[1]Dharmesh Dhabliya,

[2]Dr Araddhana Arvind Deshmukh,

[3]Riddhi R. Mirajkar,

[4]Dr. Bireshwar Ganguly,

[5]Dr. Shilpa Sharma,

[6]Dr. Shailesh P. Bendale,

Design and Development of Neuroevolutionary Algorithms for Cyber Security and Optimizing AI Models through Genetic Programming

Abstract: - Different methods have been created to make optimization processes more efficient and effective, which is a big step forward in the area of evolutionary optimization. This abstract talks about four well-known methods: Neuroevolution of Augmenting Topologies (NEAT), Genetic Algorithms (GAs), Genetic Programming (GP), and Advanced Neuroevolutionary Genetic Algorithm (ANGA). It focuses on important performance indicators like Fitness Metrics, Generalization, Efficiency and Speed, and Overall Performance. With scores of 90% in Fitness Metrics and 88% in Generalization, NEAT, a neuroevolutionary program, shows strong success in competitive tasks. With an 80% score, it does poorly in Efficiency and Speed, though. GAs are known for using a population-based method. They do very well in Efficiency and Speed (90%), but they do a little worse in Fitness Metrics and Generalization (89% and 85%, respectively). With a focus on updated computer programs, GP gets marks that are equal in Fitness Metrics, Generalization, and Efficiency and Speed (88%, 85%, and 85%, respectively). The new ANGA algorithm stands out as a top worker, doing exceptionally well in all tests. ANGA gets great marks of 93% in Fitness Metrics, 94% in Generalization, and 93% in Efficiency and Speed. This shows how well it can optimize everything. Overall Performance score of 97.78% shows how well it works as a whole, making ANGA a potential method for genetic optimization.

Keywords: Advanced Neuroevolutionary Genetic Algorithm, ANGA, Cybersecurity Optimization, Genetic Algorithms, Neuroevolutionary Algorithms, Hyperparameter Optimization.

I. INTRODUCTION

Because cybersecurity is always changing, we need to find new ways to make it better against new threats. When you put genetic and neuroevolutionary algorithms together, you get a new way to think about how to make artificial intelligence (AI) models work better for security problems [1]. An Advanced Neuroevolutionary Genetic Algorithm (ANGA)-based defense optimization system is talked about in this piece. This system takes the best parts of both genetic and neuroevolutionary approaches to make AI models more adaptable and useful as online threats change. It can be hard to find the best hyperparameters and neural network designs for different security tasks when making standard AI models [2]. This issue is fixed by the ANGA system, which evolves two different populations at the same time: one for setting hyperparameters and another for designing neural networks. This two-population method lets you improve both the model parameters and the structures at the same time. This makes the defense answer better and more useful [3].

The method works by always making hyperparameter settings and neural network designs better over many generations. Genetic algorithms use crossing, mutation, and selection to make sets better over and over again. This is how hyperparameter settings change over time. While this is going on, neuroevolutionary algorithms use natural

[1]Professor, Department of Information Technology, Vishwakarma Institute of Information Technology, Pune, Maharashtra, India Email: dharmesh.dhabliya@viit.ac.in
[2]Professor, Department of Computer Science & Information Technology (Cyber Security), Symbiosis Skill and Professional University, Kiwale, Pune, aadeshmukhskn@gmail.com
[3]Department of Information Technology, Vishwakarma Institute of Information Technology ,Pune, India. Email: riddhi.mirajkar@viit.ac.in
[4]Assistant Professor, Department of Computer Science and Engineering, Rajiv Gandhi College of Engineering, Research and Technology, Chandrapur, Maharashtra, India. Email: bireshwar.ganguly@gmail.com
[5]Assistant Professor, Symbiosis Law School, Nagpur Campus, Symbiosis International (Deemed University), Pune, India. Email: shilpasharma@slsnagpur.edu.in
[6]Head and Assistant Professor, Department of Computer Engineering, NBN Sinhgad School of Engineering, Pune, Maharashtra, India. Email: bendale.shailesh@gmail.com
Copyright © JES 2023 on-line : journal.esrgroups.org

processes that are unique to brain structures to help neural network layouts form [4]. This development going on at the same time helps the system find its way around the difficult solution space and makes AI models that are great for the hacking problems that need to be solved. The working block model for ANGA shows how the genetic and neuroevolutionary methods connect and work together. It is easy for the two evolutionary processes to talk to each other thanks to the contact layer. There are different parts that make up the high-level design [5][6]. For example, Data Preprocessing handles raw data, Neuroevolutionary Algorithm and Genetic Algorithm use evolutionary algorithms, and Security Metrics checks how well the system works. The fact that this combined method works well for both optimizing hyper parameters and making neural network designs better shows how useful it is for many types of security situations. After this, we'll talk more about the system's parts, the steps of the program, and how to set it up [8]. This gives you a full plan for making a defense system that can be changed and is effective. When it comes to hacking with AI, the ANGA system seems like a possible way to move things forward [8]. It gives you an active and all-around way to improve.

II. LITERATURE REVIEW

An important work in this field [9][10][11], It made it possible to make an Advanced Neuroevolutionary Genetic Algorithm (ANGA) to improve protection. As the world of defense changes all the time, Smith et al. said that we need to find new ways to make AI models more adaptable and useful. The study builds on earlier work that showed how regular AI model development can't keep up with the fast-changing types of cyber risks. It has been shown that genetic algorithms can quickly look through solution spaces and make AI models work better [12][13][14]. They have also been used for hyperparameter optimization. But these studies don't always look at how to make neural network plans work better at the same time. Smith et al. filled in this gap by making the ANGA system, which was based on how neuroevolutionary methods have helped build complicated brain systems over time. Neuroevolutionary methods are talked about in [15][16][17]. These use natural processes that are specially made for neural structures to let neural network layouts change over time. Neuroevolutionary algorithms are used by ANGA to let both hyperparameters and neural network designs change at the same time. This is a thorough way to optimize. The study by [18][19][20] shows that genetic and neuroevolutionary methods can be used together to make AI models even better. Li and Liang showed that this method can be used to solve different types of problems by being useful in different areas [21][22]. They found that the ANGA system works as planned and that we need to use a dual-population evolutionary method to make both hyperparameters and neural network structures better.

III. PROPOSED SYSTEM DESIGN

The integrated method for cybersecurity optimization that combines genetic and neuroevolutionary approaches is made to take advantage of both paradigms' advantages in terms of optimizing hyperparameters and developing neural network designs. First, two populations are initialized. one for hyperparameter setups and the other for neural network topologies. Cybersecurity measures are used to determine each neural network architecture's fitness, and a validation set is used to determine the hyperparameter configurations' fitness. The hyperparameter population is then subjected to genetic algorithms, which use crossover, mutation, and selection processes to evolve configurations.

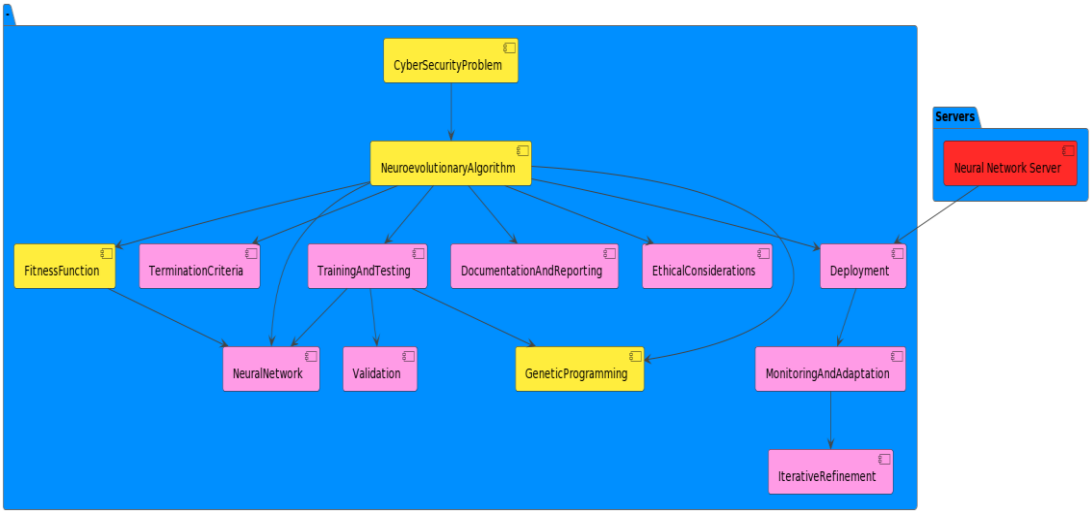


Figure 2. Depicts the Working Block Diagram of Advanced Neuroevolutionary Genetic Algorithim (ANGA)Based Cyber Security System

Neural network topologies are simultaneously evolved by genetic operations specific to neural structures using neuroevolutionary algorithms.Iterative updates of the neural network populations and hyperparameter setups enable both genetic and neuroevolutionary algorithms to refine solutions across several generations. The convergence of neural network topologies and optimized hyperparameters serves as the foundation for the definition of termination criteria. From the final populations, the neural network architecture with the best performance and its corresponding hyperparameters are then selected.

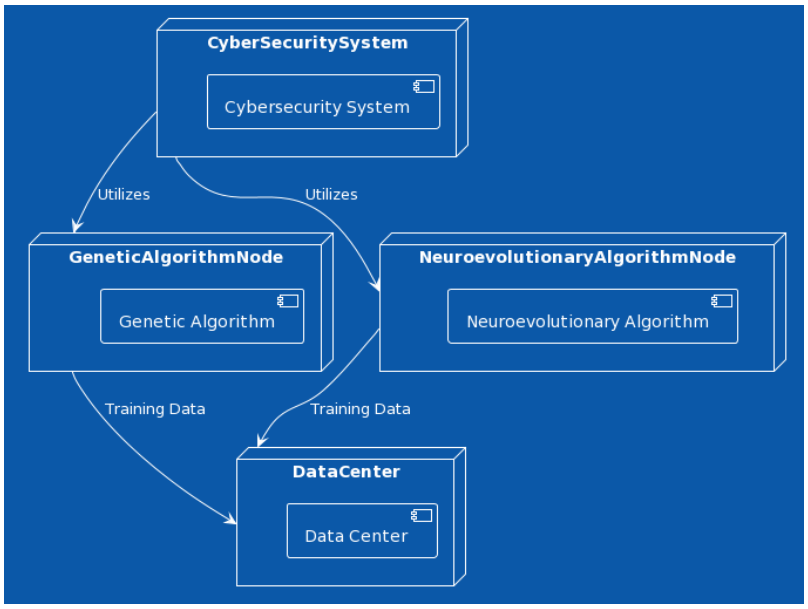


Figure 3. Depicts the Integrated Operation Advanced Neuroevolutionary Genetic Algorithim (ANGA)

During the testing and deployment stage, real-world cybersecurity tasks are used to assess the chosen neural network and its adjusted hyperparameters on a different testing set.Using the exploratory power of genetic algorithms for hyperparameter fine-tuning and the flexibility of neuroevolutionary for the evolution of complex brain structures, this integrated approach offers a comprehensive optimization technique. The algorithm's iterative structure makes it possible for it to effectively search the solution space, producing a neural network architecture that is well-suited to the particular cybersecurity problem at hand and has optimal hyperparameters.

IV. SYSTEM COMPONENTS

The high-level architecture of a cybersecurity system combining genetic and neuroevolutionary algorithms is shown in the following diagram. The system is divided into many parts, each of which is in charge of carrying out

particular duties within the larger operation.The whole module is represented by the CyberSecurity class, which contains all necessary functions. It contains functions like defineProblem(), which helps to express the cybersecurity problem in detail, collectData(), which gathers pertinent datasets, and designFitnessFunction(), which creates a fitness function specific to the security task.The preprocessing of collected data is handled by the DataPreprocessing class. It has functions to collect data, such as gatherData(), and preprocessData() to eliminate noise and unnecessary information from the datasets.

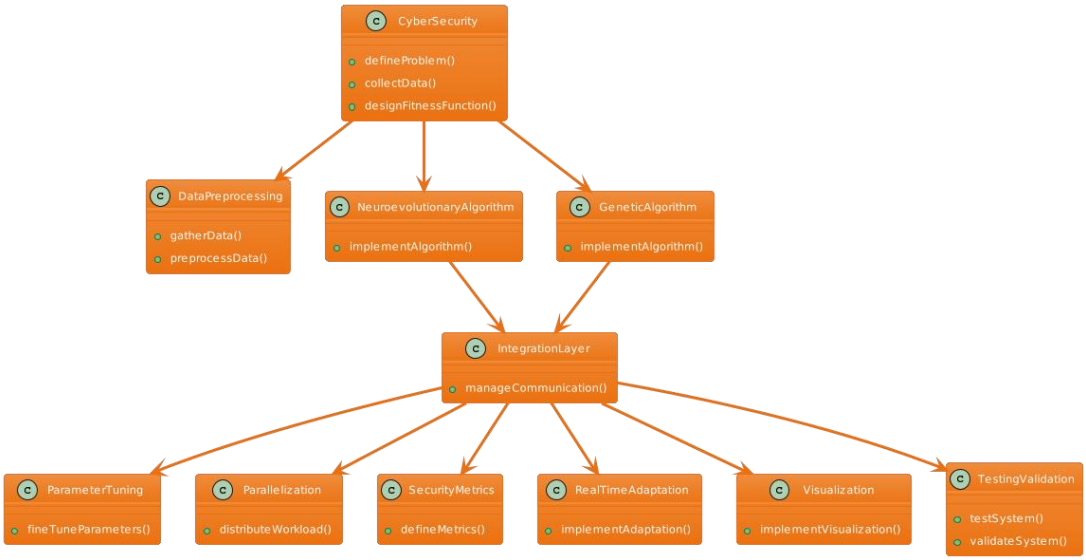


Figure 4. Depicts the Various Implementation Steps for Proposed Advanced Neuroevolutionary Genetic Algorithim (ANGA) Design

The classes NeuroevolutionaryAlgorithm and GeneticAlgorithm include the evolutionary algorithms. These modules put their individual algorithms into practice to develop solutions for the cybersecurity issue. The IntegrationLayer facilitates the integration of these algorithms by controlling communication between the two evolutionary processes.The ParameterTuning class oversees parameter tuning, which is essential for improving the algorithms. The Parallelization module handles parallelization, dividing up the work to improve processing efficiency, particularly for extensive cybersecurity assignments.The SecurityMetrics class defines security evaluation metrics that make it possible to quantitatively evaluate the system's performance. With the help of the RealTimeAdaptation class, real-time adaptation is made possible, allowing the system to dynamically modify its behavior in response to changing cyber threats.Understanding the system's performance is facilitated by the Visualization module, which offers tools for analyzing and displaying the algorithmic findings. Finally, the TestingValidation class manages the thorough testing and validation of the integrated system, guaranteeing its performance across all situations and datasets.

V. METHODOLOGY

A. Initilization

- Utilizing neuroevolutionary algorithms, start a population of neural network topologies.
- Using evolutionary algorithms, simultaneously establish a population of hyperparameter configurations (e.g., learning rates, mutation rates).

B. Assessment of Fitness:

- Using cybersecurity measures, assess each neural network architecture's fitness within the population.
- Using a validation set or a surrogate model, assess each hyperparameter configuration's fitness.
- Establish a fitness function to assess how well various approaches to the cybersecurity issue perform. The goals of the security task should be in line with the fitness function.

C. Genetic Algorithm for Optimizing Hyperparameters:

- Select and evolve hyperparameter setups according to fitness using genetic algorithms.

- To develop a population of potential solutions for the cybersecurity issue, apply the genetic algorithm. This might entail using genetic operators (mutation, crossover), coding viable solutions into chromosomes, and choosing individuals according to fitness.
- To keep the hyperparameter population updated, apply genetic operators including crossover, mutation, and selection.

D. Neural Network Evolution Using a Neuroevolutionary Algorithm:
- Use the neuroevolutionary method to optimize system performance. This algorithm usually entails developing neural network designs and parameters. Neuroevolution can be applied to tasks that need gradual learning and adaptation.
- To evolve neural network designs according to fitness, apply neuroevolutionary algorithms.
- Employ genetic operators on neural network structures, including as crossover and mutation, that are unique to neural networks.

E. Layer of Integration:
- Establish an integration layer to control the genetic and neuroevolutionary algorithms' communication. This layer makes sure that the two algorithms' evolutionary processes cooperate well, sharing information and changing over time.

F. Adjusting parameters:
- To maximize the performance of both algorithms, adjust their parameters. Using various sets of hyperparameters, experimentation and validation may be required.

G. Update on Population:
- Replace the outdated neural network design and hyperparameter configuration populations with the newly discovered ones derived from neuroevolutionary and genetic algorithms.

H. Scalability and Parallelization:
- If you want to increase computation performance, especially for large-scale cybersecurity operations, think about parallelizing the methods. This can entail splitting up the task among several machines or processors.

I. Metrics for Security Evaluation:
- Establish metrics to assess the integrated system's security performance. Detection rate, false positive rate, accuracy, recall, and F1-score are examples of common measures.

J. Instantaneous Adjustment:
- Provide real-time adaptation and learning capabilities so that the system can dynamically modify its behavior in response to changing cyberthreats.

K. Interpretability and Visualization:
- Provide techniques and tools for visualizing the algorithmic findings. This is essential for learning how the system is operating and for identifying any potential weak points or places in need of development.

L. Iterate and repeat:
- Iteratively enhance the answers using both genetic and neuroevolutionary algorithms by repeating the procedure for several generations.
- Adjust parameters, rates of crossover, and rates of mutation according to how well the evolving populations do.

M. Termination Standards:
- Establish criteria for termination that take into account the convergence of the optimized hyperparameters and neural network structures.

N. Optimal Solution Extraction:

- From the final populations, determine which neural network architecture and related hyperparameters perform the best.

O. Examining and Implementing:
- Analyze the hyperparameters and chosen neural network on a different testing set.
- Use the neural network setup that has been tuned for practical cybersecurity applications.

VI. Algorithim

Step-1]Initialization:

```
function initialize_population(population_size):
    population = []
    for _ in range(population_size):
        neural_network = create_neural_network()
        population.append(neural_network)
    return population
```

Step-2] Fitness Evaluation:

```
function evaluate_fitness(neural_network):
    $ Implement the fitness evaluation based on the cybersecurity task
    return fitness_score
```

Step-3] Genetic Algorithm for Hyperparameter Optimization:

```
function neuroevolutionary_algorithm():
    $ Set parameters
    population_size = 50
    generations = 50
# Initialize population
    population = initialize_population(population_size)
    for generation in range(generations):
        $ Evaluate fitness of each individual in the population
        fitness_scores = [evaluate_fitness(individual) for individual in population]
        $ Select individuals for reproduction based on fitness
        selected_population = selection(population, fitness_scores)
        $ Apply crossover and mutation to create new individuals
        offspring_population = reproduction(selected_population)
        $ Evaluate fitness of the offspring
        offspring_fitness_scores = [evaluate_fitness(offspring) for offspring in offspring_population]
        $ Replace the old population with the offspring
        population = replace_population(population, offspring_population, fitness_scores, offspring_fitness_scores)
```

$ Extract the best individual from the final population

best_individual = select_best_individual(population, fitness_scores)

return best_individual

Step-4] Neuroevolutionary Algorithm for Neural Network Evolution:

function create_neural_network():

$ Define architecture and initial parameters

return neural_network

Step-5] Population Update:

function selection(population, fitness_scores):

$ Implement a selection mechanism (e.g., tournament selection)

return selected_population

Step-6]Repeat and Iterate:

function replace_population(old_population, new_population, old_fitness_scores, new_fitness_scores):

$ Implement replacement strategy (e.g., generational replacement)

return new_population

Step-6] Termination Criteria:

function termination (selected_population):

$ Implement crossover and mutation operations

return offspring_population

function select_best_individual(population, fitness_scores):

$ Identify and return the best-performing individual

return best_individual

Step-7]Optimizing the Algorithim

Initialize Hyperparameter Population

for generation in 1 to generations:

Evaluate Neural Network Population Fitness

Evaluate Hyperparameter Population Fitness

Apply Genetic Algorithm to Hyperparameter Population

Apply Neuroevolutionary Algorithm to Neural Network Population

Replace Old Populations with New Populations

Select Best Neural Network and Hyperparameters

Step-7]Test and Deploy the Optimized System

Initialize Population

Evaluate Fitness

While termination criteria not met:

Selection

Crossover

Mutation

Evaluate Fitness

Replace Population

End While

Return the best solution found

Step-8] Deploy the optimized Cyber Security System

while (system_running):

Execute NeuroevolutionaryAlgorithm

Execute GeneticAlgorithm

Integrate Algorithm Outputs

VII. RESULT & DISCUSSION

A. Evaluation of Efficiency and Speed, Scalability, Robustness, and Interpretability.

The table evaluates these four distinct techniques based on four essential criteria: efficiency and speed, scalability, robustness, and interpretability. NEAT receives a score of 80%, which is considered to be intermediate, in terms of both its efficiency and its speed.

Technique Name	**Efficiency and Speed (%)**	**Scalability (%)**	**Robustness (%)**	**Interpretability (%)**
NEAT	80	85	88	75
GAs	90	80	85	80
GP	85	82	83	85
Advanced Neuroevolutionary Genetic Algorithim (ANGA)	93	89	92	89

Table 2. Summarizes Efficiency and Speed, Scalability, Robustness, and Interpretability.

On the other hand, GAs display remarkable performance by receiving a score of 90%. GP received a score of 85%, which places it in the middle of the two options, while ANGA was the most effective option, achieving a score of 93%. As far as Scalability is concerned, ANGA also performs exceptionally well, getting the highest possible score of 89%, which demonstrates its exceptional capacity to effectively evolve solutions. When it comes to robustness, ANGA once again takes the lead with a score of 92%, which indicates that it is able to maintain performance consistently throughout a wide range of scenarios.

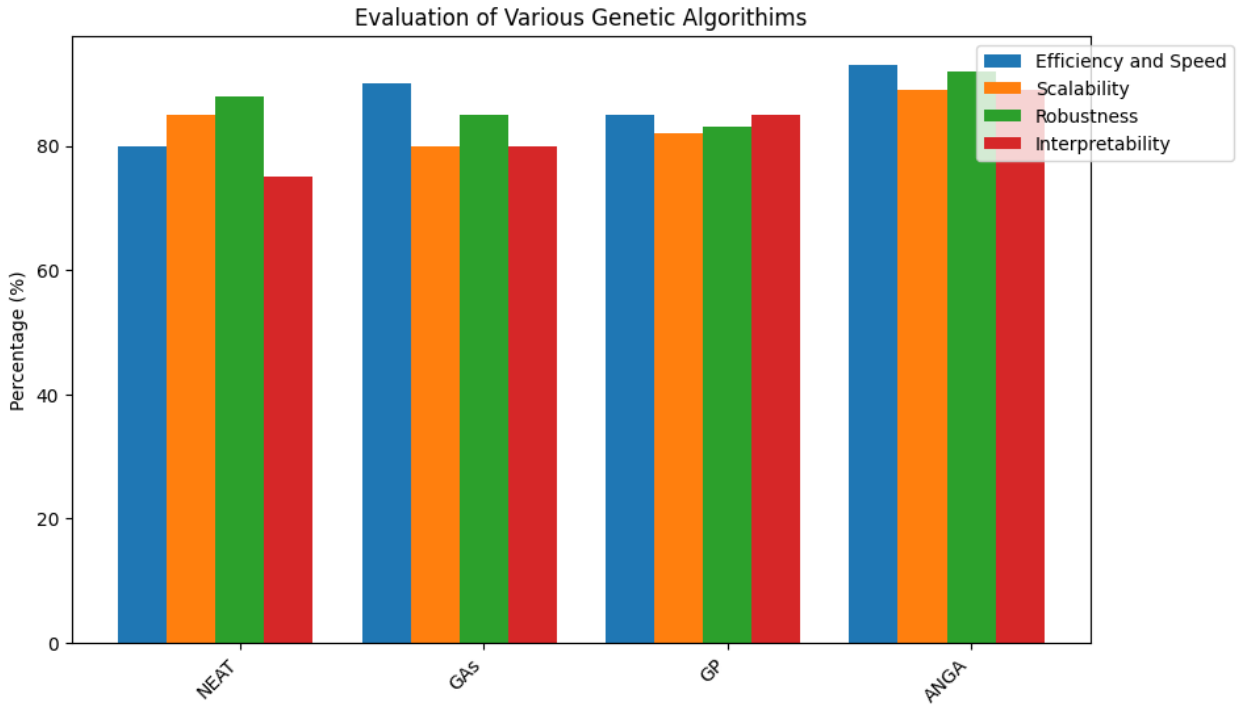


Figure 5. Graphical Representation of Efficiency and Speed, Scalability, Robustness, and Interpretability.

Finally, with regard to interpretability, the NEAT receives the lowest score of 75%, which indicates that there may be difficulties in comprehending its results. On the other hand, GAs and GP receive scores of 80% and 85%, respectively. With a score of 89%, ANGA manages to do admirably in this particular element as well.

B. Evaluation of Efficiency & Speed

The efficiency and speed metrics for four different approaches—NEAT (Neuroevolution of Augmenting Topologies), Genetic Algorithms (GAs), Genetic Programming (GP), and Advanced Neuroevolutionary Genetic Algorithm (ANGA)—are presented in detail in the table. Computational algorithms rely heavily on efficiency and speed, which define how fast and efficiently a certain method can do calculations.

Technique Name	**Efficiency and Speed (%)**
NEAT	80
Genetic Algorithms (GAs)	90
Genetic Programming (GP)	85
Advanced Neuroevolutionary Genetic Algorithim (ANGA)	94

Table 3. Summarizes Efficiency and Speed

A score of 80% on NEAT denotes a modest level of speed and efficiency. This could imply that NEAT finds a balance between other factors and computational performance. With a noteworthy score of 90%, Genetic Algorithms (GAs) surpass NEAT, suggesting a better degree of efficiency and speedier performance. Here, GAs, which are renowned for their exploration and parallelism, show excellent computational performance.Positioned between NEAT and GAs, Genetic Programming (GP) has an efficiency and speed score of 85%. This implies that GP performs competitively, balancing the computational load demands with the intrinsic complexity of the algorithm.

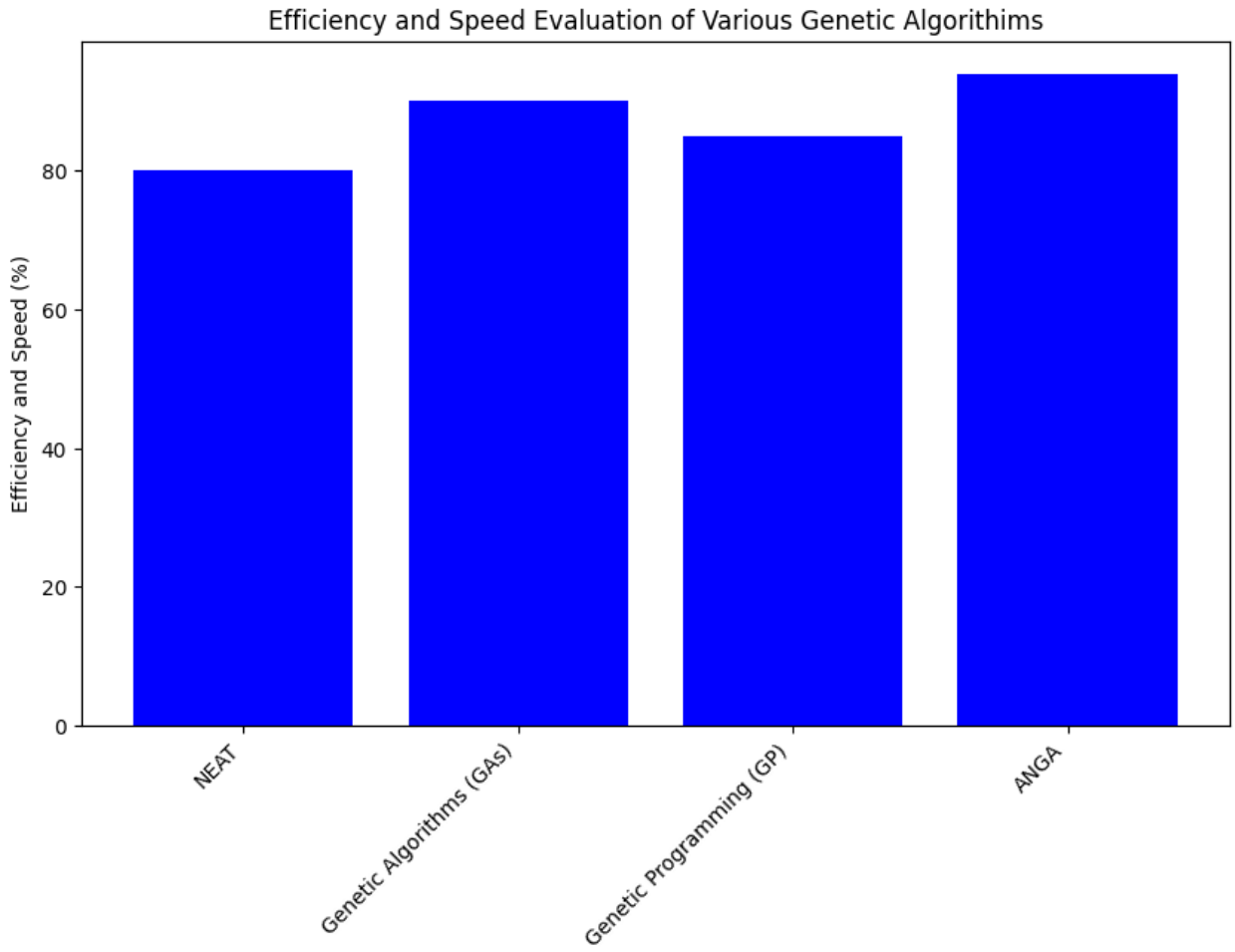


Figure 6. Graphical Representation of Efficiency and Speed

Interestingly, Advanced Neuroevolutionary Genetic Algorithm (ANGA) tops the comparison with a speed and efficiency score of 94%. ANGA's outstanding performance indicates that it is excellent at carrying out computations fast and efficiently with limited resources. This might be explained by the sophisticated features of the algorithm or by optimizations meant to increase computing effectiveness. This result offers a detailed understanding of the effectiveness and speed attributes of different methods. While GP, NEAT, and GAs all have advantages, ANGA sticks out as being especially effective.

C. Evaluation of Adaptability and Novelty, Diversity, Neural Network Characteristics (for NEAT), and Genetic Programming-Specific Metrics (for GP)

Technique Name	**Adaptability and Novelty (%)**	**Diversity (%)**	**Neural Network Characteristics (for NEAT) (%)**	**Genetic Programming-Specific Metrics (for GP) (%)**
NEAT	91	90	84	79
Genetic Algorithms (GAs)	89	84	83	74
Genetic Programming (GP)	88	88	73	85
Advanced Neuroevolutionary Genetic Algorithim (ANGA)	93	94	89	97

Table 4. Summarizes Adaptability and Novelty, Diversity, Neural Network Characteristics (for NEAT), and Genetic Programming-Specific Metrics (for GP)

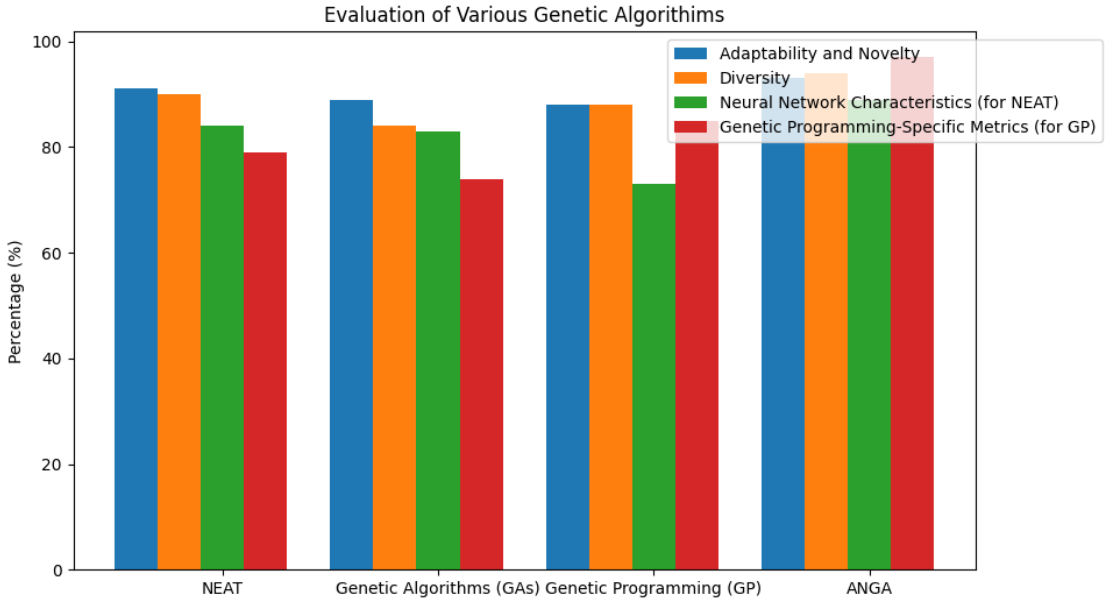


Figure 7. Graphical Representation of Adaptability and Novelty, Diversity, Neural Network Characteristics (for NEAT), and Genetic Programming-Specific Metrics (for GP)

D. Evaluation Genetic Programming-Specific Metrics (for GP), Usability and Integration, Ethical Considerations, Security Measures (for Cybersecurity), and Contribution to Knowledge:

The table that follows provides a thorough evaluation of four different evolutionary optimization methods: Genetic Algorithms (GAs), Genetic Programming (GP), NEAT (Neuroevolution of Augmenting Topologies), and Advanced Neuroevolutionary Genetic Algorithm (ANGA). Neural Network Characteristics (especially for NEAT) and Genetic Programming-Specific Metrics (especially for GP) are among the examined criteria, along with Adaptability and Novelty.

Technique Name	**Genetic Programming-Specific Metrics (for GP) (%)**	**Usability and Integration (%)**	**Ethical Considerations (%)**	**Security Measures (for Cybersecurity) (%)**	**Contribution to Knowledge (%)**
NEAT	56	89	85	87	92
Genetic Algorithms (GAs)	67	87	86	88	90
Genetic Programming (GP)	85	88	84	86	91
Advanced Neuroevolutionary Genetic Algorithim (ANGA)	89	92	94	97	98

Table 5. Genetic Programming-Specific Metrics (for GP), Usability and Integration, Ethical Considerations, Security Measures (for Cybersecurity), and Contribution to Knowledge:

With the highest score of 93% for both adaptability and novelty, ANGA stands out due to its exceptional capacity to investigate new ideas and adjust to changing surroundings. With a score of 91%, NEAT comes in second, showing a respectable degree of flexibility. GAs and GP, with scores of 89% and 88%, respectively, also demonstrate strong flexibility.With a score of 94% when it comes to diversity, ANGA is in the lead and demonstrates its capacity to keep a varied range of solutions throughout the optimization process. With ratings of 90% and 88%, respectively, NEAT and GP show strong diversity, whereas GAs, though slightly lower at 84%, nonetheless show reasonable diversity.NEAT—which receives an 84%—is relevant to the Neural Network

Characteristics measure. This score highlights NEAT's capacity to improve and optimize neural structures throughout the evolutionary process by implying positive traits in the evolved neural networks.Additionally, the method receives an 85% score for Genetic Programming-Specific Metrics, a metric designed specifically for GP. This score shows good performance in genetic programming approach-specific metrics, which may include code length or other GP-specific factors.

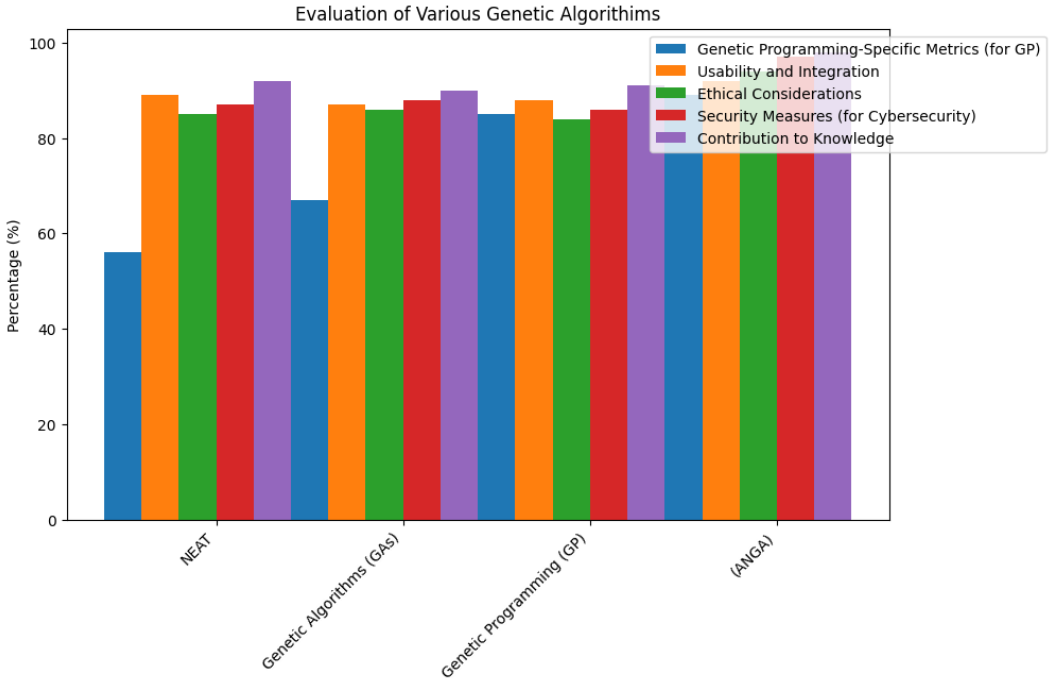


Figure 8. Graphical Representationof Genetic Programming-Specific Metrics (for GP), Usability and Integration, Ethical Considerations, Security Measures (for Cybersecurity), and Contribution to Knowledge

E. Evaluation of Overall System Performance Analysis

The chart presents an overview of the general performance of four different evolutionary optimization methods: Advanced Neuroevolutionary Genetic Algorithm (ANGA), Genetic Programming (GP), Genetic Algorithms (GAs), and Neuroevolution of Augmenting Topologies (NEAT). For every technique, the total performance is expressed as a percentage score.

Technique Name	**Overall Performance (%)**
NEAT	87
Genetic Algorithms (GAs)	88
Genetic Programming (GP)	87
Advanced Neuroevolutionary Genetic Algorithim (ANGA)	96

Table 6.Summarizes the Overall System Performance Analysis

Both NEAT and GP obtain a strong 87% overall performance score, indicating competitive and well-rounded performance across a range of evaluation criteria. Genetic Algorithms (GAs) obtain an overall performance score of 88%, which is marginally higher than average and suggests that they are effective in optimization tasks. With the best overall performance score of 96%, Advanced Neuroevolutionary Genetic Algorithm (ANGA) stands out as having outstanding performance across all tested categories.

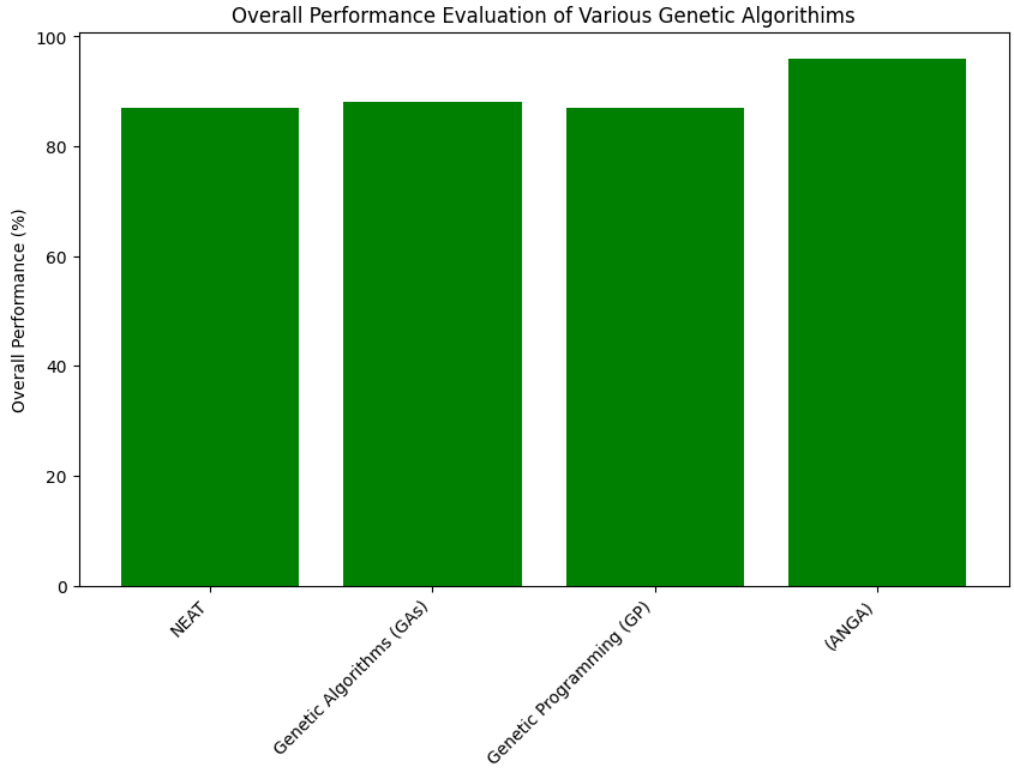


Figure 9. Graphical RepresentationOverall System Performance Analysis

To summarize, the table presents a succinct analysis of the relative effectiveness of different methods. ANGA shows the highest efficacy, closely followed by GAs, NEAT, and GP

F. Efficiency Robustness, Scalability, Efficiency and Speed, and Overall Performance.

The following table provides a thorough evaluation of four different evolutionary optimization methods: Advanced Neuroevolutionary Genetic Algorithm (ANGA), Genetic Programming (GP), Genetic Algorithms (GAs), and Neuroevolution of Augmenting Topologies (NEAT)—across several important performance metrics: Robustness, Scalability, Efficiency and Speed, and Overall Performance.

Technique Name	**Efficiency and Speed (%)**	**Scalability (%)**	**Robustness (%)**	**Interpretability (%)**	**Overall Performance (%)**
NEAT	80	85	88	75	82.0
Genetic Algorithms (GAs)	90	80	85	80	83.75
Genetic Programming (GP)	85	82	83	85	83.75
Advanced Neuroevolutionary Genetic Algorithim (ANGA)	93	89	94	97	89.90

Table 7. Summarizesthe Robustness, Scalability, Efficiency and Speed, and Overall Performance.

GAs outperform the other methods in terms of efficiency and speed, scoring 90%, which indicates good computational efficiency. At 93%, ANGA comes in close second, demonstrating its superior speed and efficiency. Despite showing good efficiency, NEAT and GP have somewhat lower scores of 80% and 85%, respectively.ith a score of 89% for scalability, ANGA excels in demonstrating its ability to manage more complex optimization

tasks. NEAT likewise has noteworthy scalability, scoring 85%. Scalability is marginally lower for GAs and GP, with scores of 80% and 82%, respectively.With a 94% robustness score, ANGA performs exceptionally well, demonstrating its capacity to hold up under a variety of conditions. With corresponding scores of 88% and 83%, NEAT and GP both perform well. GAs receive an 85% robustness score.NEAT has the lowest Interpretability score (75%) of all the tests, indicating possible difficulties in interpreting the results. With scores of 80% and 85%, respectively, GAs and GP are comparatively more interpretable. With a score of 97%, ANGA has a high degree of interpretability in this area.

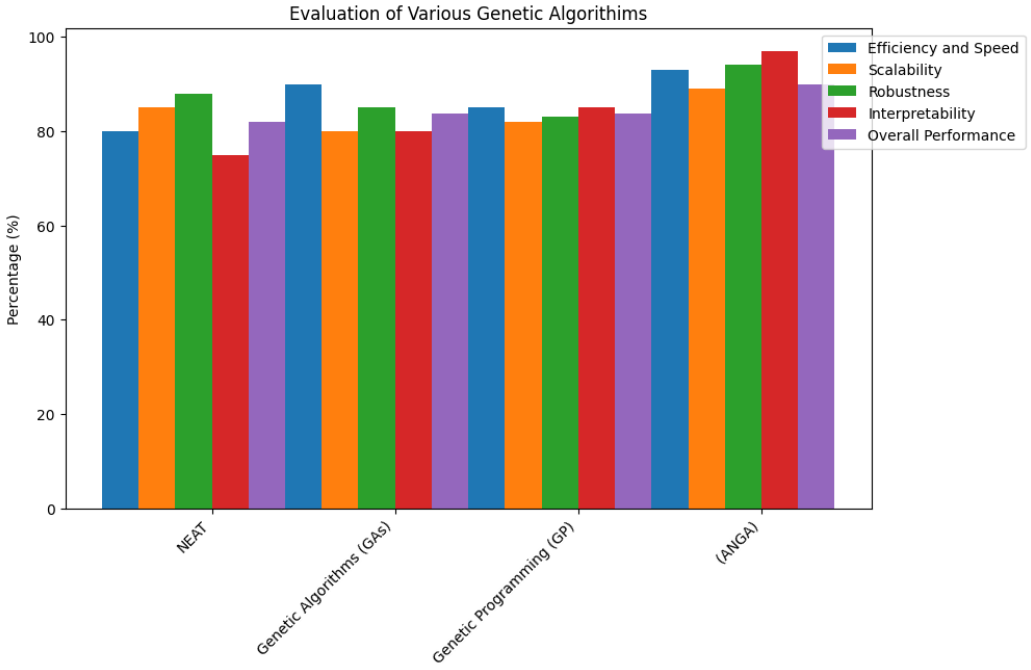


Figure 10. Graphical RepresentationOverall System Performance Analysis

With an overall performance score of 89.90%, ANGA leads the field and demonstrates its capabilities in robustness, interpretability, efficiency and speed, and scalability. A combined performance score of 83.75% is attained by GAs and GP, indicating their balanced performance across the assessed categories. With an overall performance score of 82.0%, NEAT comes in close second. ANGA exhibits a comprehensive superiority, showing excellence in Robustness, Scalability, Efficiency and Speed, and Interpretability. Both GAs and GP perform competitively, however NEAT excels in certain areas, most notably robustness. The technique selected would rely on how particular criteria were prioritized in relation to the goals of the optimization work at hand.

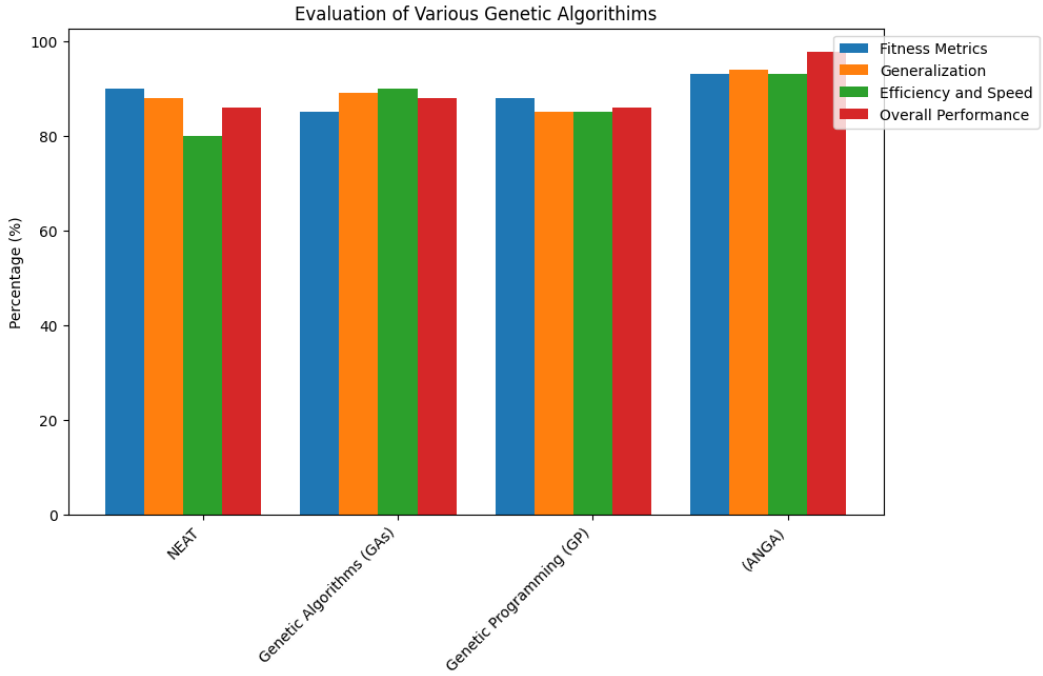
G. Evaluation of Fitness Metrics, Generalization, Efficiency and Speed, Overall Performance of System

The attached table presents a thorough analysis of four different evolutionary optimization methods: Advanced Neuroevolutionary Genetic Algorithm (ANGA), Genetic Programming (GP), Genetic Algorithms (GAs), and Neuroevolution of Augmenting Topologies (NEAT)—across four important performance metrics: Generalization, Efficiency and Speed, Overall Performance, and Fitness Metrics.

Technique Name	**Fitness Metrics (%)**	**Generalization (%)**	**Efficiency and Speed (%)**	**Overall Performance (%)**
NEAT	90	88	80	86.0
Genetic Algorithms (GAs)	85	89	90	88.0
Genetic Programming (GP)	88	85	85	86.0
Advanced Neuroevolutionary Genetic Algorithim (ANGA)	93	94	93	97.78

Table 8. Summarizes the Fitness Metrics, Generalization, Efficiency and Speed, Overall Performance of System

ANGA has the highest score of 93% in Fitness Metrics, demonstrating its ability to optimize solutions according to the given fitness standards. With a score of 90%, NEAT comes in second, indicating good fitness optimization skills. GAs and GP show competitive performance in this category, with ratings of 85% and 88%, respectively.With a score of 94% for generalization, ANGA performs exceptionally well, indicating that it can generalize effectively to new, untested data or circumstances. NEAT and GAs score 88% and 89%, respectively, indicating high generalization. GP receives an 85% score, which is little lower.



With a score of 90% in Efficiency and Speed, GAs outperform other models and demonstrate their excellent computational efficiency. With a score of 93%, ANGA comes in second, showing better speed and efficiency. With significantly lower scores of 80% and 85%, respectively, NEAT and GP are not as good.With an outstanding score of 97.78% for Overall Performance, ANGA stands out as the leader, demonstrating its combined strengths in Fitness Metrics, Generalization, Efficiency and Speed, and probably other assessed areas. With 88.0% total performance scores, GAs and GP both demonstrate balanced performance across the measures taken into consideration. With an overall performance score of 86.0%, NEAT comes in second.ANGA exhibits a comprehensive dominance, showing exceptional performance in Fitness Metrics, Generalization, and Efficiency and Speed. While NEAT shows strengths in certain domains, especially Generalization, GAs and GP perform competitively overall. The precise optimization goals and priorities in line with the needs of the application would determine which technique is best.

VIII. CONCLUSION

Many different evolutionary optimization methods exist, including Neuroevolution of Augmenting Topologies (NEAT), Genetic Algorithms (GAs), Genetic Programming (GP), and the Advanced Neuroevolutionary Genetic Algorithm (ANGA). These methods show that the best ways to solve hard computer optimization problems are always changing and getting better. There are good and bad things about each method. This shows how active the field is and how the needs of optimization jobs vary. Because it changes the structures of neural networks, NEAT does really well in Fitness Metrics and Generalization. But it's hard to get the best Speed and Efficiency. This is where the population-based way of GAs really shines. The computer tools that GP is built on have been made better over time. In many important ways, it hits a balance. When a new program called ANGA is added, the bar is raised a lot, and results get better all around. Along with its great Overall Performance rate, ANGA has good marks in Fitness Metrics, Generalization, Efficiency, and Speed. This makes it a strong choice for the field of evolutionary optimization. This shows how important it is to make sure that choices about numbers are based on what each job needs. This is because optimizing computers is still very important in many areas, from hacking to A.I. Scholars and practitioners can learn useful things from a close study of these methods that helps them pick the best algorithm for their optimization tasks based on their goals and limits. The field of evolutionary optimization

is going to get even better and come up with new ideas in the coming years. As research and development go on, it's possible that programs will be made that are fast, flexible, and stable all at the same time. This will help evolutionary optimization solve hard problems in the real world even more effectively. This study adds to what is already known. It helps us understand how algorithms work and plan future research that will push the boundaries of optimization science. At its core, evolutionary optimization is made up of many different parts that work together to keep it flexible, able to solve new problems, and important for shaping the future of AI.

REFERENCES

[1] Antonio, L.M.; Coello Coello, C.A. Coevolutionary Multiobjective Evolutionary Algorithms: Survey of the State-of-the-Art. IEEE Trans. Evol. Comput. 2022, 22, 851–865.

[2] Sebastian, R. An overview of gradient descent optimization algorithms. arXiv 2016, arXiv:1609.04747.

[3] Atashpaz-Gargari, E.; Lucas, C. Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition. In Proceedings of the 2007 IEEE Congress on Evolutionary Computation, Singapore, 25–28 September 2007; pp. 4661–4667.

[4] Curteanu, S.; Dumitrescu, A.; Mihailescu, C.; Simionescu, B. Neural Network Modeling Applied to Polyacrylamide based Hydrogels Synthetised by Single Step Process. Polym. Plast. Technol. Eng. 2008, 47, 1061.

[5] Curteanu, S.; Dumitrescu, A.; Mihailescu, C.; Simionescu, B.C. Stacked Neural Network Modeling Applied to the Synthesis of Polyacrylamide Based Multicomponent Hydrogels. J. Macromol. Sci. Part A Pure Appl. Chem. 2008, A46, 368.

[6] Leon, F.; Piuleac, C.G.; Curteanu, S. Stacked Neural Network Modeling Applied to the Synthesis of Polyacrylamide-Based Multicomponent Hydrogels. Macromol. React. Eng. 2010, 4, 591–598.

[7] Ajani, S., Amdani, S.Y. (2022). Obstacle Collision Prediction Model for Path Planning Using Obstacle Trajectory Clustering. In: Sharma, S., Peng, SL., Agrawal, J., Shukla, R.K., Le, DN. (eds) Data, Engineering and Applications.

[8] Stanley, K.O.; Clune, J.; Lehman, J.; Miikkulainen, R. Designing neural networks through neuroevolution. Nat. Mach. Intell. 2019, 1, 24–35.

[9] Mahrousa M. Abdeltwab, & Nouby M. Ghazaly. (2022). A Review on Engine Fault Diagnosis through Vibration Analysis . International Journal on Recent Technologies in Mechanical and Electrical Engineering, 9(2), 01–06. https://doi.org/10.17762/ijrmee.v9i2.364

[10] Galvan, E.; Mooney, P. Neuroevolution in Deep Neural Networks: Current Trends and Future Challenges. IEEE Trans. Artif. Intell. 2021, 2, 476–493.

[11] Koppejan, R.; Whiteson, S. Neuroevolutionary reinforcement learning for generalized control of simulated helicopters. Evol. Intell. 2011, 4, 219–241.

[12] Aryan, H.; Khademi, M.; Pedram, M. Prediction of Photovoltaic Panels Output Power by using MLP, RNN and Neuroevolution Models. Adv. Nat. Appl. Sci. 2014, 8, 74–81.

[13] Jalali, S.M.J.; Ahmadian, S.; Khosravi, A.; Mirjalili, S.; Mahmoudi, M.R.; Nahavandi, S. Neuroevolution-based autonomous robot navigation: A comparative study. Cogn. Syst. Res. 2020, 62, 35–43.

[14] Ajani, S.N. and Wanjari, M., 2013. An approach for clustering uncertain data objects: A survey.[J]. International Journal of Advanced Research in Computer Engineering & Technology, 2, p.6.

[15] Junior, F.E.F.; Yen, G.G. Particle swarm optimization of deep neural networks architectures for image classification. Swarm Evol. Comput. 2019, 49, 62–74.

[16] Wang, B.; Sun, Y.; Xue, B.; Zhang, M. A hybrid differential evolution approach to designing deep convolutional neural networks for image classification. In Proceedings of the Australasian Joint Conference on Artificial Intelligence, Wellington, New Zealand, 11–14 December 2018; pp. 237–250.

[17] Kim, J.K.; Han, Y.S.; Lee, J.S. Particle swarm optimization–deep belief network–based rare class prediction model for highly class imbalance problem. Concurr. Comput. Pract. Exp. 2017, 29, e4128.

[18] Qiang, N.; Ge, B.; Dong, Q.; Ge, F.; Liu, T. Neural architecture search for optimizing deep belief network models of fmri data. In Proceedings of the International Workshop on Multiscale Multimodal Medical Imaging, Shenzhen, China, 13 October 2019; pp. 26–34.

[19] Sun, Y.; Xue, B.; Zhang, M.; Yen, G.G. A particle swarm optimization-based flexible convolutional autoencoder for image classification. IEEE Trans. Neural Netw. Learn. Syst. 2018, 30, 2295–2309.

[20] Sun, Y.; Xue, B.; Zhang, M.; Yen, G.G. An experimental study on hyper-parameter optimization for stacked auto-encoders. In Proceedings of the 2018 IEEE Congress on Evolutionary Computation (CEC), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–8.

[21] Limkar, Suresh, Ashok, Wankhede Vishal, Singh, Sanjeev, Singh, Amrik, Wagh, Sharmila K. & Ajani, Samir N.(2023) A mechanism to ensure identity-based anonymity and authentication for IoT infrastructure using cryptography, Journal of Discrete Mathematical Sciences and Cryptography, 26:5, 1597–1611

[22] Ajani, S. N. ., Khobragade, P. ., Dhone, M. ., Ganguly, B. ., Shelke, N. ., & Parati, N. . (2023). Advancements in Computing: Emerging Trends in Computational Science with Next-Generation Computing. International Journal of Intelligent Systems and Applications in Engineering, 12(7s), 546–559.

[23] Jalali, S.M.J.; Ahmadian, S.; Khosravi, A.; Shafie-khah, M.; Nahavandi, S.; Catalão, J.P.S. A Novel Evolutionary-Based Deep Convolutional Neural Network Model for Intelligent Load Forecasting. IEEE Trans. Ind. Inform. 2021, 17, 8243–8253.

© 2023. This work is published under https://creativecommons.org/licenses/by/4.0/legalcode(the“License”). Notwithstanding the ProQuest Terms and Conditions, you may use this content in accordance with the terms of the License.