Explainable AI Assisted Evolutionary Search of Engineering Designs

Rahul Dubey
Computer Science, Missouri State University
Email: RahulDubey@MissouriState.edu

2024 IEEE 36th International Conference on Tools with Artificial Intelligence (ICTAI) | 979-8-3315-2723-5/24/$31.00 USG | DOI: 10.1109/ICTAI62512.2024.00070

***Abstract*—Evolutionary algorithms are widely utilized to discover optimal and feasible solutions for NP-hard problems. However, the factors influencing a solution's performance, such as objective values, feasibility, or infeasibility, often lack explanation. This paper presents a novel Explainable AI (XAI)-assisted evolutionary search approach that utilizes evolutionary data to learn problem characteristics and guide the search process, thereby enhancing the quality of evolved solutions. The proposed method employs local and global explainable techniques efficiently and precisely to optimize parameters. The paper also introduces a framework that evolves engineering designs using the proposed XAI-assisted evolutionary approach. A practical application involving the optimization of a 2D car chassis demonstrates the effectiveness of the proposed algorithm. Experimental results show that the proposed XAI-assisted evolutionary algorithm improves the quality of solutions by between 8.15% to 39.95% over four iterations compared to traditional methods.**

***Index Terms*—XAI, GA, machine learning, design**

I. Introduction

Since the introduction of Explainable AI (XAI) by DARPA [1], numerous XAI techniques have been developed [2]–[5] and applied to domain-specific tasks involving deep learning models for predictions [6]. Recently, there has been a growing emphasis on integrating XAI with evolutionary search algorithms to clarify the behavior of evolved solutions. One such domain is engineering design, where explainability is essential to gain the trust and confidence of human designers. Beyond explainability, knowledge discovery plays a vital role in helping human designers improve the quality of their solutions. Previous research [7] suggests that knowledge discovered during evolutionary searches can aid in understanding the characteristics of problems.

This paper aims to enhance the explainability of solutions, facilitate knowledge discovery from evolutionary runs, and improve the performance of evolutionary search in engineering design. The paper introduces a novel XAI-assisted evolutionary search approach and a framework that an autonomous design agent uses to evolve higher quality solutions. Modern engineering designs face increasing complexity and critical demands that traditional methodologies struggle to address effectively. Even experienced human designers often find it challenging to navigate the intricate decision spaces required for optimal designs. The framework presented here leverages knowledge gained from evolutionary data to redefine the design process, offering a transformative approach that surpasses manual methods.

The proposed approach incorporates XAI to improve solution quality across multiple evolutionary runs. XAI is crucial for providing transparency and interpretability in understanding the relationship between solutions and their performance. With minimal information—just genomes and performance evaluations (fitness and constraints)—XAI techniques help clarify the underlying rationales. This integration enables the iterative enhancement of design solutions, using insights from previous genetic algorithms (GA) runs to refine future searches. The methodology section of this paper provides detailed insights into the proposed approach and framework.

An autonomous designer agent using the proposed framework can greatly enhance the efficiency of the design process by reducing the time and computational resources needed for optimal solutions, compared to human designers. Moreover, it ensures that designs meet essential safety and regulatory standards critical in structural engineering. Additionally, understanding critical decision variables improves solution quality and deepens problem understanding.

To evaluate the effectiveness of the proposed approach and framework, this study selects a 2D car chassis design problem for which an autonomous designer agent evolves solutions. Chassis design is a well-understood and extensively studied problem, allowing the knowledge discovered using XAI techniques to be compared with fundamental design concepts. Several simulation experiments were conducted, starting with the evolution of 2D car chassis designs using a canonical GA. Subsequently, local and global XAI techniques were applied to find which genes (decision variables) were more influential in fitness calculation, and the relationship between decision variables and performance. This process provided valuable insights into the problem, which were then used to adjust the decision space. Results indicate performance improvements of 8.15%, 24.48%, and 39.95% across three iterations. These findings demonstrate the effectiveness of the proposed XAI-assisted evolutionary algorithms (EA) in solving real-world problems.

The main contributions of this paper are as follows: 1) to the best of the author's knowledge, this is the first paper to utilize data generated during multiple evolutionary runs for understanding problem characteristics and knowledge discovery, which are subsequently used to enhance the quality of evolved solutions, and 2) the proposed framework can be modified and extended to other problems where evolutionary algorithms are used to find solutions, thus increasing its generalizability.

DOI 10.1109/ICTAI62512.2024.00070

II. Related Work

Traditional engineering design methodologies, which heavily rely on human expertise and iterative testing, often face challenges in navigating complex decision spaces and efficiently meeting stringent performance criteria. The emergence of XAI offers a promising avenue to enhance these methodologies by introducing transparency and interpretability into automated decision-making processes. XAI allows engineers to understand the reasoning behind AI-driven decisions, which is crucial for validating design choices and optimizing solutions effectively [8]. Key challenges in autonomous engineering design include scalability, the interpretability of AI-driven decisions, and the integration of domain-specific knowledge into AI models.

For many years, researchers have used evolutionary algorithms to address complex real-world problems, resulting in the development of various effective EA techniques [9]. To further enhance performance, the integration of different machine learning techniques with EAs has been investigated across various domain-specific problems [10], [11]. For instance, Qian [12] proposed a deep learning guided evolutionary algorithm (DLGEA) designed to improve efficiency by optimizing the search space. DLGEA trains and uses a deep neural network to identify the relationship between decision variables and outcomes. Subsequently, for a given candidate solution, DLGEA guides the initialization and refines the search space of EAs. Majdi introduced a rule-based reinforcement learning (RL) methodology to improve the performance of evolutionary algorithms in constrained environments [13]. Initially, RL proximal policy optimization agents are trained to learn and adhere to specific problem rules and constraints. These trained agents then leverage their experiences to guide various evolutionary and stochastic algorithms, such as genetic algorithms, simulated annealing, particle swarm optimization, differential evolution, and natural evolution strategies. A major drawback of these approaches is their reliance on external datasets for training models, which are often unavailable.

Alternatively, data generated during an evolutionary search can be used to train models and subsequently improve performance. Erik introduced an evolutionary optimization algorithm that leverages knowledge extracted during its operation to guide its search strategy [14]. This approach uses a self-organizing map to extract knowledge and identify promising areas by narrowing the search space. However, despite improvements in performance, the knowledge extraction method used here provides limited insight into problem characteristics, which is essential for explaining why certain solutions perform better than others. Machine learning-guided evolutionary algorithms aim to enhance performance through predictive models but often do not prioritize interpretability.

In contrast, XAI-guided evolutionary algorithms has potential to improve both performance and transparency by employing techniques that make decisions more understandable. These methods ensure that stakeholders can trust and comprehend the algorithm's decisions, balancing efficiency with interpretability. Integrating evolutionary algorithms with XAI can become a powerful approach for autonomously searching and refining solutions [15].

The integration of XAI and EA is still in its early stages, with some initial work emerging in recent years. Yi [16] conducted a survey on the use of genetic programming as an interpretable and explainable tool. Wang [17] utilized XAI techniques to clarify genetic algorithms in job scheduling problems. Fan [18] introduced ExpGA, an explanation-guided method that combines a model-agnostic fairness testing approach with genetic algorithms for software fairness testing. The results indicated that ExpGA was highly efficient and effective in detecting discriminatory samples. However, it depends on external datasets for model training and initial solution generation. In the literature, Local Interpretable Model-agnostic Explanations (LIME) [2] and SHapley Additive exPlanations (SHAP) [3] are the two most popular techniques for local and global explanation techniques. Using these, Wang [19] introduced a multi-objective genetic algorithm to evolve local interpretable model-agnostic explanations in image classification problems. In the field of engineering design, there is limited work, but some initial efforts at the system engineering level have been proposed in [20].

III. Methodology

This section first outlines the proposed framework, including design encoding and performance evaluation, followed by a detailed explanation of the proposed XAI-assisted evolutionary algorithm.

A. Proposed Framework

The proposed framework as illustrated in Algorithm 1 considers the upper and lower boundaries (UB, LB) to define the problem, executes the proposed XAI-assisted EA for a number of iterations, and returns the best solution found. The problem definition encompasses design definition, genome/design encoding, performance evaluation, data storage generated during the evolutionary run, and the search space update mechanism.

A 2D car chassis design is created using the Delaunay triangulation [21] of a set of points, which includes support points, load/impact points, and initialization points, as illustrated in Figure 1. The location of these points influences the topology of the structure, affecting performance in terms of volume of chassis and node displacements. Support and load points are user-defined, while initialization points can be generated anywhere within the design space boundary. An evolutionary algorithm is employed to find the optimal or near-optimal placement of these initialization points to produce 2D chassis designs. Optimal placement of these initialization points can result in an optimal structural topology, so the points are encoded into real-valued chromosomes. A structure can be represented by a graph with N nodes and E edges. Besides the structural topology, the size of the structure significantly affects performance, hence the thickness of edges is also encoded into the chromosome.

Algorithm 1: Proposed Framework

Input : UB, LB
Output: P_{best}
Problem = defineProblem(Pop, Gen, UB, LB)
$P_{best} = inf$
for *i in Iters* **do**
  $C_{best}, \delta_n, \delta_t = XAI - Assiated\ EA(Problem)$
  $data_n, data_t$ = $getData()$
  $C_n, C_t = Correlation(data_n, data_t)$
  If $C_{best} \leq P_{best}$ *then* $P_{best} = C_{best}$
  $Problem.updateBounds(C_n, C_t)$
end
return P_{best}

A chromosome consists of two types of genes: node genes (n) and thickness genes (t). For this problem, 20 node genes and 14 thickness genes are needed to optimize a design, making the total length of a genome 34.

Once a design is initialized, the fitness and constraints are calculated using Eq. 1. The goal is to evolve chassis designs with lower weight that can withstand various crash scenarios. Different types of impacts result in different force distributions on the structure, and the evolved design must endure all these scenarios. To ensure driver safety, the displacement caused by impact forces must remain within specified safe limits. If a design fails to meet this criterion, it is considered infeasible.

$$\min \quad V = \sum_{i=1}^{e} A_i L_i$$
$$\text{s.t.} \quad \sum_{i=0}^{i=s} g_i = \left\{ \begin{array}{ll} 0, & \text{if } d_{min} \leq d_i \leq d_{max} \\ |d_{max} - d_i|, & \text{if } d_i \geq d_{max} \\ |d_i - d_{min}|, & \text{if } d_{min} \leq d_i \end{array} \right\} \tag{1}$$

In equation 1, V is the volume, A_i and L_i are the cross-sectional area and length of the i^{th} edge, and e is the total number of edges in a given design, s represents different crash scenarios (different impact locations), d is the displacement at the driver's location, and g_i is the constraint in the i^{th} crash scenario. Finite Element Analysis (FEA) simulation is conducted to compute the displacement under different impact/loading scenarios. Once the problem is defined, the framework executes the proposed XAI-assisted EA algorithm, which returns the current best-evolved solution (C_{best}) along with binary information indicating which genes (node, thickness) in the decision space need adjustment. $\delta_n = 1$ refers that node genes affect the fitness, not constraint. Likewise $\delta_t = 1$ refers that thickness genes affect the fitness, but not the constraint. Depending on δ_n and δ_t values, search spaces can be modified to guide the evolutionary search towards more promising regions. To update the search space of either node or thickness genes, first correlation between genes and output (fitness) is computed. A positive correlation means that as gene parameter increase, the fitness increases, thus the upper bound of search space is lowered.

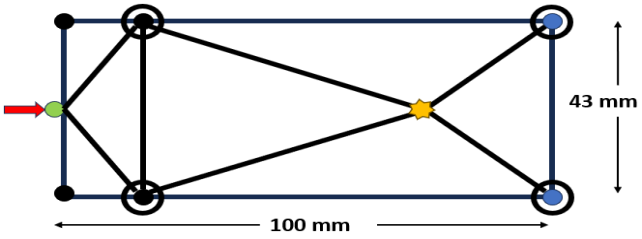


Fig. 1. The figure illustrates an example design where the driver is indicated by a yellow star. Support points are marked by blue dots, the impact point is shown in green, and initialization points are represented by black dots.

If C_{best} is better than the all-time best solution (P_{best}), then P_{best} is updated, and the bounds of the search space are adjusted based on the values of node gene correlation (C_n) and thickness gene correlation (C_t). In one iteration, the bounds are changed by 20%. Other values were tested, but this percentage worked best for this problem. However, this value may not be optimal for other problems and warrants further investigation. After the search space adjustment, the XAI-assisted EA runs the evolutionary search again, repeating this process for $Iters$ iterations, a user defined number. The proposed framework can be adapted for various real-world problems, enhancing its generalizability. By systematically applying XAI techniques to improve decision-making and solution refinement, the framework aims to set a new standard in autonomous engineering design.

B. Proposed XAI-Assisted Evolutionary Algorithm

Algorithm 2 outlines the proposed XAI-assisted EA, which takes a defined problem and returns the best-evolved solution along with the importance of decision variables (node and thickness) for predicting fitness and constraints. The algorithm initializes a population, evaluates it, and stores records for further analysis in each generation. Here P_{best} represents the best evolved solution by the end of the evolutionary run. The stored data is divided into training and testing sets (D_{tr}, D_{te}) to train different models which late help in computing each decision variable's importance towards fitness and constraint.

In this study, local (LIME) and global (SHAP) explanation techniques are employed to determine the contribution of decision variables to the final outcome. Both LIME and SHAP are known to be computationally intensive approaches [3], as they generate a large number of perturbed samples and then calculate the outcome (fitness and constraints) to assess the importance of each decision variable. Since evaluating the performance of a genome can be a time-consuming process, various regression and classification models were trained. These models enable faster and more cost-effective prediction of fitness and constraints for a given genome.

The fitness of a solution is represented by a real value, while constraint violations are indicated by binary outcomes. Therefore, separate regression models (R_{models}) and classification models (C_{models}) are trained using the training dataset

Algorithm 2: XAI-Assisted EA

Input : $Problem$
Output: $\mathrm{P}_{best}, \delta_n, \delta_t$

$Pop = Problem.Initialization()$
$runGA(pop)$
$\mathrm{P}_{best} = Problem.bestSolution()$
$\mathrm{D}_{tr}, D_{te} = Problem.splitData()$
$\mathrm{R}_{models}.fit(D_{tr})$
$\mathrm{C}_{models}.fit(D_{tr})$
$\mathrm{best}_r, best_c = R_{models}.best(), C_{models}.best()$
***Fitness-based Decision Variables Importance* **
$\mathrm{n}_{lr}, t_{lr} = LIMEExp(best_r, D_{te})$
$\mathrm{n}_{gr}, t_{gr} = SHAPExp(best_r, D_{te})$
$\delta_n, \delta_t = compareRanks(n_{lr}, t_{lr}, n_{gr}, t_{gr})$
Constraint-based Decision Variables Importance
$\mathrm{n}_{lr}, t_{lr} = LIMEExp(best_c, D_{te})$
$\mathrm{n}_{gr}, t_{gr} = SHAPExp(best_c, D_{te})$
$\delta_n, \delta_t = compareRanks(n_{lr}, t_{lr}, n_{gr}, t_{gr})$
return $\mathrm{P}_{best}, \delta_n, \delta_t$

D_{tr} to predict fitness (real values) and constraint violations (binary classification), respectively. The performance of the regression models is evaluated using the R^2 metric, while classification models are assessed based on accuracy. Based on these metrics, the best-performing regression ($best_r$) and classification ($best_c$) models are selected for further analysis. Once these optimal models are identified, they are used to compute the importance scores of the decision variables (of test dataset D_{te}), which are then ranked accordingly. Feature ranking is first performed using the regression model, followed by the classification model. Ranking of decision variables is computed using equation 2.

$$\bar{r}_j = \frac{R_j}{N} = \frac{1}{N}\sum_{i=1}^{N} r_{ij} \tag{2}$$

In this equation, $\bar{r}j$ represents the average rank of the j^{th} decision variable (feature), N is the total number of test samples, R_j is the total sum of ranks for the j^{th} feature across all test samples, and rij is the rank of the j^{th} feature for the i^{th} test sample. Initially, LIME is used to compute the rankings (node local rank (n_{lr}) and thickness local rank (t_{lr})), followed by SHAP, which calculates the importance of each decision variable and ranks them (node global rank (n_{gr}) and thickness global rank (t_{gr})). This approach offers valuable insight for guiding the search towards more optimal areas of the search space by emphasizing the importance of node and thickness genes.

It is important to note that decision variables should be adjusted in a way that reduces fitness without violating constraints. Therefore, if node or thickness genes are found to be crucial for maintaining constraints, it should not be altered. By comparing these rankings, decisions can be made regarding which type of gene should be modified and which ones should remain fixed. If $\delta_n = 1$, it indicates that the node genes have a greater influence on fitness compared to the thickness genes and less influence on constraint. Conversely, if $\delta_t = 1$, it signifies that the thickness genes are more influential than the node genes. Since the goal in the next iteration of evolutionary search is to reduce fitness, this information is crucial for determining how to adjust the search space to steer evolution towards a more optimal region. Finally, algorithm 2 returns P_{best}, δ_n, and δ_t, allowing algorithm 1 to make informed decisions for subsequent steps.

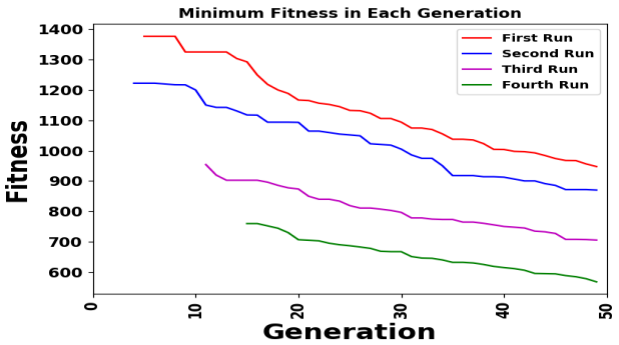
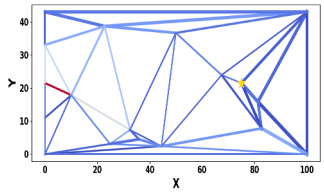


Fig. 2. The figures shows minimum fitness obtained in each generation during four different evolutionary runs. Except the first GA run, the evolutionary search is guided by knowledge learned using different local and global XAI techniques. The best solutions' fitness during four runs are 947.70, 870.43, 706.22, and 569.06.

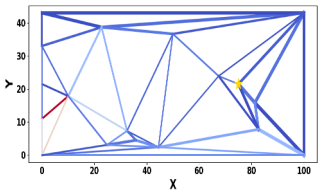
IV. Results and Discussion

This paper presents an XAI-assisted EA and a framework that an intelligent autonomous agent (designer) uses to evolve high-quality, feasible 2D car chassis designs. Simulation experiments were conducted to evaluate the effectiveness of the proposed approach. The experimental setup involved running a canonical GA with a population size of 50 for 50 generations, employing simulated binary crossover (SBX) and polynomial mutation operators. Other values for population size and number of generations were considered, but this set was chosen because it offers a good balance between exploration and exploitation of the search space.

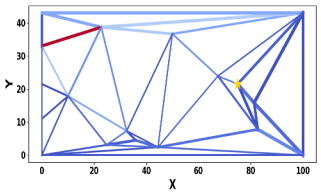
Each potential solution, represented by a real-valued chromosome, encodes 20 node genes and 14 thickness genes. These 20 node genes encode 10 point coordinates (X and Y). To initialize a design, a crash impact/ loading point, a driver's location, four fixed points, and ten initialization points are considered, where the GA evolves the locations of the initialization points. Node locations control the topology of the structure/design and affect performance. Similarly, edge thickness controls the size of the structure and thus performance. Once a chromosome (genotype) is decoded and a structure/design is created (phenotype), the volume is calculated. The displacement at the driver's location is evaluated under three distinct loading/impact scenarios to assess structural resilience, and thus, FEA analysis is conducted three times for each design.



(a) Impact point at $(0, 21.5)$



(b) Impact point at $(0, 11)$



(c) Impact point at $(0, 33)$

Fig. 3. Figure depicts the best-evolved design ($Best_1$) under the three impact scenarios with a volume of 947.70, in the first run. The thickness of edges is represented by the line thickness: thicker lines indicate greater thickness, and vice versa. Different colors of edges indicate the distribution of forces, where dark red signifies higher force and dark blue signifies lower force.

TABLE I
PERFORMANCE OF REGRESSION AND CLASSIFICATION MODELS

Model	Regression (R^2)			Classification (Accuracy)		
	First	Second	Third	First	Second	Third
KNN	0.9407	0.9380	0.9590	0.822	0.817	0.8413
MLP	0.9579	0.9509	0.9733	0.852	0.828	0.8473
SVM	0.9570	0.9460	0.9497	0.842	0.823	0.8413
RF(5)	0.9449	0.9281	0.9528	0.866	0.829	0.8386
RF(15)	0.9592	0.9621	0.9781	0.892	0.851	0.8653
RF(25)	**0.9593**	**0.9622**	**0.9782**	**0.90**	**0.872**	**0.8666**

A. XAI Assisted Evolutionary Search: First Run

In the initial run, solutions or designs are randomly initialized within the original decision or search space. Figure 2 shows the minimum volume obtained in each generation during the first GA run, indicated by the red line. This plot starts from the 6^{th} generation because feasible solutions were only found starting from that point. A solution is considered feasible only when the displacement at the driver's location falls within the given safe limits in all three crash scenarios. Figure 3(a), (b), and (c) show the best evolved ($Best_1$) feasible design under three impact scenarios at the end of the first GA run, with a volume of 947.70, where the driver is located at $(75, 21.5)$, denoted by the yellow star.

The evolved topology ensures that the force is distributed across other edges, reducing the impact at the driver's location and keeping the driver's displacement within safe limits. In these figures, red-colored edges indicate high force, while blue-colored edges indicate low force. After completing the first GA run, a dataset containing 2500 records (50 population $\times$ 50 generations) was compiled. This dataset includes the performance metrics of evaluated solutions across the three loading scenarios and serves as the basis for training different machine learning models. The recorded data includes all chromosomes and their corresponding fitness (real value output) and constraint violation (binary output). This means the models should predict a given solution's fitness (continuous value) and constraint violation (binary classification).

B. Model Training

Various ML models, including K-nearest neighbors (KNN), support vector machines (SVM), neural networks (NN), and random forest (RF), were trained as regressors to predict fitness and as classifiers to predict constraint violations (CV). For the RF model, three different depths were tested: 5, 15, and 25. From the total recorded samples, 80% were used for training, and the remaining 20% were used to test the performance of the trained models. Table I shows and compares the performance of these regressors and classifiers at the end of each evolutionary run. After the first run, the RF(25) regressor's R^2 score and the RF(25) classifier's accuracy score were the highest, indicating that the RF(25) regressor and classifier outperformed the other ML models. Higher R^2 and accuracy scores indicate a better fit for the regressor and classifier models, thus RF(25) was chosen for further analysis.

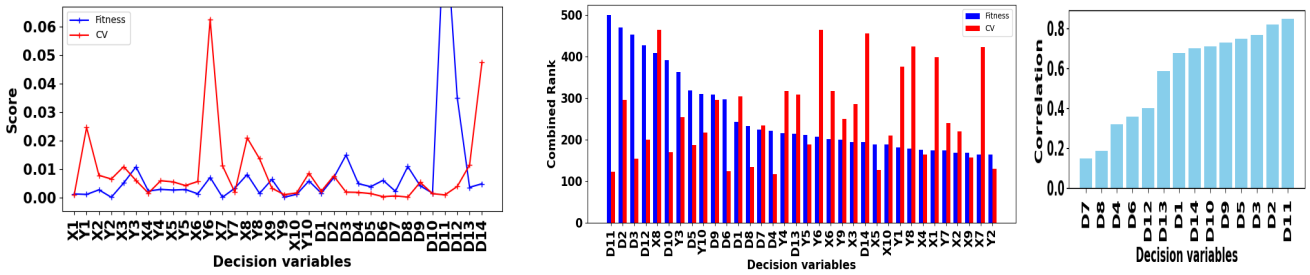
C. Local and Global Explanations

To enhance the performance of the evolutionary search algorithm, it is crucial to understand the interactions between decision variables and fitness and constraint. In this work, LIME and SHAP are employed for decision variable importance analysis. The best-evolved solution from the end of the first run, along with the RF(25) regressor and classifier models, were selected for this analysis. Figure 4(a) illustrates the importance scores for each decision variable in predicting fitness (blue) and CV (red).

A close examination of Figure 4(a) reveals some intriguing insights. For the fitness performance metric, indicated by the blue line, most thickness genes (represented by D's) have higher importance scores than node genes (represented by X, Y), suggesting that thickness genes are more influential in fitness prediction. In contrast, node genes play a significant role in making a design feasible, as evidenced by the red line showing higher importance scores for most of the X's and Y's compared to the D's. The node locations, which control the topology of the design, are crucial for diverting force distribution away from the driver's location, thereby ensuring the design's feasibility.

These findings and insights gained from explainable AI methods align perfectly with real-world structural engineering design principles. An autonomous designer agent does not initially have access to this information, but by using the data from the evolutionary run and the proposed XAI-assisted EA, it learns important characteristics of the problem that can later be used to enhance performance. For example, to reduce the volume of the structure, expert designers carefully reduce the thickness of edges that experience less stress and force.

The findings in Figure 4(a) correspond to the best-evolved solution and provide a local interpretation. To determine whether these characteristics hold true for other solutions on



(a) Decision variables importance of the best solution (b) Decision variables ranking over 500 samples. (c) Correlation

Fig. 4. Figure (a) presents the scores of node and thickness genes in predicting fitness (blue) and constraint violation (red), as calculated using LIME. This analysis indicates that thickness genes are crucial for volume, while node genes play an essential role in ensuring feasibility. Figure (b) shows the ranking of decision variables across 500 test samples, further supporting the influence of node and thickness genes on fitness and CV. Figure (c) shows how thickness genes are correlated with fitness.

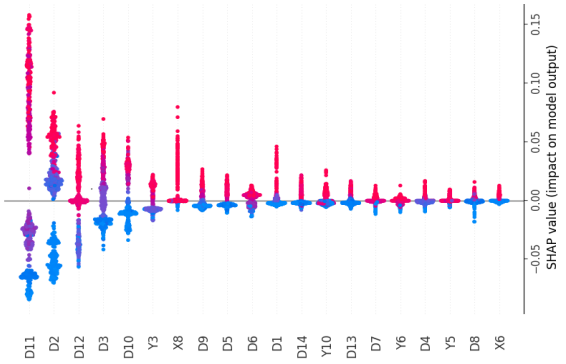


Fig. 5. Global ranking of top node and thickness genes.

a global scale, the entire test dataset was used to calculate decision variable scores with LIME. These scores were then ranked according to Equation 2. Figure 4(b) shows the ranking of decision variables over 500 samples, again indicating that thickness genes are more influential in fitness prediction, while node genes are more important for CV prediction. To further validate these results, SHAP, a global explanation technique, was employed, and the results are presented in Figure 5. Both local and global explanations suggest that to optimize fitness, thickness genes should be adjusted. Whereas node genes primarily govern the topology and maintain feasibility, and thus should remain fixed.

D. Correlation Between Decision Variables and Fitness

To steer the evolutionary search towards more promising regions, the search space boundaries need to be adjusted. Based on previous observations, node genes should remain fixed to maintain feasibility, while thickness genes can be varied. Therefore, the correlation of each thickness gene, represented by D's, with fitness is computed and shown in Figure 4(c). For a given gene, a positive correlation indicates that as the gene's value increases, fitness also increases, and vice versa. Since all thickness genes show positive correlation, reducing the upper bounds of the thickness genes will help lower the fitness.

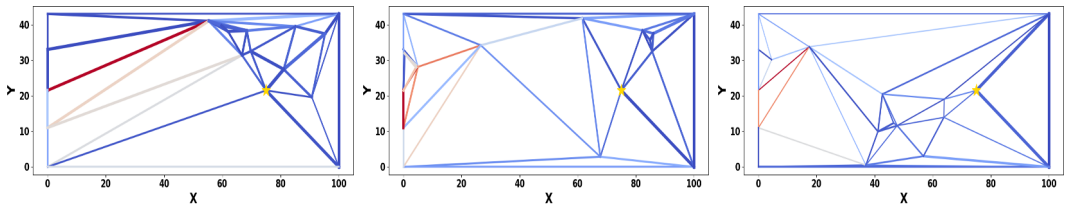
E. XAI Assisted Evolutionary Search: Next Three Runs

Leveraging insights gained from the proposed XAI-assisted EA, the framework directs the evolutionary search by adjusting the boundary of the thickness gene space. A GA with a population size of 50 was executed for 50 generations a second time, and the minimum fitness values in each generation are shown in Figure 2 (blue line). This time, the fitness of the best-evolved solution ($Best_2$) is 870.43, which is 8.15% lower than the fitness of the best solution from the first run ($Best_1$). This indicates that the proposed framework and XAI-assisted EA have enhanced performance.

Figure 6(a) displays the best-evolved design at the end of the second GA run under the first loading scenario. By the end of this run, an additional dataset of 2500 records was generated, increasing the total dataset size to 5000. The same analysis as before was conducted. Table I indicates that RF(25) achieved the best performance in the second run compared to other models, leading to its selection for further analysis.

Based on these results, the decision was made once more to adjust the space for thickness genes while keeping the node genes space fixed. The correlation between thickness genes and fitness was calculated, and since the correlations were positive, the upper boundaries of the thickness genes were further reduced. Due to space constraints, LIME, SHAP, and correlation analyses from the second run are not presented here.

Following the results of the first two XAI-assisted evolutionary runs, two additional iterations of the evolutionary search were conducted. A consistent decrease in fitness values was observed, indicating the effectiveness of the proposed XAI-assisted EA and framework in optimizing solutions over time. Notably, the decision space is iteratively reduced using explainable AI techniques in each subsequent run. This reduction in decision space leads to an increased number of generations required to identify feasible solutions, which is why the feasible solution plots for subsequent runs begin at



(a) $Best_2$, 8.15% lower than $Best_1$ (b) $Best_3$, 24.48% lower than$Best_1$ (c) $Best_4$, 39.95% lower than $Best_1$

Fig. 6. Figure (a) displays the most optimal feasible design achieved in the second run, with an objective value of 847.08. This represents an 8.15% reduction in volume compared to the best-evolved solution ($Best_1$) from the first run, highlighting the improvement in solution quality enabled by the XAI-guided evolutionary search. Similarly, (b) and (c) shows the best evolved designs at the end of the third and fourth evolutionary run.

different generations (on the x-axis), as illustrated in Figure 2. These results highlight the robust optimization capabilities of the GA, further enhanced by the XAI-driven reduction of the decision space. Figure 6(b) and (c) present the best-evolved design at the end of the third ($Best_3$) and fourth ($Best_4$) GA runs under the first loading scenario, with a volume of 706.22 and 569.06 respectively. ($Best_3$) is 25.48% lower than ($Best_1$) whereas ($Best_4$) is 39.95% lower than ($Best_1$).

V. Conclusion and Future Work

This paper introduces a novel XAI-assisted evolutionary algorithm and a framework that enables an intelligent autonomous designer agent to develop higher-quality engineering designs compared to conventional evolutionary algorithms. The study specifically applied this approach to the design of a 2D car chassis, where the autonomous designer autonomously encoded solutions, performed finite element analysis, and utilized the proposed algorithms to iteratively enhance the designs. Simulation results show that in four iterations, the performance improved between 8.15% to 39.95% compared to standard GA performance. The findings underscore the significance of targeted design modifications and highlight the value of interpretable machine learning models in structural design. Looking ahead, the focus will be on optimizing decisions related to individual node and thickness genes to further enhance performance. Future research should aim to refine XAI techniques, boost computational efficiency, and improve the robustness of machine learning models used in the autonomous design process.

References

[1] D. Gunning, E. Vorm, Y. Wang, and M. Turek, "Darpa's explainable ai (xai) program: A retrospective," *Authorea Preprints*, 2021.

[2] M. T. Ribeiro, S. Singh, and C. Guestrin, "" why should i trust you?" explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1135–1144.

[3] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Advances in neural information processing systems*, vol. 30, 2017.

[4] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 618–626.

[5] G. Montavon, A. Binder, S. Lapuschkin, W. Samek, and K.-R. Müller, "Layer-wise relevance propagation: an overview," *Explainable AI: interpreting, explaining and visualizing deep learning*, pp. 193–209, 2019.

[6] A. Das and P. Rad, "Opportunities and challenges in explainable artificial intelligence (xai): A survey," *arXiv preprint arXiv:2006.11371*, 2020.

[7] C.-H. Chen, L. P. Khoo, Y. T. Chong, and X. F. Yin, "Knowledge discovery using genetic algorithm for maritime situational awareness," *Expert Systems with Applications*, vol. 41, no. 6, pp. 2742–2753, 2014.

[8] T. Miller, P. Howe, and L. Sonenberg, "Explainable ai: Beware of inmates running the asylum or: How i learnt to stop worrying and love the social and behavioural sciences," *arXiv preprint arXiv:1712.00547*, 2017.

[9] A. Slowik and H. Kwasnicka, "Evolutionary algorithms and their applications to engineering problems," *Neural Computing and Applications*, vol. 32, pp. 12 363–12 379, 2020.

[10] B. Li, Z. Wei, J. Wu, S. Yu, T. Zhang, C. Zhu, D. Zheng, W. Guo, C. Zhao, and J. Zhang, "Machine learning-enabled globally guaranteed evolutionary computation," *Nature Machine Intelligence*, vol. 5, no. 4, pp. 457–467, 2023.

[11] H. Al-Sahaf, Y. Bi, Q. Chen, A. Lensen, Y. Mei, Y. Sun, B. Tran, B. Xue, and M. Zhang, "A survey on evolutionary machine learning," *Journal of the Royal Society of New Zealand*, vol. 49, no. 2, pp. 205–228, 2019.

[12] K. Qian, J. Jiang, Y. Ding, and S.-H. Yang, "Dlgea: a deep learning guided evolutionary algorithm for water contamination source identification," *Neural Computing and Applications*, vol. 33, no. 18, pp. 11 889–11 903, 2021.

[13] M. I. Radaideh and K. Shirvan, "Rule-based reinforcement learning methodology to inform evolutionary algorithms for constrained optimization of engineering applications," *Knowledge-Based Systems*, vol. 217, p. 106836, 2021.

[14] E. Cuevas and J. Galvez, "An optimization algorithm guided by a machine learning approach," *International Journal of Machine Learning and Cybernetics*, vol. 10, no. 11, pp. 2963–2991, 2019.

[15] J. Bacardit, A. E. Brownlee, S. Cagnoni, G. Iacca, J. McCall, and D. Walker, "The intersection of evolutionary computation and explainable ai," in *Proceedings of the Genetic and Evolutionary Computation conference companion*, 2022, pp. 1757–1762.

[16] Y. Mei, Q. Chen, A. Lensen, B. Xue, and M. Zhang, "Explainable artificial intelligence by genetic programming: A survey," *IEEE Transactions on Evolutionary Computation*, vol. 27, no. 3, pp. 621–641, 2022.

[17] Y.-C. Wang and T. Chen, "Adapted techniques of explainable artificial intelligence for explaining genetic algorithms on the example of job scheduling," *Expert Systems with Applications*, p. 121369, 2023.

[18] M. Fan, W. Wei, W. Jin, Z. Yang, and T. Liu, "Explanation-guided fairness testing through genetic algorithm," in *Proceedings of the 44th International Conference on Software Engineering*, 2022, pp. 871–882.

[19] B. Wang, W. Pei, B. Xue, and M. Zhang, "A multi-objective genetic algorithm to evolving local interpretable model-agnostic explanations for deep neural networks in image classification," *IEEE Transactions on Evolutionary Computation*, 2022.

[20] P. Geyer, M. M. Singh, and X. Chen, "Explainable ai for engineering design: A unified approach of systems engineering and component-based deep learning," *arXiv preprint arXiv:2108.13836*, 2021.

[21] D.-T. Lee and B. J. Schachter, "Two algorithms for constructing a delaunay triangulation," *International Journal of Computer & Information Sciences*, vol. 9, no. 3, pp. 219–242, 1980.