Exploring Randomized Optimization: Enhancing Supervised Learning in Neural Networks

Mar Tejedor Ninou
CS7641 Spring 2024

I. Introduction

This paper delves into the exploration of randomized optimization algorithms, specifically focusing on Randomized Hill Climbing, Simulated Annealing, Genetic Algorithms, and Mutual-Information-Maximizing Input Clustering (MIMIC). The objective is to implement and analyze these algorithms across discrete optimization problems and then apply the same algorithms to the challenge of optimizing weights for a neural network. This assignment transcends traditional optimization boundaries, venturing into supervised learning by optimizing neural network weights using randomized search algorithms. Through comprehensive analysis, we aim to show algorithm behavior, understand observed results, and identify paths for performance enhancement.

II. Optimization Problems

In this research paper, we present three optimization problems designed for discrete-valued parameter spaces, specifically using bit strings. The chosen problems—Continuous Peaks, Flip Flop, and Four Peaks—are carefully chosen to demonstrate the strengths of Genetic Algorithms, Simulated Annealing, and MIMIC.

A. Continuous Peaks

In the Continuous Peaks problem, the goal is to optimize a binary string by maximizing the number of contiguous values that are either all 0s or all 1s. This problem presents a challenging optimization scenario characterized by a deceptive landscape. It involves navigating a prolonged flat region leading to a high peak, making it essential for optimization algorithms to strike a balance between exploration and exploitation. The deceptive nature of the landscape and the need for an effective exploration-exploitation trade-off makes Continuous Peaks an interesting problem suitable for randomized optimization algorithms like Genetic Algorithm.

B. Flip Flop

The Flip Flop problem is a binary optimization challenge where the goal is to maximize transitions (0 to 1 or 1 to 0) in a binary string. With numerous local optima and a discrete landscape, it tests algorithms' abilities to navigate and escape suboptimal solutions. The problem's characteristics make it an ideal candidate for assessing randomized optimization algorithms like Simulated Annealing, as they must strategically explore the binary space to find the global optimum.

C. Four Peaks

The Four Peaks problem goal is to find the binary string configuration that maximizes a fitness function. The fitness landscape consists of a global maximum and multiple deceptive local optima. This landscape challenges optimization algorithms to efficiently balance exploration and exploitation. The problem's combination of local and global features makes it a compelling scenario for evaluating the effectiveness of algorithms like Genetic Algorithms and Mutual-Information-Maximizing Input Clustering (MIMIC). The problem's suitability for MIMIC arises from its probabilistic modeling approach, enabling efficient navigation through the complex landscape. Evaluating algorithms for early convergence becomes crucial in understanding their efficiency in capturing the global optimum without succumbing to suboptimal solutions.

III. Hyperparameters

In the process of evaluating randomized optimization algorithms, a meticulous approach to hyperparameter tuning was adopted. The initial focus across all algorithms was on the problem size, where the optimal configuration was identified. Subsequently, specific hyperparameters were fine-tuned for each algorithm to enhance their effectiveness. SA underwent assessment with different decay types, selecting the one demonstrating superior convergence characteristics. RHC's exploration capability was influenced by tuning the number of restarts. GA's performance was optimized by adjusting the mutation probability and population size. MIMIC was fine-tuned based on the keep percentage and population size. This systematic hyperparameter tuning ensured that each algorithm was configured optimally, contributing to a comprehensive evaluation of their performance on the given problems.

IV. Continuous Peaks Problem

The Continuous Peaks problem is characterized by a deceptive landscape featuring an elongated flat region leading to a high peak. Examining Figure 1 shows distinct algorithmic behaviors. GA consistently outperforms others in fitness across various problem sizes, showcasing its adaptability. SA and MIMIC demonstrate competitive fitness, while RHC faces challenges due to its randomness component. Figure 2 illustrates the trade-offs in algorithmic efficiency. Although GA achieves superior fitness, it requires significantly more time when the problem size is bigger. SA and RHC exhibit balanced efficiency, and MIMIC, despite premature convergence, displays competitive time performance. As shown in the graphs a better performance for all algorithms is achieved when problem size of 60 is chosen, so for the following graphs and for the individual analysis of each algorithm a problem size of 60 will be used. In Figure 3, GA's swift convergence is evident, reaching around 150 fitness in 1000 iterations, outperforming other algorithms in both iterations and fitness improvement. MIMIC struggles with premature convergence, while SA performs exceptionally well, achieving fitness levels comparable to GA but requiring more iterations.

A. Genetic Algorithms Analysis

The Continuous Peaks problem accentuates GA's strengths by demanding a strong exploration-exploitation trade-off. The flat region necessitates the exploration of diverse solutions, and GA's ability to maintain a diverse population facilitates efficient traversal. As shown in Figure 3 GA emerges as a strong contender for the Continuous

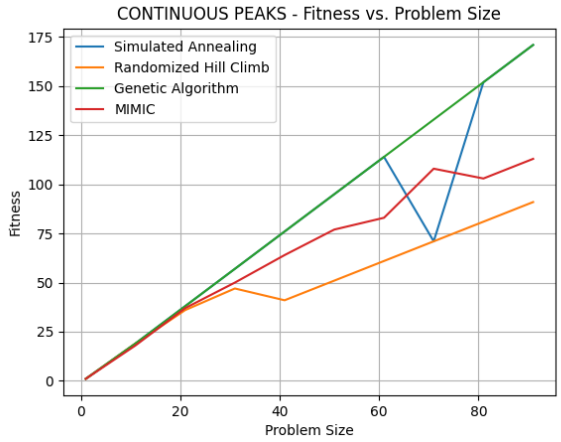

Fig. 1. Continuous Peaks - Fitness vs. Problem Size comparison.

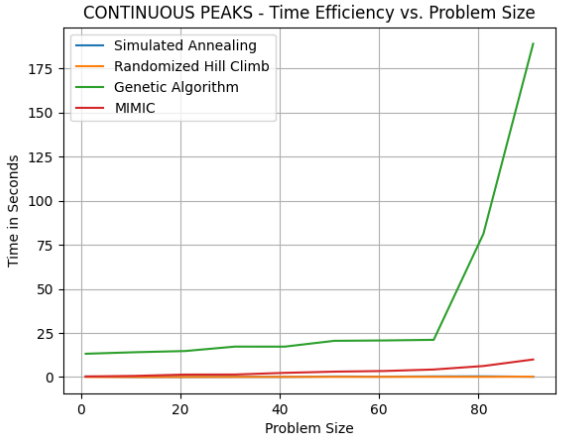

Fig. 2. Continuous Peaks - Time Efficiency vs. Problem Size comparison.

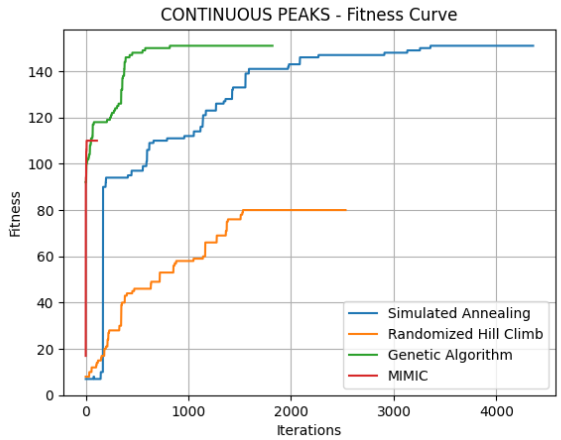

Fig. 3. Continuous Peaks - Fitness vs. Iterations comparison.

Peaks problem, consistently achieving the best fitness among all algorithms. However, the trade-off lies in its time-intensive nature, taking ten times more than other algorithms, Figure 2. Despite this, GA showcases rapid convergence, reaching more than 150 fitness in 1000 iterations with mutation probability 0.1 - 0.3 and population 500. The rationale behind the optimal parameters involves the delicate balance between mutation probability and population size. A mutation probability range of 0.1 to 0.3 allows for sufficient exploration, preventing premature convergence, while a population size of 500 ensures diversity for effective exploration. This balance enables GA to outperform other algorithms in fitness, showcasing its adaptability to the Continuous Peaks problem.

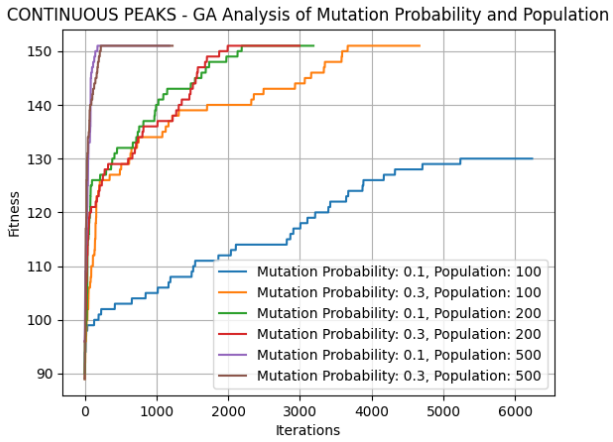

Fig. 4. Continuous Peaks - Genetic Algorithm analysis with mutation probability and population.

B. Simulated Annealing Analysis

The Continuous Peaks problem is particularly interesting for SA due to its distinctive characteristics. SA's probabilistic acceptance of worse solutions aligns well with the need for exploration in the long flat region. This property allows SA to venture into different regions of the solution space, potentially avoiding local optima that might hinder convergence. SA exhibits commendable fitness results, across all three decay types (exponential, linear, and geometric), Figure 5, comparable to GA but with the need of many more iterations. Notably, geometric decay converges quicker with fewer iterations, although achieving the same fitness level as other decay types. The rationale behind the preference for geometric decay lies in reducing temperature ensuring a smooth and controlled transition, making it the preferred choice for SA in this context.

C. Randomized Hill Climbing Analysis

Continuous Peaks poses an interesting challenge for RHC due to its focus on local exploration. The algorithm's quick convergence to local optima can be disadvantageous traversing the flat region efficiently as is shown in Figure 3. The impact of restart strategies becomes particularly interesting, as they influence RHC's ability to escape local optima and affect the convergence speed. Interestingly, RHC with four restarts achieves the quickest convergence with fewer iterations, Figure 6, closely rivaling the performance of 16 restarts. The rationale behind this lies in the algorithm's ability to escape local optima and suboptimal regions more efficiently, leading to quicker convergence. The consistency in fitness values across

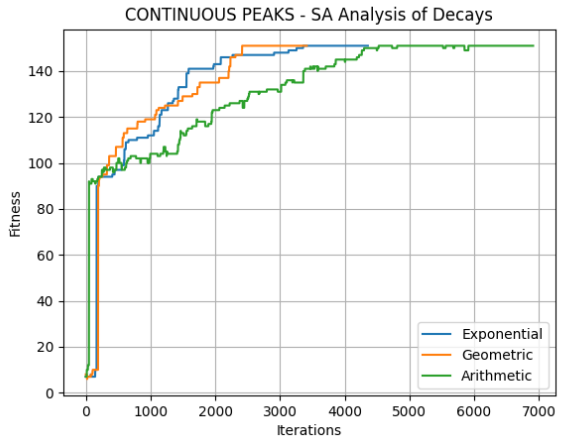

Fig. 5. Continuous Peaks - Simulated Annealing analysis with decays.

different restarts underscores RHC's resilience and adaptability to the deceptive landscape of the Continuous Peaks problem.

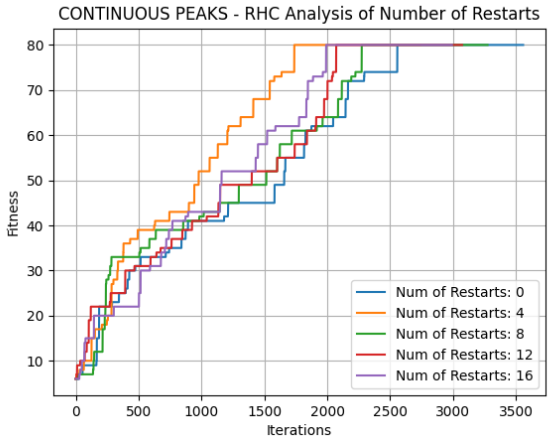

Fig. 6. Continuous Peaks - Randomized Hill Climbing analysis with number of restarts.

D. MIMIC Analysis

Continuous Peaks introduces a challenge for MIMIC, emphasizing the algorithm's struggle with premature convergence. The nature of the problem poses difficulties for MIMIC's probabilistic modeling, often leading to premature convergence before thoroughly exploring the solution space. A careful analysis reveals that maintaining a percentage of 0.1 and a population of 500 yields quicker convergence to fitness 120 compared to a higher percentage of 0.3, Figure 7. The preference for the lower percentage suggests that a more conservative modeling process better captures the deceptive structure of the problem. This configuration leads to quicker convergence to fitness, highlighting the delicate balance required in configuring MIMIC for optimal performance in the Continuous Peaks problem.

In conclusion, the Continuous Peaks problem serves as a dynamic testbed for algorithmic evaluation. GA showed the best performance consistently excelling in fitness/iterations but lagging in wall clock time efficiency, prompting considerations for parallelization or advanced optimization strategies.

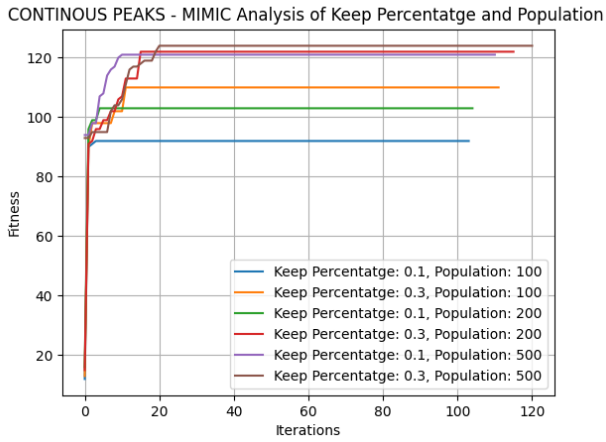

Fig. 7. Continuous Peaks - MIMIC analysis with keep percentage and population.

V. Flip Flop Problem

The analysis of the Flip Flop problem across GA, SA, RHC and MIMIC discovers interesting dynamics. In Figure 8, fitness vs. problem sizes exhibits a positive linear progression for all algorithms, showcasing their scalability. However, in terms of time efficiency, Figure 9, GA diverges significantly, requiring much more time compared to SA and RHC, which barely consume computational resources. In comparison with Continuous Peaks the time efficiency is similar for all algorithms. In this problem, a problem size of 80 is chosen for the subsequent analysis since it provides the best performance in fitness and time for all algorithms. The examination of Figure 10 positions SA as the standout performer, achieving superior results even with more than 7000 iterations. On the contrary, GA, MIMIC and RHC converge prematurely, falling much short of the optimal fitness of almost 80. These outcomes reflect the relation between algorithmic strategies and the unique characteristics of the Flip Flop problem.

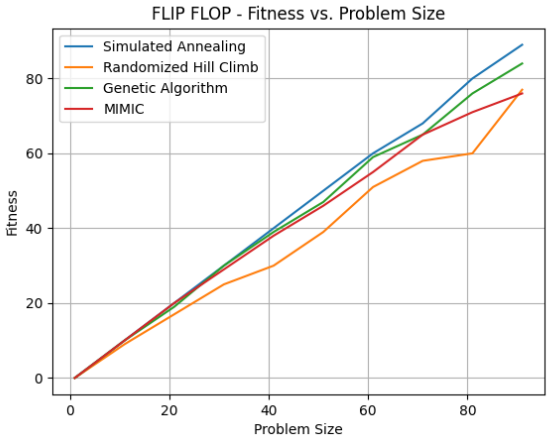

Fig. 8. Flip Flop - Fitness vs. Problem Size comparison.

A. Genetic Algorithms Analysis

Even though GA performance much worse than SA due to its premature convergence, GA achieves best fitness in the Flip Flop prob-

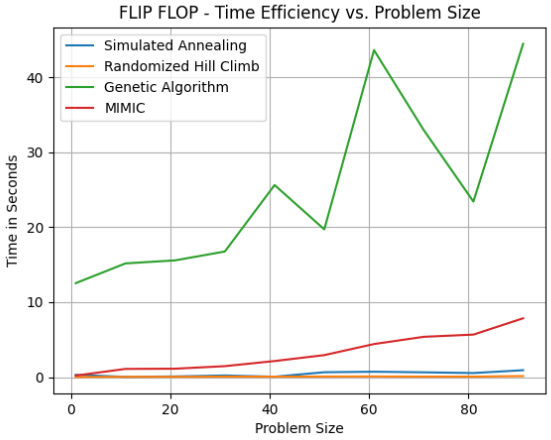

Fig. 9. Flip Flop - Time efficiency vs. Problem Size comparison.

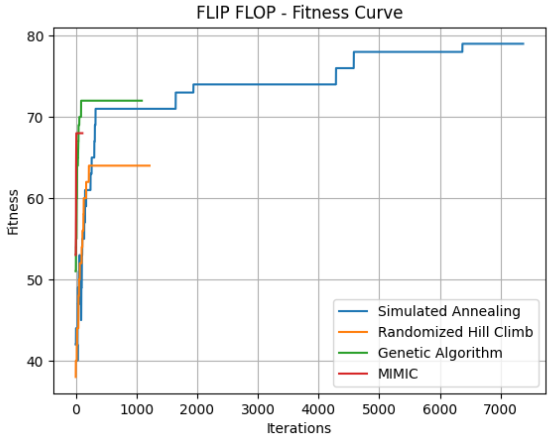

Fig. 10. Flip Flop - Fitness vs. Iterations comparison.

lem when employing a mutation probability of 0.3 and a population size of 200, Figure 11. Other attempts with different hyperparameters converge prematurely without reaching a better fitness. The success of GA lies in distinct a delicate balance between mutation probability and population size. A mutation probability range high 0.3 enables sufficient exploration, preventing premature convergence, while a population size of 200 ensures diversity for effective exploration. This tuning allows GA to perform better other algorithms (RHC and MIMIC) in fitness, demonstrating its better adaptability to the challenges posed by the Flip Flop problem compared to the other two algorithms.

B. Simulated Annealing Analysis

Within the domain of the Flip Flop problem, the performance of SA hinges on the selected decay schedule. As we can see in Figure 12, optimal results manifest with Exponential decay, demonstrating consistent performance. The strength of Exponential decay lies in its gradual reduction of temperature and systematic balance between exploration and exploitation throughout the solution space. In contrast, arithmetic decay introduces noticeable visual fluctuations in fitness over ten thousand iterations. SA stands out as the true choice for Flip

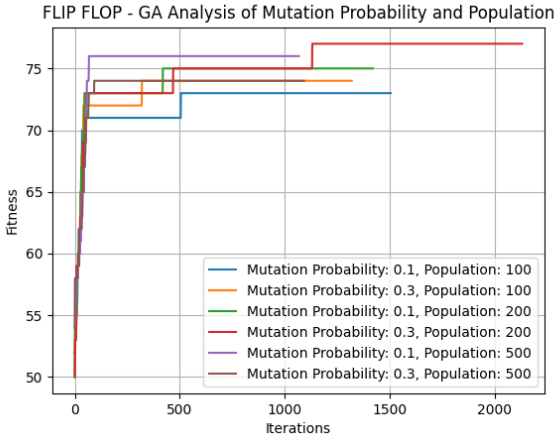

Fig. 11. Flip Flop - Genetic Algorithm analysis with mutation probability and population.

Flip problem, showcasing the best fitness performance and being the only algorithm to avoid premature convergence.

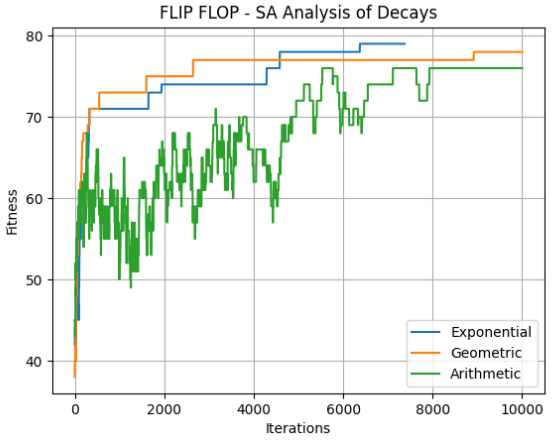

Fig. 12. Flip Flop - Simulated Annealing analysis with decays.

C. Randomized Hill Climbing Analysis

In the exploration of the Flip Flop problem, RHC demonstrates remarkable consistency across different restarts. All restarts perform nearly identically, Figure 13, with the optimal results achieved with 12 restarts converging the earliest. The exception lies in 0 restarts, failing to attain the same fitness level as the others. The impact of restarts becomes evident in overcoming local optima, without restarts, the algorithm may fall to premature convergence. RHC explores local neighborhoods through random moves, and the effectiveness is evident in consistent performance across restarts.

D. MIMIC Analysis

Within the domain of the Flip Flop problem, MIMIC demonstrates improved fitness when configured with a percentage of 0.3 and a population size of 500, achieving convergence in a remarkably low number of iterations (approximately 20), as illustrated in Figure 14. However, this low number of iteration raises concerns about premature convergence. This may be attributed to the algorithm's

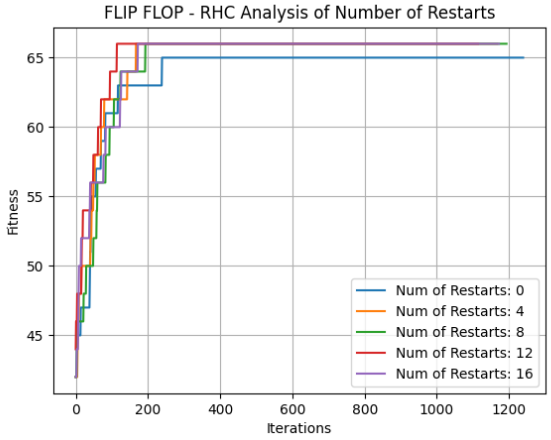

Fig. 13. Flip Flop - Randomized Hill Climbing analysis with number of restarts.

probabilistic modeling approach, which relies on the mutual information between variables to guide the search. In cases where the problem exhibits a deceptive structure, the probabilistic model implemented by MIMIC might converge too quickly to a suboptimal solution, limiting the algorithm's ability to explore the full solution space.

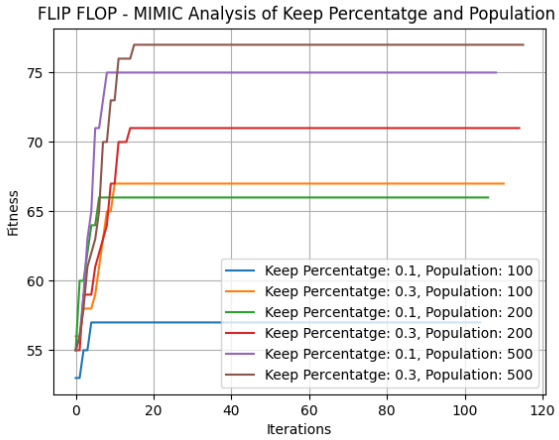

Fig. 14. Flip Flop - MIMIC analysis with keep percentage and population.

SA emerges as the top-performing algorithm for the Flip Flop problem. The key factors contributing to SA's superiority include its adept use of exponential decay, enabling a smooth transition between exploration and exploitation. SA's probabilistic acceptance of worse solutions proves instrumental in escaping local optima. Unlike other algorithms, SA prioritizes solution quality over convergence speed, ensuring it reaches optimal fitness levels even with a higher number of iterations.

VI. Four Peaks Problem

In tackling the intricacies of the Four Peaks problem, GA demonstrates a linear correlation with problem size and fitness, Figure 15. This aligns with GA's proficiency in navigating and exploiting the solution space efficiently. Employing a probabilistic modeling approach, MIMIC showcases linear fitness results, closely trailing GA. Conversely, SA and RHC exhibit greater instability with changing peaks, indicating potential challenges in adapting to varying problem sizes. Notably, GA emerges as the most time-intensive algorithm, Figure 16, demanding substantial computational resources. MIMIC follows as the second-highest in time consumption, same results that Continous Peaks and Flip Flop problem. For the following analysis, problem size is set to 80 since it provides the best performance for all the optimization algorithms, as shown in the previous graphs. When comparing fitness and iterations, Figure 17, MIMIC excels in achieving superior performance with just a few iterations, outpacing all other algorithms in fitness optimization. This early convergence underscores MIMIC's exceptional ability to efficiently capture and exploit the underlying structure of the Four Peaks problem, emphasizing the significance of early convergence for optimal results.

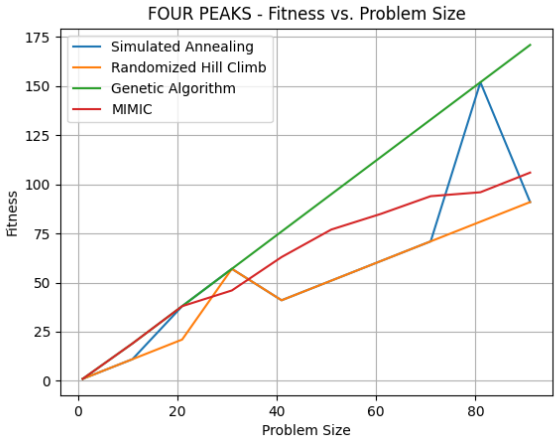

Fig. 15. Four Peaks - Fitness vs. Problem Size comparison.

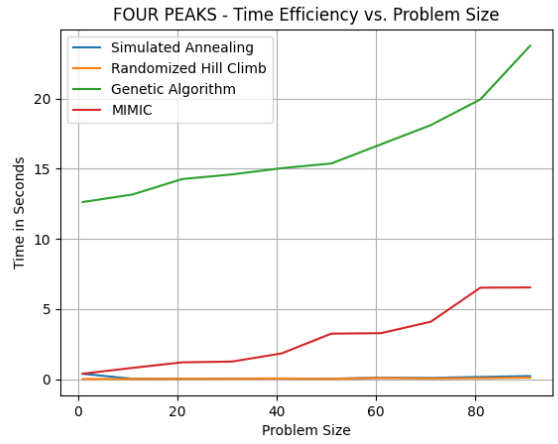

Fig. 16. Four Peaks - Time efficiency vs. Problem Size comparison.

A. Genetic Algorithms Analysis

Among various configurations in GA for Four Peaks, Figure 18, the combination of a mutation probability of 0.1 and a population size of 500 emerges as the optimal choice. While other configurations closely compete in terms of fitness and iterations, one outlier with a probability of 0.3 and a population of 100 performs significantly worse. This discrepancy highlights the sensitivity of GA to specific

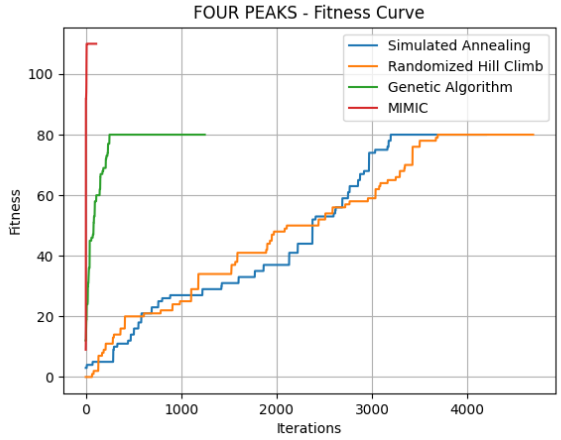

Fig. 17. Four Peaks - Fitness vs. Iterations comparison.

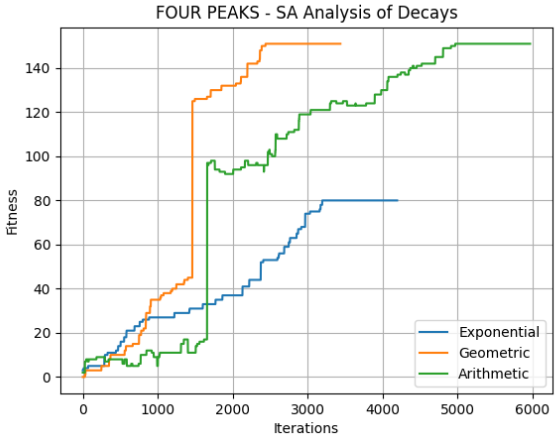

Fig. 19. Four Peaks - Simulated Annealing analysis with decays.

hyperparameter choices and underscores the importance of fine-tuning. GA's mathematical foundation involves the principles of natural selection. The population-based approach allows GA to explore various regions, providing strongness in traversing flat regions. However, tuning crossover and mutation rates becomes crucial for an effective exploration-exploitation trade-off.

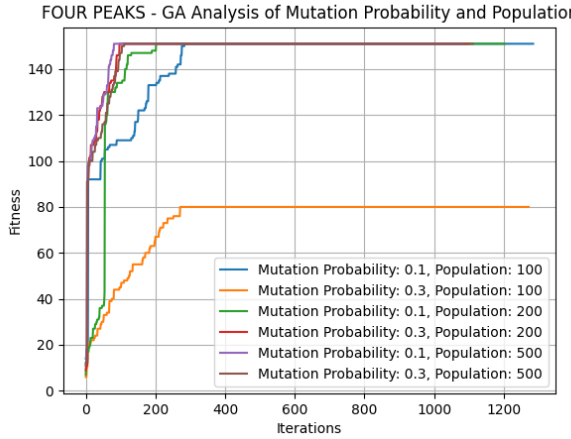

Fig. 18. Four Peaks - Genetic Algorithm analysis with mutation probability and population.

B. Simulated Annealing Analysis

In the context of the Four Peaks problem, SA demonstrates superior performance with geometric decay, Figure 19. This decay schedule proves effective, yielding higher fitness scores with fewer iterations. Geometric decay facilitates a gradual reduction in temperature, striking an optimal balance between exploration and exploitation. The efficiency of SA in this scenario is attributed to its adaptability to the problem's characteristics, allowing it to converge more rapid and achieve higher fitness levels.

C. Randomized Hill Climbing Analysis

RHC showcases a peculiar trend in the Four Peaks problem. As seen in Figure 20 all restarts achieve identical fitness levels, indicating that restarting the algorithm does not significantly impact the final fitness. Surprisingly, the 0 restarts scenario requires fewer iterations, challenging the conventional wisdom of utilizing restarts. The contrast in performance raises interesting questions about the necessity of restarting in the Four Peaks problem and warrants further investigation. RHC is well-suited for the Four Peaks problem due to its simplicity and ability to escape local optima. However, the algorithm may struggle in cases where the initial solution is far from the global optimum.

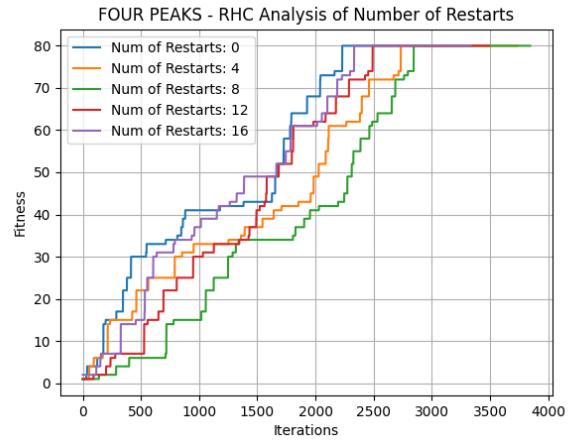

Fig. 20. Four Peaks - Randomized Hill Climbing analysis with number of restarts.

D. MIMIC Analysis

Similar to GA, MIMIC exhibits sensitivity to population size in the Four Peaks problem, Figure 21. Configurations with a population of 100, irrespective of the chosen probabilities (0.1 or 0.3), result in substantially poorer fitness compared to other cases. This emphasizes the critical role of population size in MIMIC's ability to efficiently model and converge to the optimal solution. The divergence in performance underscores the need for careful consideration of hyperparameters. MIMIC, with its probabilistic modeling strategy, excels in capturing and representing the underlying structure of the problem space. The algorithm efficiently samples and models the probability distribution

of candidate solutions, allowing it to explore the diverse regions of the landscape effectively.

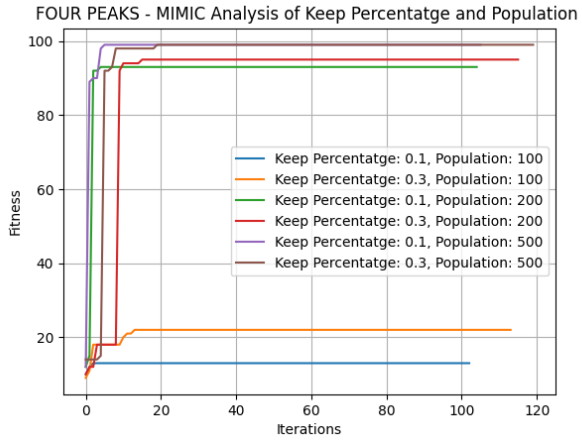

Fig. 21. Four Peaks - MIMIC analysis with keep percentage and population.

In conclusion, MIMIC proves to be a strong and efficient algorithm for addressing the Four Peaks problem. Through its innovative probabilistic modeling, MIMIC navigates the complex landscape of the problem, showcasing adaptability to deceptive structures. MIMIC stands out as a valuable tool for optimization tasks, particularly in scenarios where quick convergence is essential.

VII. Neural Networks with Randomized Optimization

In tandem with exploring discrete optimization problems, this section extends the application of the first three algorithms—Randomized Hill Climbing (RHC), Simulated Annealing (SA), and Genetic Algorithms (GA)—to the world of neural networks. The focus shifts from conventional backpropagation to leveraging these optimization algorithms for determining optimal weights.

VIII. Dataset

Building on the importance of the Wisconsin Diagnostic Breast Cancer (WDBC) dataset, which has previously been employed in A1, its selection for A2 underscores its significance in advancing our understanding of randomized optimization algorithms. With 569 samples and 32 features, the dataset focuses on predicting tumor malignancy based on physical properties. The higher dimensionality and smaller sample size present a challenging scenario for algorithmic classification, particularly within the context of imbalanced data. Thus, the WDBC dataset continues to be a pivotal tool for evaluating the effectiveness of neural networks optimized through randomized techniques in addressing real-world medical classification challenges.

IX. Training Loss

In a maximization problem, the goal is to maximize the objective function or fitness score. However, when plotting the training loss, it is common to negate the fitness values, so the graph still goes down as we can see in our graphs.

A. Backpropagation

The training loss curve for backpropagation reveals a smooth descent, Figure 22, characteristic of the algorithm's iterative refinement process. Convergence is achieved after approximately 1000 iterations, demonstrating a steady and consistent reduction in the loss over time. Backpropagation is known for its deterministic and smooth convergence, gradually adjusting weights through each iteration. The steady decline in the loss indicates a well-behaved optimization process, taking enough iterations to converge.

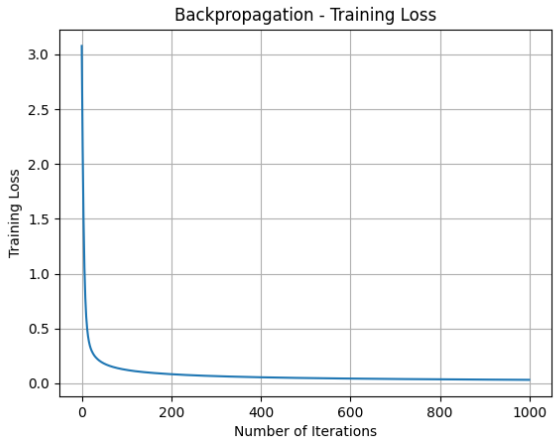

Fig. 22. Backpropagation Neural Network training loss curve.

B. Genetic Algorithm

Contrastingly, as seen in Figure 23, the GA training loss curve exhibits more marked steps, indicating the algorithm's exploration-exploitation trade-off through the evolution of a population of solutions. This is because, GA converges in less than 140 iterations, showcasing its ability to rapidly adapt and refine the neural network weights. The distinctive steps in the curve reflect GA's population-based approach, where each generation introduces diverse solutions. The algorithm efficiently explores the solution space, converging in a shorter time frame, demonstrating the effectiveness of genetic operators in traversing the optimization problem.

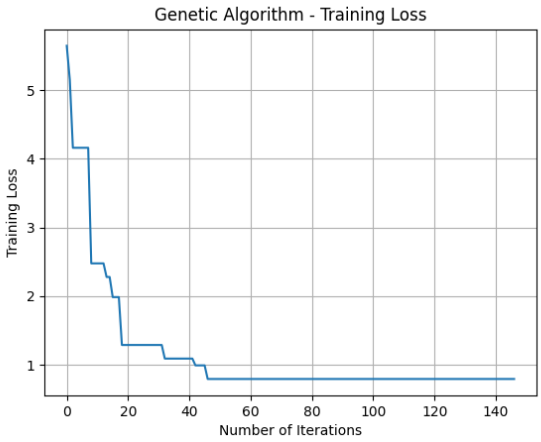

Fig. 23. Genetic Algorithm Neural Network training loss curve.

C. Randomized Hill Climbing

Similar to the smooth descent observed in backpropagation, RHC presents a refined training loss curve. However, RHC takes approximately twice as many iterations compared to backpropagation to achieve convergence, suggesting a more cautious exploration of the solution space, Figure24. While the smooth curve indicates a

stable optimization process, the extended convergence time may be attributed to the algorithm's reliance on local search, which can hinder the discovery of global optima.

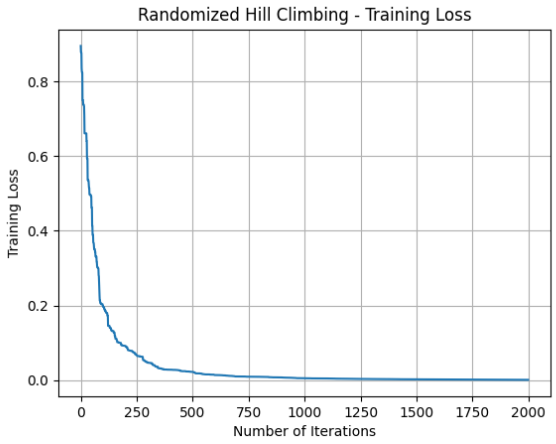

Fig. 24. Randomized Hill Climbing Neural Network training loss curve.

D. Simulated Annealing

The training loss curve for SA reveals an interesting pattern as is shown in Figure 25. In the initial 200 iterations, the algorithm does not exhibit a clear descent in the training loss. However, after this initial phase, SA converges. The convergence of SA is achieved after almost 2000 iterations, emphasizing its distinctive annealing process and the importance of an extended exploration period. SA's initial lack of progress can be attributed to its exploration-exploitation balance. The algorithm's acceptance of worse solutions early on facilitates exploration, delaying convergence. The subsequent decline indicates a shift toward exploitation as the annealing schedule adapts, ultimately converging after extensive exploration.

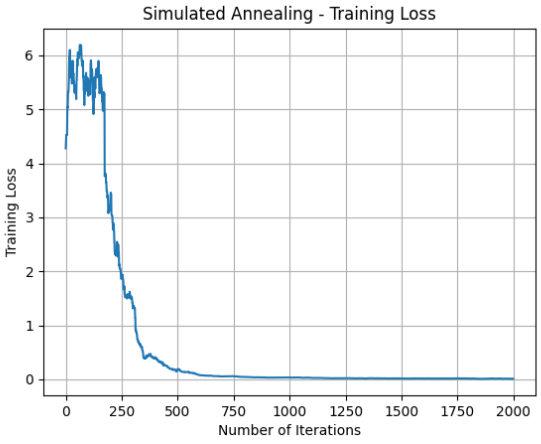

Fig. 25. Simulated Annealing Neural Network training loss curve.

X. Performance

Neural networks stand as versatile tools in machine learning, and optimizing their training process is crucial for achieving efficient convergence and strong generalization. One fundamental aspect of optimization algorithms is the time they require to converge. In our experiments, Figure 26, RHC takes around 10 seconds per iteration, SA takes around 4.5 seconds, GA takes around 5 seconds, and Backpropagation requires around 3 seconds. These variations in time can be attributed to the distinct optimization strategies employed by each algorithm. RHC, involving iterative local search and exploration-exploitation dynamics, naturally consumes more time per iteration. In contrast, GA, with its population-based approach, SA, known for its efficient exploration, and Backpropagation, a deterministic optimization method, exhibit comparatively shorter iteration times.

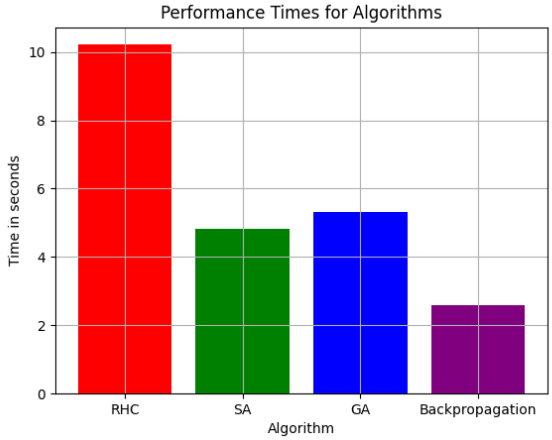

Fig. 26. Performance time for each randomized algorithm.

In this evaluation, the dataset is divided into training and testing sets, with 80% allocated for training and 20% for testing. Further, the training set is then subdivided into 80% for actual training and 20% for validation. This split ensures that the testing set is entirely new for the algorithm, allowing an unbiased assessment of the model's performance on unseen data.

The subsequent evaluation of model scores offers insights into how well each optimizer generalizes to unseen data and addresses potential overfitting to the training set. In the training phase, Figure 27, RHC achieves a perfect score of 100%, hinting at overfitting as the model may have memorized the training data. SA and Backpropagation both score just under 99%, demonstrating strong generalization to the training set, while GA achieves a slightly lower score, just under 98%, showcasing less effective generalization capabilities. Transitioning to the validation set, Figure 28, both RHC and GA exhibit almost 98% accuracy, reflecting their ability to generalize effectively, whereas Backpropagation achieves a slightly lower score of 97%, indicating good generalization. SA performs well on unseen data, with a score of 95.5% but not as good as the other algorithms.

In the test phase, Figure 29, RHC, despite potential overfitting, achieves almost 94% accuracy. SA demonstrates strong generalization with a test score of 97%, while GA outperforms other optimizers with over 98% accuracy. Backpropagation, though achieving a respectable 95.5%, shows some sensitivity to unseen data. In conclusion, the diverse training times and model scores among optimizers reveal detailed trade-offs in their optimization approaches, providing GA as the better option for this training set problem. GA is able to perform better than the other algorithms for the validation set, while keeping a good training phase without signs of overfitting on the samples. One of the explanations on why GA performs great on the Cancer Dataset which is small but with great number of features is because the recombination process in GA, particularly crossover, allows for

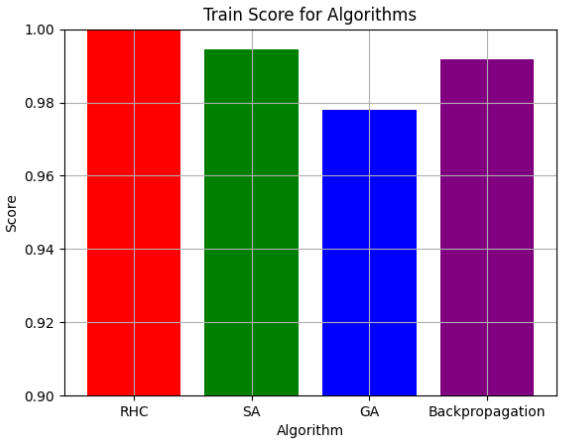

Fig. 27. Training score for each randomized algorithm.

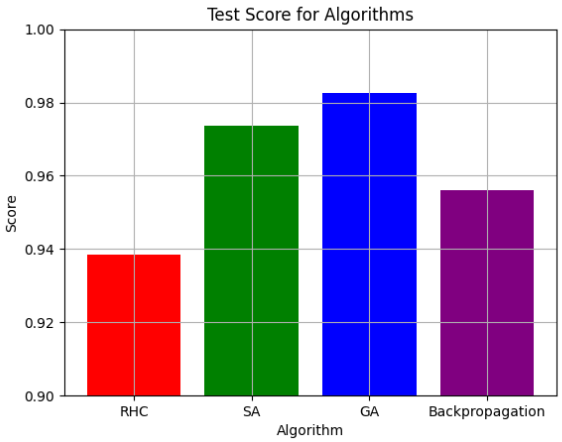

Fig. 29. Testing score for each algorithm.

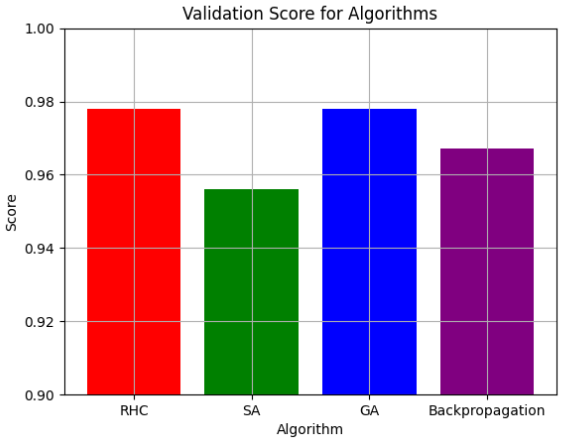

Fig. 28. Validation score for each randomized algorithm.

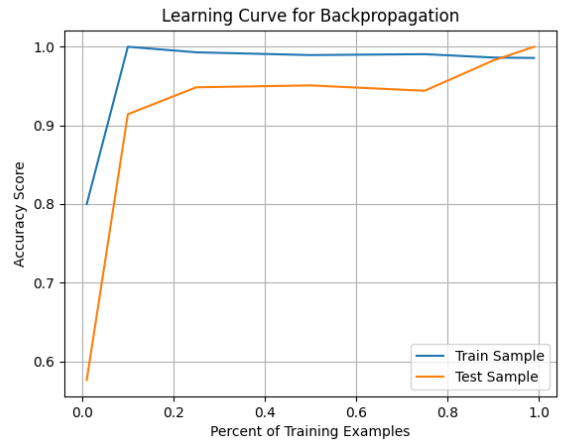

Fig. 30. Backpropagation Neural Network Learning Curve.

the combination of features from different individuals. This can be advantageous in identifying effective combinations of features that contribute to the overall performance of the algorithm.

XI. Learning Curve

Understanding the learning curves for each optimizer provides a dynamic perspective on their performance across varying training sample sizes.

A. Backpropagation

The learning curve for Backpropagation reveals strong performance in the training samples, reaching up to a 99% score in Figure 30. This suggests the model's ability to learn intricacies from the training data. However, the consistently high test score, while desirable, raises concerns about potential overfitting, indicating that the model may have memorized the training set specifics rather than generalizing well to new data.

B. Genetic Algorithm

The learning curve for GA depicts an interesting pattern. Initially, there is evidence of overfitting with a few training samples, Figure 31, as indicated by a high training score. However, as the number of training samples increases, the overfitting diminishes, and the test score improves. This behavior highlights GA's adaptability to diverse data and its capacity to achieve better generalization with an increased amount of training samples.

C. Simulated Annealing

SA's learning curve exhibits a tendency toward overfitting in the first half of the training data, with a high training score, Figure 32. However, the test score shows a declining trend, indicating potential challenges in generalizing to unseen data. As the entirety of the training samples is considered, the test score gradually improves, demonstrating SA's ability to enhance generalization with more extensive training.

D. Randomized Hill Climbing

RHC's learning curve serves as a clear example of overfitting, Figure 33, showcasing a perfect training score for all samples. However, the test score performs significantly poorer compared to other algorithms. This emphasizes the limitations of RHC in generalizing to unseen data, leading to suboptimal test performance.

The observed overfitting in the learning curves aligns with the overfitting trends noted in the test and validation results. This discrepancies can be attributed to the models' tendency to memorize training

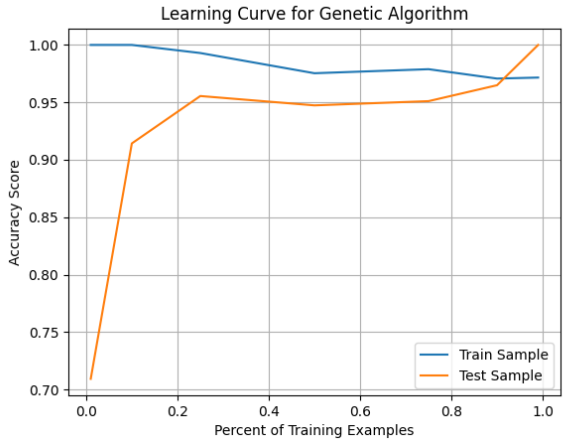

Fig. 31. Genetic Algorithm Neural Network Learning Curve.

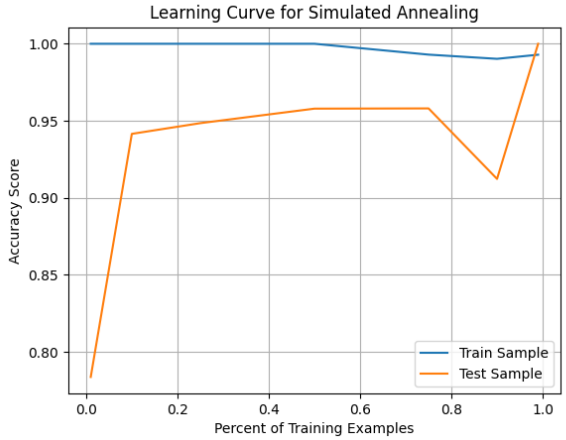

Fig. 32. Simulated Annealing Neural Network Learning Curve.

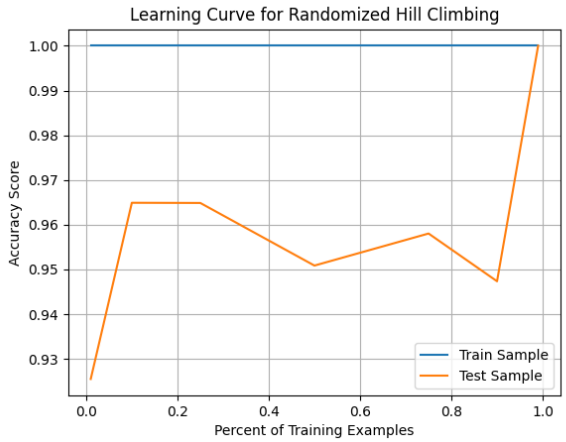

Fig. 33. Randomized Hill Climbing Neural Network Learning Curve.

data specifics. Learning curves provide a visual representation of this phenomenon, where overfitting is evident when the training score remains consistently perfect while the test or validation score diverges. Notably, both Genetic Algorithms (GA) and Backpropagation showed the best performance on the learning curves. However, it is important to highlight that Backpropagation exhibited signs of overfitting during the training phase, as evidenced by the widening gap between the training and validation/test scores. In contrast, GA demonstrated a smooth convergence in both training and testing phases, without apparent signs of overfitting. Considering these observations, GA emerges as a promising algorithm for this specific dataset problem, showcasing effective generalization and stability across different phases of the machine learning process.

XII. Conclusion

This exploration of randomized optimization algorithms has provided valuable insights into their performance across discrete optimization problems and neural network training. The chosen algorithms—Randomized Hill Climbing (RHC), Simulated Annealing (SA), Genetic Algorithms (GA), and Mutual-Information-Maximizing Input Clustering (MIMIC)—were systematically applied to three discrete optimization problems: Continuous Peaks, Flip Flop, and Four Peaks.

In the context of the Continuous Peaks problem, GA showcased superior fitness across various problem sizes, demonstrating its adaptability. SA and MIMIC demonstrated competitive fitness, while RHC faced challenges due to its randomness component. Although GA achieved superior fitness, it required significantly more time. The learning curve analysis further emphasized the strengths and weaknesses of each algorithm, highlighting GA's strong convergence, SA's adaptability, and MIMIC's struggle with premature convergence.

The exploration of the Flip Flop problem revealed SA as the standout performer, showcasing the best fitness performance and being the only algorithm to avoid premature convergence. GA achieved good fitness but faced premature convergence issues, and RHC struggled due to its quick convergence to local optima. The learning curve analysis shed light on the importance of decay schedules for SA and the challenges faced by MIMIC in efficiently navigating the Flip Flop problem's landscape.

In addressing the Four Peaks problem, MIMIC demonstrated a linear correlation with problem size, showcasing superior fitness achievement. MIMIC proved to be a strong and efficient algorithm, excelling in achieving superior performance with just a few iterations. The learning curve analysis further emphasized GA's proficiency and MIMIC's exceptional ability to efficiently capture and exploit the underlying structure of the Four Peaks problem.

The extension of these algorithms to neural network for the WDBC dataset training provided insights into their diverse performance. RHC exhibited potential overfitting, GA showcased effective generalization, SA demonstrated strong performance, and Backpropagation showed sensitivity to unseen data. The learning curve analysis focus on the trade-offs in generalization capabilities, with some algorithms exhibiting overfitting tendencies and highlighting GA as the best performance algorithm for the WDBC dataset problem.

In conclusion, this research contributes to the understanding of randomized optimization algorithms across different problem domains. The exploration of neural network training further extends the applicability of these algorithms, offering a comprehensive perspective on their performance in diverse machine learning scenarios.