracy metrics.
- Focusing again on banknote authentication, Oyedele [18] conducted a thorough analysis of machine learning techniques applied to this domain, while Pranathi et al. [19] surveyed recent advancements emphasizing feature extraction and classification, helping consolidate the current state of research.
- Lastly, Ravichandran et al. [20] proposed a system for detecting counterfeit banknotes based on deep learning, employing convolutional neural networks. Their model demonstrated superior accuracy and robustness, highlighting the potential of deep learning techniques in security-critical applications.

III. RESEARCH GAP

- Despite progress in machine learning algorithms and visualization techniques, there remains a lack of integrated platforms that combine user-friendly interfaces, algorithmic flexibility, and interpretability for complete model development.
- Most existing frameworks emphasize either algorithm optimization or visualization independently, without offering a cohesive, interactive environment for non-programmers to experiment, visualize, and interpret models in real-time.
- Studies demonstrating the performance of algorithms such as SVM, Decision Tree, Random Forest, and Neural Networks for classification tasks (e.g., counterfeit banknote detection) are largely code-based or use static dashboards with no interactive feedback or comparative model evaluation in a GUI.
- The absence of interpretability features—such as feature importance analysis, confusion matrix visualization, or statistical indicators (skewness, kurtosis, entropy)—in accessible GUI environments limits model transparency and makes assessment of feature contributions difficult.
- Although Explainable AI (XAI) and Human-in-the-Loop (HITL) systems are gaining attention, very few GUI-based tools have effectively integrated these paradigms for real-world domains such as financial fraud or counterfeit detection.
- Current visual analytics frameworks are generally designed for advanced researchers rather than educators, students, or industry professionals who require an intuitive, learning-oriented tool.

- he identified research gap lies in developing an interactive, explainable, and domain-adaptable GUI framework that unifies data preprocessing, visualization, model selection, parameter tuning, and performance evaluation to enhance both accessibility and trust in machine learning applications.

IV. Proposed Model

A. Model Specification

The proposed Machine Learning Visualizer (a GUI-based tool for Data Visualization and Machine Learning Model Training) is a modular, user-centric platform designed to bridge the gap between technical machine learning processes and end-user usability. It integrates powerful machine learning capabilities with an intuitive interface and a scalable backend infrastructure. The architecture is structured into three primary layers:

Core Machine Learning Component

This module handles all aspects of machine learning operations, including:

- Data Preprocessing: The Data Preprocessing module handles data cleaning, feature encoding, normalization, and dataset partitioning (train/test/validation).
- Model Training and Selection: Supports a large variety of machine learning algorithms, such as neural networks, ensemble techniques like Random Forest and Gradient Boosting, and traditional models like Decision Trees and SVMs.
- Inference and Evaluation: Once trained, models can be used to make predictions, and performance metrics such as accuracy, F1-score, and confusion matrix are generated for analysis.
- Advanced Techniques: Incorporates ensemble learning and deep learning tech- niques to improve the accuracy and resilience of models, especially on complex datasets.

User Interaction Layer

The front-end GUI layer ensures a seamless user experience through:

- Interactive Visualizations: Heatmaps, box plots, scatter plots, and histograms are examples of real-time data visualization tools.
- User Inputs: Simple form-based interfaces for hyperparameter selection, file upload (via drag-and-drop), and triggering training processes.
- Model Output Display: Visual and textual representation of model performance metrics, prediction outputs, and evaluation graphs.
- Feedback Mechanism: Enables users to iteratively refine model inputs and configurations based on output insights.

Backend Management System

The backend layer supports operational, computational, and storage requirements, including:

- Database Integration: Connects with various data sources (SQL/NoSQL) for flexible data ingestion and persistence.
- API Architecture: RESTful APIs enable modular communication between the front-end and ML engine, ensuring separation of concerns.
- Cloud Deployment: Designed to be cloud-ready (e.g., AWS, Azure, GCP) for scalability, high availability, and performance optimization.
- Model Lifecycle Management: Supports saving, loading, and deploying trained models, facilitating reproducibility and reuse.

B. Technology used

The implementation of the system has been structured into several components encom- passing machine learning algorithms, graphical user interface development, database systems, evaluation mechanisms, and version control tools. All of these are essential to the project's fulfillment. The selection of technologies was driven by performance requirements, scalability, and ease of integration within the overall architecture. By employing widely adopted frameworks and robust software stacks, the system ensures reliability, maintainability, and efficient collaboration throughout the development lifecycle.

Machine Learning Algorithms

Classification Models: Some of the models used include Random Forest, Gradient Boosting, SVM, MLP, Linear Regression, Logistic Regression, and Decision Trees (DT). These models are applicable to a variety of activities including the analysis of user behavior, data categorization, and predictive modeling. In this context, they have

been utilized for training and validation using appropriate datasets for specific applications such as credit scoring and bankruptcy prediction.

TABLE I: TECHNOLOGY STACK AND COMPONENTS

Component	Technology / Library	Purpose
Programming Language	Python	Core logic, ML, and GUI
GUI Framework Data Handling	Tkinter / PyQt / Streamlit pandas, numpy	User interaction, file upload, parame- ter input, and visualization
Visualization ML Algorithms	matplotlib, seaborn scikit-learn	CSV loading, preprocessing, numerical operations
Model Saving	pickle / joblib	Plotting graphs, exploratory data analysis
Version Control	Git / GitHub	Training SVM, Decision Tree, Naive Bayes models
		Save and load trained models
		Code management and updates

Deep Learning Models: Neural networks such as Multilayer Perceptrons (MLPs) and pretrained transformers like BERT are used when tasks require high- dimensional data analysis, such as Natural Language Processing (NLP) or the generation of real-time feedback.
Ensemble Methods: Ensemble models like AdaBoost and LogitBoost are used to improve prediction accuracy by combining multiple outputs from weak classifiers, thus enhancing model stability and robustness.

Graphical User Interface Development Tools
Frameworks: Tkinter, PyQt, and web-based frameworks like React.js are used to design dynamic and interactive user interfaces, enabling smooth user interaction and control over model parameters.
Visualization Libraries: Matplotlib has been integrated to provide real-time visual feedback and analytical insights to users. It supports plots such as line graphs, histograms, and scatter plots to visualize model performance and dataset trends.

Database Systems
For future scalability, SQL-based systems such as MySQL and PostgreSQL can be combined with NoSQL alternatives to support the storage of both structured and unstructured data. For large-scale and distributed data management, cloud-based ser- vices such as AWS, Google Cloud, and Microsoft Azure can be utilized, enabling real-time synchronization and collaboration as required.

Evaluation Mechanisms
The following metrics are employed to assess model performance and ensure reliable predictions:
- Accuracy: The proportion of accurate predictions to all predictions is the measure of the overall accuracy of predictions.(TP + TN).
- Precision: Represents the ratio of accurately identified positive cases out of all predicted positives. A high precision value indicates a lower number of false positives.
- Recall (Sensitivity): Measures how well the model identifies actual positives. A high recall means most positive cases are correctly detected.
- F1-Score: The harmonic mean of precision and recall is represented by the F1-score. A high F1-score indicates a favorable equilibrium between precision and recall.

Development Tools and Version Control
Development Environment: Visual Studio Code has been used to develop and test Python-based scripts, ensuring efficient debugging and modular design.
Version Control: Git and GitHub have been used for collaborative development, tracking changes, and managing different versions of the project.

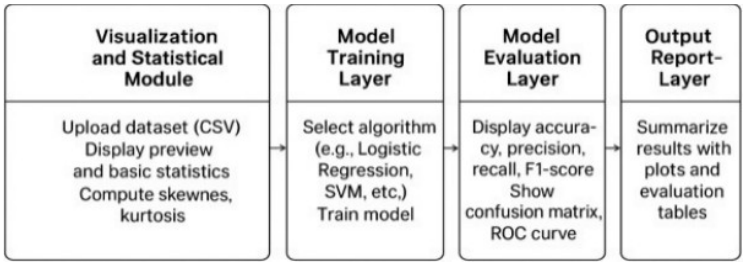

Fig. 1 Work Flow of the Proposed System

Work flow: The proposed system operates through a structured workflow consisting of five key layers. Each layer has distinct responsibilities that contribute to the system's efficiency, interpretability, and ease of use.

- Data Input Layer: The Data Input Layer serves as the primary interface between the user and the ML system. Its main role is to allow users to provide the raw data for analysis.

Key Features:

- Data Upload: Users can upload datasets, usually in CSV, Excel, or similar formats.
- Preview Functionality: The system displays a preview of the dataset (e.g., first 5–10 rows) to ensure correct file upload.
- Basic Statistical Info: Automatically computes descriptive statistics including mean, median, mode, standard deviation, variance, minimum, maximum, and number of missing values per column.

Purpose: This layer ensures that the dataset is correctly formatted and pro- vides a quick overview of the data's characteristics, which is crucial for downstream preprocessing and model selection.

- Visualization and Statistical Module:This module allows users to explore and understand the dataset visually. Visualization helps identify patterns, trends, and anomalies.

Key Features:

- Graphs and Plots:
- Histograms: Show the frequency distribution of numeric variables.
- Boxplots: Detect outliers and visualize data spread.
- Scatter Plots: Show relationships between two variables.
- Pairplots/Correlation Matrices: Highlight correlations among multiple features.
- Statistical Measures:
- Skewness: Measures asymmetry of the data distribution.If the skew is positive, the tail is on the right; if it is negative, the tail is on the left.
- Kurtosis: Measures "tailedness" or the tendency for extreme values. High kurtosis indicates heavy tails.

Purpose: This layer allows the user to visually inspect data quality and dis- tributions, making it easier to identify features that may need normalization, transformation, or removal.

- Model Training Layer

This is the core of the ML system, where users train machine learning models on the dataset.

Key Features:

- Algorithm Selection: Users select an algorithm from a dropdown menu, e.g., Logistic Regression, SVM, Decision Tree, Random Forest, K-Nearest Neighbors (KNN), Neural Networks, and Linear Regression.
- Data Splitting: Automatically divides data into training and testing sets (commonly 70–30 or 80–20 split).
- Model Training: Utilizes libraries such as Scikit-learn for model training, handling preprocessing (e.g., normalization, handling missing values) internally if enabled.
- Hyperparameter Selection: Allows tuning parameters such as tree depth, number of estimators, and regularization terms.

Purpose: This layer makes machine learning accessible for non-experts by providing a simple interface for model building without requiring programming expertise.

- Model Evaluation Layer

Following training, the model's functionality must be evaluated to ensure generaliza- tion to unseen data.

Key Features: Metrics of performance

- Accuracy – overall correctness of predictions.
- Precision – accuracy among positive predictions.
- Recall (Sensitivity) – capacity to recognize true positives.
- F1-Score – harmonic mean of precision and recall, useful for imbalanced datasets.
- Confusion Matrix – visual representation of true/false positives and negatives.
- ROC Curve – compromise between the rates of false positives and true positives.
- AUC Score – area under ROC curve, which shows how well the model can differentiate between classes.

Purpose: This layer provides insights into model effectiveness and allows users to compare multiple models visually and numerically.

- Output and Reporting Layer

This layer summarizes the results and generates interpretable reports for end users.

Key Features:

- Visualization of Results: Displays feature importance plots, predicted vs. actual plots, and performance tables comparing multiple models.
- Export Options: Allows exporting results and reports in PDF, Excel, or image formats.

- Recommendations: Optionally suggests the best-performing model based on chosen metrics.

Purpose: The output layer translates complex machine learning results into action- able insights for decision-making, enhancing system usability for business and research purposes.

V. Conclusion

The platform under examination effectively closes the gap between sophisticated cod- ing and the real-world implementation of data science methods. By providing an intuitive interface for data visualization and machine learning, it not only serves as an effective learning tool but also as a powerful rapid prototyping environment. This project demonstrates the potential of creating user-centric software that makes sophisticated analytical methods accessible to a broader audience, fostering greater innovation and collaboration in the data science community.

Future work will focus on:

- Implementation of contextual filtering to eliminate excessive and redundant predic- tions, further improving accuracy.
- Expanding the dataset to include general seasonal and regional variations for enhanced generalization.
- Development of real-time video-based detection systems for continuous monitoring via dashcams, drones, or surveillance feeds.
- Exploring multi-modal approaches by fusing visual data with GPS or LiDAR for improved pothole detection accuracy.
- Creating a community-built mobile app for collaborative reporting and validation.
- Deployment on edge devices for on-the-fly detection in vehicles or road inspection robots.
- Utilizing semi-supervised learning to reduce human effort in manual labeling and continuously improve the model over time.
- Developing richer deployment pipelines through lightweight models designed for resource-constrained environments like rural areas or mobile platforms.

References

[1] H. Li et al., “Interactive Machine Learning by Visualization,” IEEE International Conference on Big Data, 2019.

[2] T. Spinner et al., “explAIner: A Visual Analytics Framework for Interactive and Explainable Machine Learning,” IEEE Transactions on Visualization and Computer Graphics, vol. 26, no. 1, pp. 106–116, Jan. 2020.

[3] M. B. Shakeel et al., “Automated Model Training (AMT) GUI,” arXiv, arXiv:2311.

[4] Q. Wang, Z. Chen, Y. Wang, and H. Qu, “ML4VIS: Applying Machine Learning Advances to Data Visualization,” arXiv, arXiv:2002.05271, 202013808, 2023.

[5] A. Chatzimparmpas, R. Martins, K. Kucher, and A. Kerren, “FeatureEnVi: Visual Analytics for Feature Engineering Using Stepwise Selection and Semi- Automatic Extraction Approaches,” IEEE Transactions on Visualization and Computer Graphics, vol. 27, no. 2, pp. 1710–1720, 2021.

[6] Q. Wang et al., “HypoML: Visual Analysis for Hypothesis-Based Evaluation of Machine Learning Models,” IEEE Transactions on Visualization and Computer Graphics, vol. 26, no. 1, pp. 154–164, 2020.

[7] X. Xuan, X. Zhang, O.-H. Kwon, and K.-L. Ma, “VAC-CNN: A Visual Analyt- ics System for Comparative Studies of Deep Convolutional Neural Networks,” IEEE Transactions on Visualization and Computer Graphics, vol. 27, no. 2, pp. 1691–1701, 2021.

[8] H. N. Nguyen, J. Gonzalez, J. Guo, N. V. T. Nguyen, and T. Dang, “VisMCA: A Visual Analytics System for Misclassification Correction and Analysis,” IEEE Transactions on Visualization and Computer Graphics, vol. 27, no. 2, pp. 1740–1751, 2021.

[9] D. Sonje and M. Sonje, “Bank Note Authentication and Classification Using Advanced Machine Learning Algorithms,” Journal of Science and Technology (JST), vol. 6, no. 3, pp. 92–99, 2021.

[10] A. Wang, G. Goldsztein, and Z. Sun, “Banknote Authentication Using Logistic Regression and Artificial Neural Networks,” Journal of Student Research, vol. 11, no. 1, pp. 112–118, 2022.

[11] A. Rajitha et al., “Classification and Detection of Banknotes Using Machine Learning,” IJRASET, vol. 12, no. 4, pp. 340–345, Apr. 2024.

[12] A. Yamani et al., “Heuristics for Improving the Usability of GUI-Based Machine Learning Tools for Novice Users,” arXiv, arXiv:2405.08313, 2024.

[13] F. Doshi-Velez and B. Kim, “Towards a Rigorous Science of Interpretable Machine Learning,” arXiv, arXiv:1702.08608v3, updated 2020.

[14] J. Jim´enez-Luna, F. Grisoni, and G. Schneider, “Drug Discovery with Explainable Artificial Intelligence,” Nature Machine Intelligence, vol. 2, pp. 573–584, 2020.

[15] S. Maroulis et al., “ExperimentLens: Interactive Visual Analytics and Explain- ability for ML Experiment Management,” Proceedings of VLDB Workshops (GuideAI), 2025.

[16] A. Chatzimparmpas et al., “Visual Analytics for Explainable and Trustworthy Artificial Intelligence,” arXiv, arXiv:2507.10240, 2025.
[17] H. K. Kondaveeti et al., “Evaluation of Deep Learning Models Using Explainable AI,” Scientific Reports (Nature), vol. 15, no. 7, pp. 1–14, 2025.
[18] O. Oyedele, “Analysis of Banknote Authentication Using Machine Learning Techniques,” ResearchGate preprint, 2025.
[19] B. Pranathi et al., “A Study on Banknote Authentication Using Machine Learn- ing,” International Advanced Research Journal in Science, Engineering and Technology (IARJSET), vol. 12, no. 3, pp. 198–204, 2025.
[20] R. Ravichandran et al., “Deep Learning-Based Fake Banknote Detection System,” Journal of Emerging Technologies and Innovative Research (JETIR), vol. 12, no. 4, pp. 761–768, 2025.