UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

Laboratory Exercise 2:

Train Neural Networks with Evolutionary Algorithms

Computational Intelligence

João Valério

Eirik Grytøyr

Date: 29/11/2022

Index

1. INTRODUCTION

The purpose of this assignment is to develop a method for training artificial neural networks using evolutionary algorithms, as an alternative to traditional backpropagation techniques.

This exercise will focus on using a Multi-Layer perceptron (MLP) network, with a single layer of hidden neurons and synthetic data used to control various aspects of the problem, including sample size, noise, and difficulty.

The goal is to optimise the performance of the Artificial Neural Network, taking into account factors such as the number of hidden neurons, regularization and the choice of the error function. The assignment will involve comparing the results of different evolutionary algorithms and regularization strategies and analyzing the trade-offs involved in each approach.

Finally, the project is divided into 5 parts. First, it is performed the description of the dataset selected for the classification problem, regarding its properties. The second part intends to optimize the MLP, through derivative methods strategies. The third unit intends to use Genetic algorithms, while the fifth part is relative to Evolution Strategies. Finally, the last one intends to compare these 3 distinct approaches based on execution time, the estimate of the true error and the size of the chosen model.

2. DATA DESCRIPTION

The California Housing dataset is a database of information on housing prices in California, including metrics such as the median house price, median income, and population. It was derived from the 1990 U.S. Census and includes information for different cities, towns, and neighbourhoods in the state. The dataset includes the following 13 features and a target variable, respectively:

- **CRIM:** per capita crime rate by town
- **ZN:** proportion of residential land zoned for lots over 25,000 sq. ft.
- **INDUS:** proportion of non-retail business acres per town
- **CHAS:** Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
- **NOX:** nitric oxides concentration (parts per 10 million)
- **RM:** average number of rooms per dwelling
- **AGE:** proportion of owner-occupied units built prior to 1940
- **DIS:** weighted distances to five Boston employment centres
- **RAD:** index of accessibility to radial highways
- **TAX:** full-value property-tax rate per $10,000
- **PTRATIO:** pupil-teacher ratio by town
- **B:** 1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town
- **LSTAT:** % lower status of the population
- **MEDV:** the target variable corresponds to the median value of a house in a city or town

Of the 13 features, 1 is categorical (CHAS), the remaining 12 are numerical, and the target variable, median_house_value, is a numerical variable that represents the median value of a house in a city or town. Furthermore, the dataset contains 506 instances, with no missing values.

Relatively to each feature the basic statistical descriptions are the following:

- **CRIM:** the values range from 0.00632 to 88.97620, with a mean of 3.61352 and a standard deviation of 8.601545.
- **ZN:** the values range from 0.00 to 100.00, with a mean of 11.36 and a standard deviation of 23.32.

- **INDUS:** the values range from 0.46 to 27.74, with a mean of 11.14 and a standard deviation of 6.86.
- **CHAS:** the values are either 0 or 1, with a mean of 0.06917 and a standard deviation of 0.253976.
- **NOX:** the values range from 0.3850 to 0.8710, with a mean of 0.5547 and a standard deviation of 0.115878.
- **RM:** the values range from 3.561 to 8.780, with a mean of 6.285 and a standard deviation of 0.702617.
- **AGE:** the values range from 2.90 to 100.00, with a mean of 68.57 and a standard deviation of 28.12.
- **DIS:** the values range from 1.130 to 12.127, with a mean of 3.795 and a standard deviation of 2.105710.
- **RAD:** the values range from 1.000 to 24.000, with a mean of 9.549 and a standard deviation of 8.707259.
- **TAX:** the values range from 187.0 to 711.0, with a mean of 408.2 and a standard deviation of 168.537.
- **PTRATIO:** the values range from 12.60 to 22.00, with a mean of 18.46 and a standard deviation of 2.164946.
- **B:** the values range from 0.32 to 396.90, with a mean of 356.67 and a standard deviation of 91.294863.
- **LSTAT:** the values range from 5.00 to 50.00, with a mean of 22.53 and a standard deviation of 9.197104.
- **MEDV:** the values range from 5.00 to 50.00, with a mean of 22.53 and a standard deviation of 9.197104.

Finally, the California Housing dataset is a regression dataset, with 13 features and one target variable.

3. DATA PREPROCESSING

In order to feed the three models that are going to be developed, it is essential to preprocess the data mentioned above. So, the subsequent steps are applied in the following order:

1) **Missing values**: Since there are no missing values, this step is not necessary.
2) **Split**: The split ratio is 80% for training, 10% for validation and 10% for the test set. It is also important to denote that the folds constitution is kept in the three models and cross-validation is not performed due to time constraints.
3) **Normalisation**: The method implemented is the min-max scaling, in which the values are scaled between 0 and 1.

Through this process, it is obtained the proper dataset to perform regression.

4. MODEL'S PROCEDURE

In the following three chapters, 5, 6 and 7 the MultiLayer Perceptron will be developed regarding different techniques. These are:

- **Chapter 5:** Backpropagation
- **Chapter 6:** Genetic Algorithms
- **Chapter 7:** Evolution Strategies

For each one, the loss function selected (MSE), the size (number of units) and the training time (in seconds) will be evaluated along with the defined parameters of the model. From that, it is pretended to conclude which one is the best model. However, the metric defined is simply the lowest MSE.

Continuing the previous reasoning, in chapter 8, the three models will be compared.

5. BACKPROPAGATION

In the first and most simple approach is intended to train, validate and test a MultiLayer Perceptron, as a regression model, through backpropagation, a derivative method.

From that perspective, the model implemented consists of an input layer, one hidden layer, and an output layer, in which, the loss function used to indicate how far the predictions are from the target values is the Mean Squared Error (MSE). Furthermore, the hidden and output layers' transfer functions are the logsig, since it is the most appropriate one for the study. Finally, the convergence is implemented by the Stochastic Gradient Descent (SGD), in which the parameters are the learning rate and the momentum.

Taking that into account, the MLP was optimized, through 42 combinations, accordingly with the following parameters:

Table 5.1 – MLP parameters.

Parameters		
Learning Rate	**Momentum**	**Layer Size**
0.01	0.7	5
0.1	0.8	10
-	0.9	20
-	-	30
-	-	40
-	-	50
-	-	60

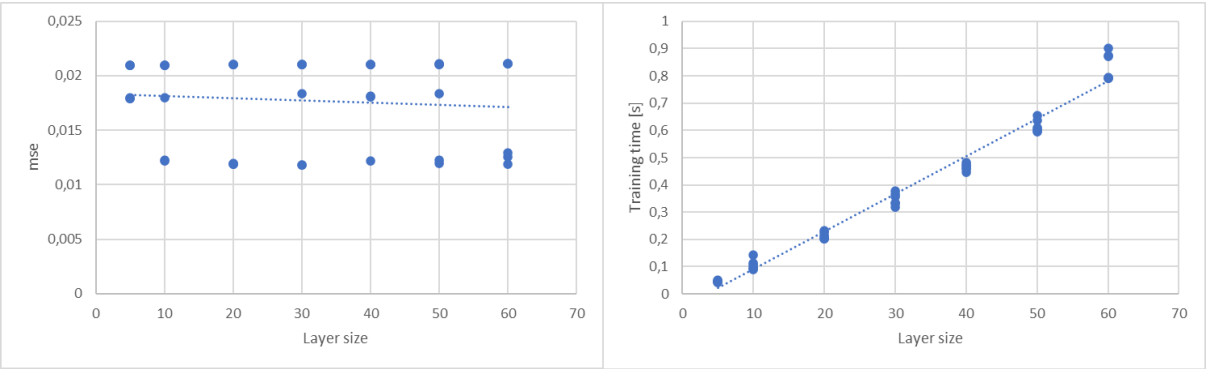
* The size refers to the number of units.

It is important to note that the results are optimized, in order that the comparison in the further chapters allows for a conclusion about the best approach.

From the previous configurations, the results obtained are exposed in the first table of Appendix A, in which the results are illustrated by the following plots 5.1, 5.2 and 5.3.

Plot 5.1 exposes the variation of MSE and training time along with the Layer size.

Figure 5.1 – MSE and training time with layer size variation.

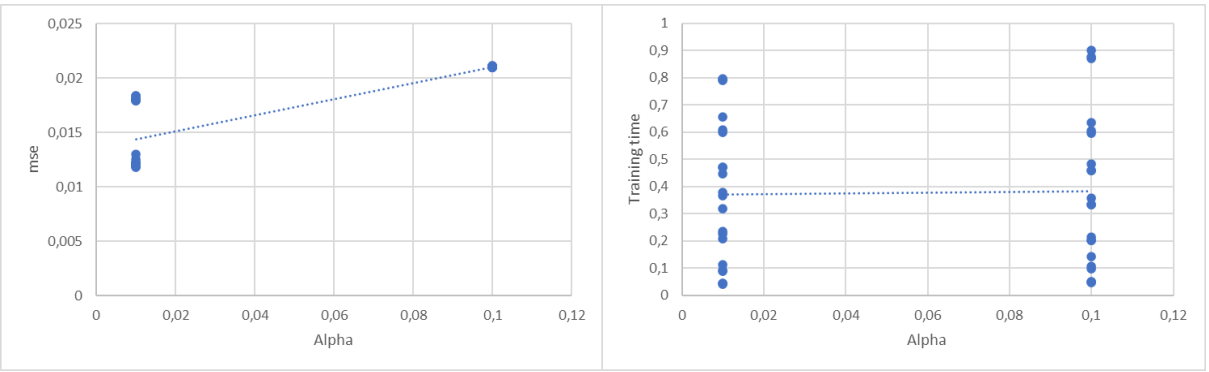


Firstly, it is seemingly visible that the training time increases with the number of units, while the MSE remains roughly constant. Precisely, in the MSE, the error within a single layer size can have a considerable variation, indicating that the learning rate and the momentum play an important role. Furthermore, the lowest value, corresponding to the most accurate model, is visually difficult to obtain. However, according to table 10.1 in appendix A the optimal number of units is 60, with a validation MSE of 0.006303. On the other hand, the lowest training time is achieved with 5 units, because there are fewer weights to be adjusted along the convergence.

As the metric to define the best model is through the lowest MSE, the optimal size is 60.

Plot 5.2 shows the influence of the learning rate on MSE and training time

Figure 5.2 – MSE and training time with learning rate variation.

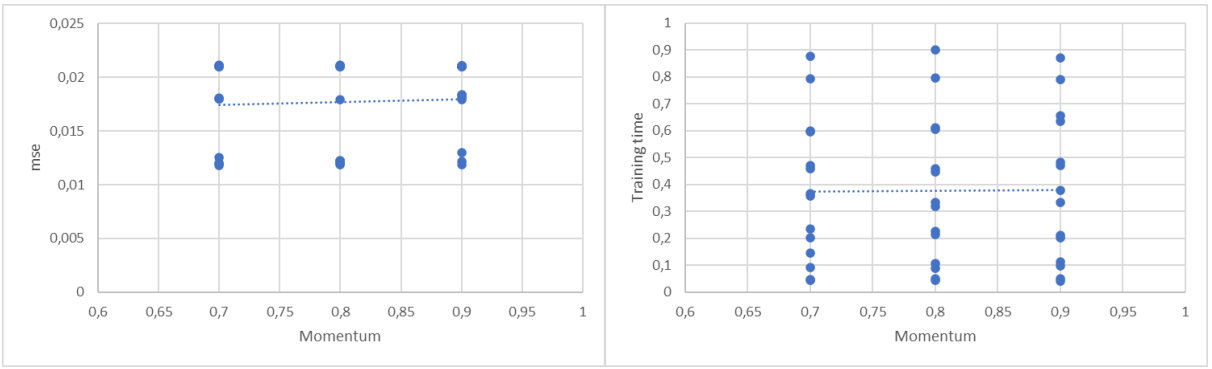


From the tested parameters, it can be inferred that, as expected, a learning rate of 0.01 is able to reach the most suitable results. In terms of time, there is no proper conclusion about the preferred configuration. This might be due to the minimality of the results, in which the time execution error produces a visible difference.

In general, the best value for the learning rate is 0.01.

Plot 5.3 displays the influence of the momentum on MSE and training time

Figure 5.3 – MSE and training time with momentum variation.



Finally, the last parameter seems to have a very low influence on the MSE and the training time. This is due to the fact that the momentum term is used in conjunction with Stochastic Gradient Descent (SGD), which already has a good convergence rate and can effectively minimise the MSE on its own. Furthermore, the beta parameter assumes a low value, indicating a low strength of the parameter.
So, even though it is not possible to reach a plausible solution visually, through the values exposed in appendix A 0.7 stands as the most accurate parameter.
Overall, the best configuration, the one with the lowest MSE, is in table 5.2.

Table 5.2 – MLP optimal parameters.

Optimal Parameters		
Learning Rate	**Momentum**	**Layer Size**
0.01	0.7	60

With which the training time and MSE in the training time are:

Table 5.3 – MSE and training time in MLP optimal parameters.

Evaluation Metrics	
Validation MSE	**Training Time**
0.006303	0.7926529

According to table 5.3, it is observed that MLP is able to reach accurate and efficient results. However, the size configuration is considerably complex (60 units), for the dataset selected.

6. GENETIC ALGORITHM

In chapter 6, instead of using derivative methods like in chapter 5, it is pretended to train a neural network using a Genetic Algorithm (GA). In this approach, a heuristic optimization method, inspired by the principles of natural evolution, is executed. The process involves iteratively generating and selecting successive generations of solutions (in this case, weights for the neural network) with the goal of finding the best solution. This process ends when 100 iterations are reached or there are no improvements after 10 consecutive generations.

From that perspective, the three main parameters selected are the hidden layer size, the population size and the mutation rate.

Table 6.1 – GA parameters.

Parameters		
Mutation Rate	**Population Size**	**Layer Size**
0.1	50	5
0.2	100	10
-	200	20
-	400	30
-	-	40
-	-	50
-	-	60

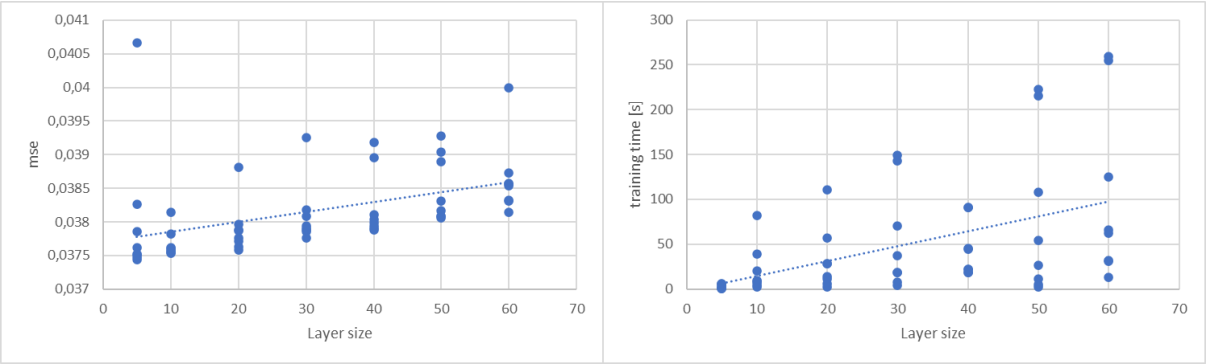
Precisely, the mutation rate and the population size are the ones different from the previous approach. These can be defined as

- **Mutation Rate:** a probability that determines the likelihood that a solution will undergo a mutation at each generation. The values selected for the purpose are 10% and 20%. According to the commonly used values, it can be said that 10% is low, indicating that the solutions may become too similar and the GA may get stuck in a local minimum. On the other hand, 20% can have a considerable influence on the solutions, in which the risk of the solutions becoming too dissimilar increases.
- **Population size:** which is the number of solutions that are included in each generation. So, the higher the value the bigger the diversity in the generation.

Taking these 3 parameters into account, figures 6.1, 6.2 and 6.3 are obtained.

Plot 6.1 exposes the variation of MSE and training time along with the Layer size.

Figure 6.1 – MSE and training time with layer size variation.

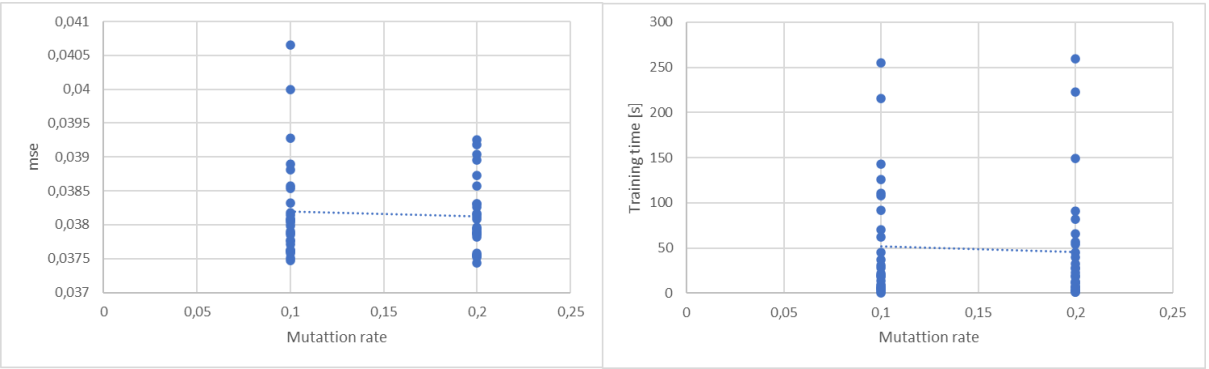


In the GA the layer size variation has an influence on the MSE and, considerably, on the training time. In the first metric, once again, for the same layer size, the MSE variation is relevant. For instance, 5 units can set the lowest and highest MSE value. This indicates the high influence and importance of the mutation rate and population size on the accuracy of the outcomes. Moreover, the layer size has a dramatic impact on the training time, especially when the population size is larger, as will be analysed.

Overall, a layer size of 5 units contributes to the most efficient and accurate model composition.

Plot 6.2 exposes the same metrics depending on the mutation rate.

Figure 6.2 – MSE and training time with mutation rate variation.

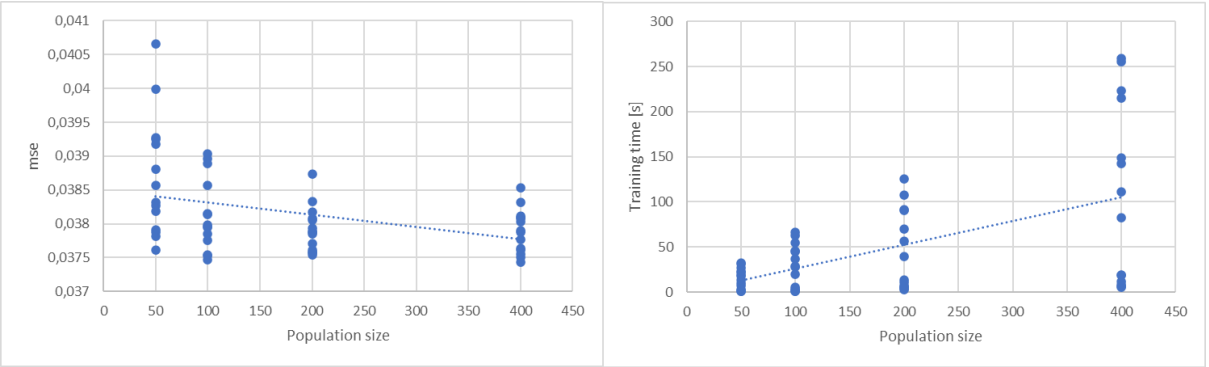


According to the previous values, in general, the mutation rate has not a considerable influence on the MSE and training time outcomes, because the difference between both considerations is only 10%. Thus, the discrepancy in MSE and efficiency is more relevant within the same mutation rate, where the layer and population sizes set their influence.

In fact, within the same mutation rate, the difference between the MSEs due to population and layer sizes is larger for lower mutation rates. However, this pattern might be specific to this dataset and not a conclusion that can be exploited for the overall problems.

The subsequent plot reveals the same metrics depending on the population size.

Figure 6.3 – MSE and training time with population size variation.



Finally, the last metric to analyse is the population size effect. So, according to the plots, while the MSE decreases with the increment of the population size, the training time follows an upward trend. Overall, at each generation, the GA evaluates the fitness of each solution and selects the ones that are most fit to be included in the next generation. So, the population size determines how many solutions are included in each generation and, therefore, how many solutions the GA is able to consider and explore at each step. This way, with more diversity, the solutions found might lead to better results, because the model has a lower chance of getting stuck in a suboptimal local minimum. On the other hand, this extensive research for the most suitable disposition leads to a less efficient solution.

Furthermore, it is observable that in lower population sizes, the discrepancy of the MSE values within the same vertical line is ampler, than for larger sizes. This is due to two main reasons:

1) First, lower population size may limit the GA's ability to explore different parts of the solution space, which can reduce its ability to find good solutions. This may be especially true if the solution space has several suboptimal solutions.
2) Second, lower population size may make the GA more sensitive to the values of other GA parameters, such as the mutation rate and the layer size. This is because a lower population size may reduce the amount of diversity in the population of solutions, which, once again, might lead to a suboptimal solution.

The same pattern occurs in the training time, however, the differences within the same population size are sounder for bigger values. This pattern is, particularly, influenced by the layer size. For instance, for the larger population size (400), the discrepancy in the influence between 5 and 60 units is larger than when the population size is 50. In other words, the model's efficiency depends heavily on the layer and population sizes.

Overall, the best configuration, the one with the lowest MSE, is in table 6.2.

Table 6.2 – GA optimal parameters.

Optimal Parameters		
Mutation Rate	**Population Size**	**Layer Size**
0.1	100	5

According to the values obtained, it is comprehended that the most accurate solutions were not achieved for the highest populations and layer sizes.

Firstly, it can be inferred that the highest population size does not lead necessarily to the most accurate values, because a lower value might lead to a more focused search and a higher likelihood of finding a high-quality solution. This way, it is possible to observe the importance of finding the balance between the lack of diversity, considering only a few solutions, and the excess, where unnecessary suboptimal solutions are assessed, without any positive gain.

Additionally, the layer size determines the model's complexity. So, this has to be in balance with the feature space complexity. Nonetheless, as the GA's population size is able to evaluate the fitness of each solution and select the ones that are most fit to be included in the next generation, the complexity of the model (learning capacity) needed can be reduced or it might lead to some overfitting.

The training time and MSE in the training time for the referred values are:

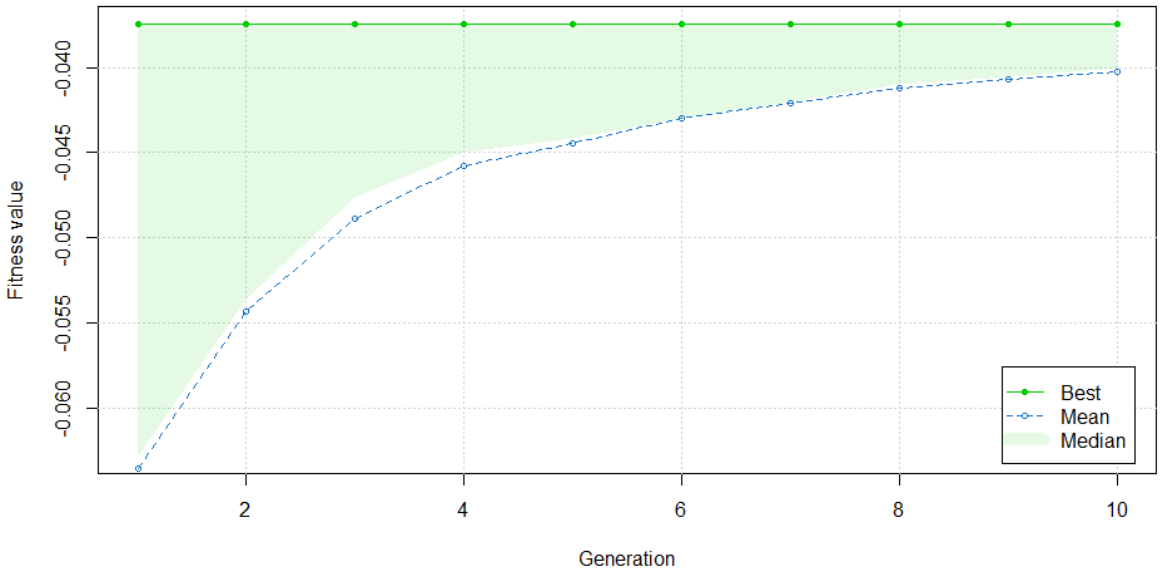
Table 6.3 – MSE and training time in GA optimal parameters.

Evaluation Metrics	
Validation MSE	**Training Time**
0.057286	1.342912

According to table 6.3, it is observed that GA does not reach as accurate and efficient results as the MLP. However, further comparisons will be accomplished in chapter 8.

The last analysis regarding the GA follows the Fitness function evolution along the generations.

Figure 6.4 – Fitness function evolution along the generations.



The evolution process of the graph indicates that a genetic algorithm involves an iterative improvement in the quality of the solutions of the population by selecting the fittest individuals, which the process continues for a specified number of generations.

So, the relationship between the fitness function and the generations is that the fitness function drives the evolution process by guiding the selection of individuals for reproduction and determining the likelihood that an individual will be included in the next generation. As the algorithm progresses through successive generations, the average fitness of the population improves or converges as

better solutions are selected and reproduced. This leads to the goal of finding a high-quality solution to the problem.

Finally, as the plot display, the best individual is found in the early generations without further improvement, while the average fitness value of the individuals gradually progresses. This indicates that there are a lot of local minima in the search space. This way the individual is trapped and is not able to escape during the first 10 generations. A way to improve this could be to increase the population size or experiment with other mutation or crossover rates.

7. EVOLUTION STRATEGY

The final implementation analysed regards the implementation of an evolution strategy for optimization in the training of a neural network. This approach, from the stochastic optimization algorithms family, relies on a population of candidate solutions (offspring) to search for the optimal solution. In each iteration, the algorithm generates a new population of offspring by perturbing the parameters of the current best solution with some random noise controlled by the step size. The offspring is then evaluated and the most suitable one is selected as the new current best solution. This process is repeated until the mean fitness value doesn't change more than a specified threshold or the number of iterations is 100.

From that perspective, the three main parameters selected are the hidden layer size, the offspring and the step size.

Table 6.1 – GA parameters.

Parameters		
Step Size	**Offspring Size**	**Layer Size**
0.01	50	5
0.02	100	10
-	200	20
-	400	30
-	-	40
-	-	50
-	-	60

Precisely, the step size and the offspring are the ones different from the backpropagations approach. These can be defined as

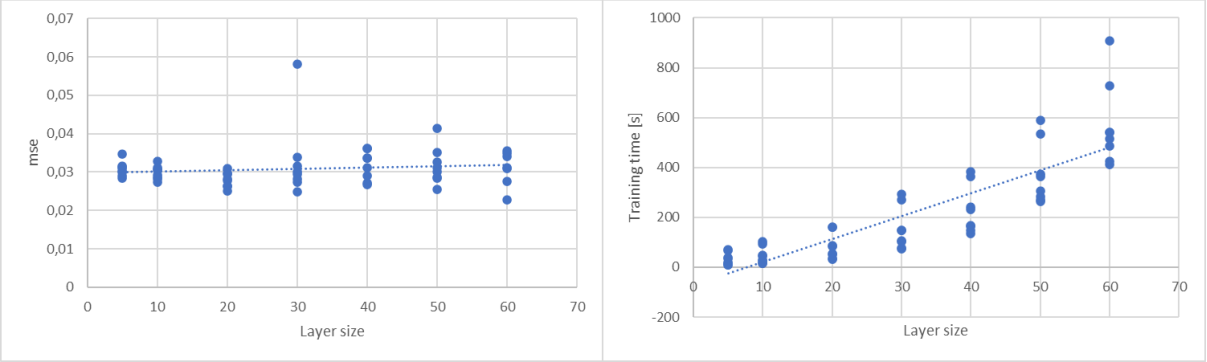
1) **Step Size:** it is the magnitude of the perturbation applied to the parameters of the current best solution (parent) to generate a new set of candidate solutions (offspring). The step sizes are small (0.01 and 0.02) to ensure that the offsprings are close to the parent and are likely to be similar in terms of performance.

2) **Offspring:** it is a set of candidate solutions generated by perturbing the parameters of the current best solution (parent) using some random noise. The number of offspring generated in each iteration is controlled by a hyperparameter offspring size.

Taking these 3 parameters into account, figures 7.1, 7.2 and 7.3 are obtained.

Plot 7.1 exposes the variation of MSE and training time along with the Layer size.

Figure 7.1 – MSE and training time with layer size variation.

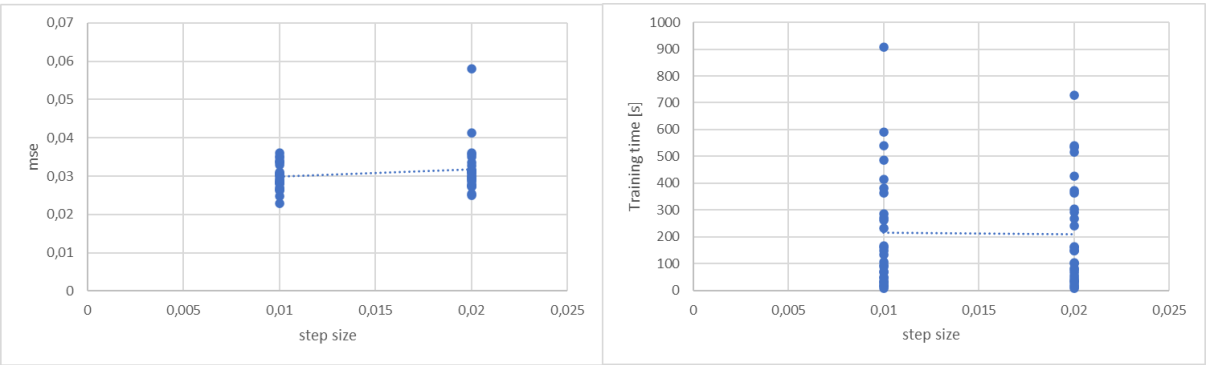


In the layer size graphical evolutions, it is inferred that the number of units does not allow for significant improvements in the accuracy of the model. Furthermore, contrasting what was seen in the GA, there is not a high variation inside each layer size, indicating that the step size and the offspring do not produce as much influence in the MSE, as the parameters of the GA. On the other hand, in terms of efficiency, the effect of the layer size in the training time is more significant than in the GA and backpropagation.

Overall, a layer size of 20 units contributes to the most accurate model composition. However, this selection comes with a high time cost (161,3366 sec.).

Plot 7.2 exposes the same metrics depending on step size.

Figure 7.2 – MSE and training time with step size variation.



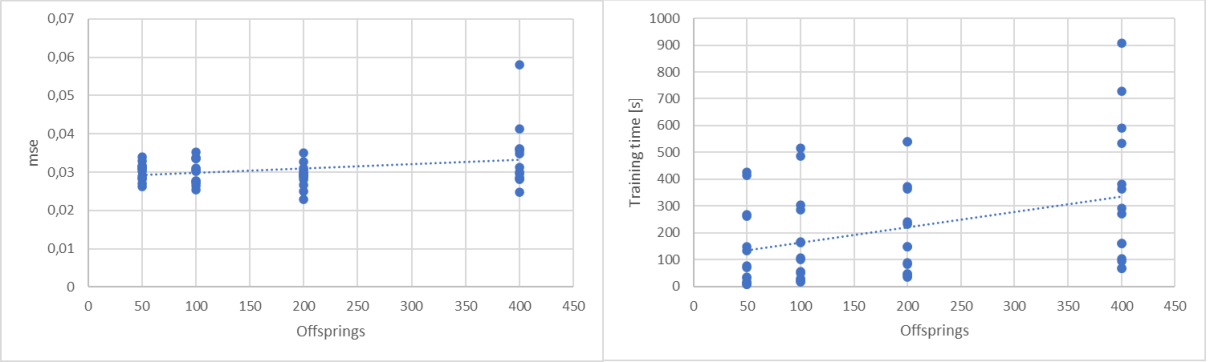
According to the previous values, in general, the step size has not a considerable influence on the MSE and training time outcomes.

Thus, the discrepancy in MSE and efficiency is more relevant within the same mutation rate, where the layer and the offspring set their influence. In fact, the trending lines indicate that, as expected, the model is slightly slower when the step size is 0.01, instead of 0.02, because the lower the value the greater the convergence time. In terms of accuracy, the step size affects the MSE by determining the magnitude of the perturbation applied to the parameters of the current best solution to generate new candidate solutions. As both values are close and low, the offsprings are similar to the parent.

Even though for this problem these are the optimal parameters, in other situations, low values may not allow the full needed exploration of the search space.

The subsequent plot reveals the same metrics depending on the offspring.

Figure 7.3 – MSE and training time with offspring variation.



Finally, the last metric to analyse is the offspring, which seems to be the one with the most increased influence on the MSE values. Compared to the GA techniques for the population size, this line tendency is seen with a contradictory tendency. However, this variation is not relevant. Finally, it is interesting to see that, as in the GA population size parameter, the influence of the remaining parameters is higher when the accuracy is worse and the offspring number is larger. This can be seen by the distance between the data points of the same offspring.

Moreover, in training time, the tendency is the same observed in GA's population size or backpropagation layer size. However, these values are considerably higher, leading to a less efficient approach to optimization.

Overall, the best configuration, the one with the lowest MSE, is in table 7.2.

Table 7.2 – ES optimal parameters.

Optimal Parameters		
Step Size	**Offspring**	**Layer Size**
0,02	400	20

According to the values obtained, it is comprehended that, compared to backpropagation, the layer size is also smaller, as seen in GA. However, in this turn, the number of units is 20, indicating that the model needs more learning capacity to reach the optimal solution than in GA.

Additionally, since the optimal value is 0.02, a lower step size does not necessarily mean an optimal solution. Therefore, the relationship between the step size and the MSE is not necessarily monotonic. This depends on the specific characteristics of the problem and the optimization landscape. For instance, in this problem, with lower step size, the offspring will not be able to escape from local minima or suboptimal solutions.

The training time and MSE in the training time for the referred values are:

Table 7.3 – MSE and training time in ES optimal parameters.

Evaluation Metrics	
Validation MSE	**Training Time**
0,043318	161,3366

According to table 7.3, it is registered an improvement in the validation MSE, with a high time cost in the training. However, further comparisons will be accomplished in chapter 8.

The last analysis regarding the ES follows the Fitness function evolution along the generations.

Figure 7.4 – Fitness function evolution along the generations.

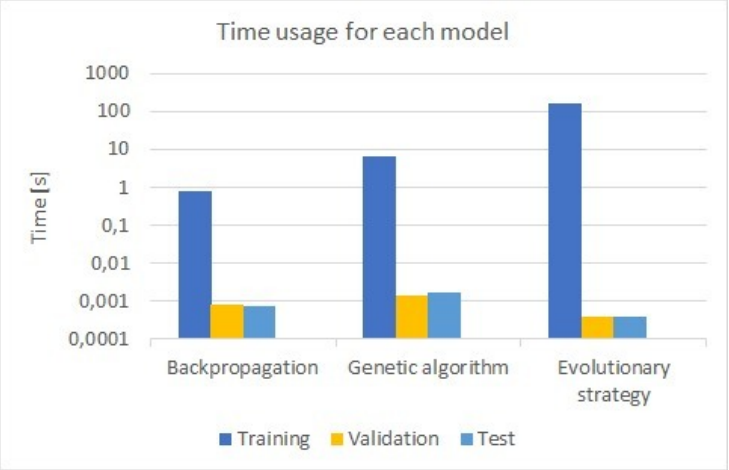
The evolution process of the graph indicates that, in order to reach the optimal solution, it is necessary to reach at least 100 iterations. Precisely, between the 5th and the 80th iteration, the models seem to not do any progress with the selections chosen. Thus, this period seems to be the bottleneck of the model's optimization, since outside this range the convergence is considerably efficient. Furthermore, comparing both downward tendency periods, it is observed that in the initial iterations the optimization is faster, than in the last ones. Periods not improving have been observed in many of the es models. It can seem like it gets stuck in local minima which can take multiple iterations to get out of.

Since the model stopped by exceeding 100 iterations, a bigger limit could lead to a better model. However, the time cost is significantly high for the purpose.

8. OVERALL COMPARISON

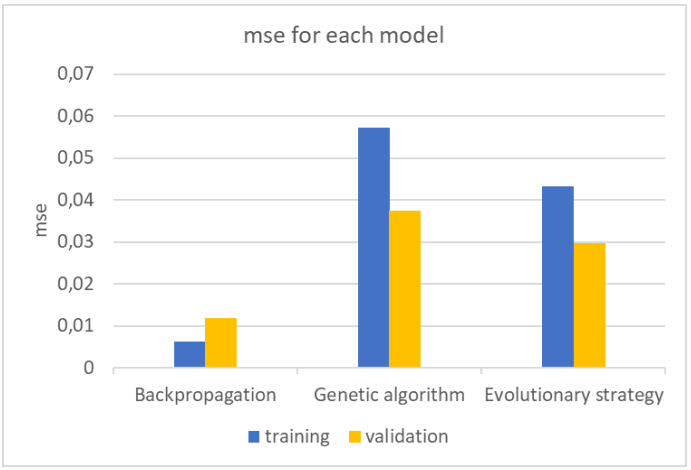
In order to compare the distinct models, plots 8.1, 8.2 and 8.3 represent the 3 main metrics that are pretended to evaluate the models in terms of time, MSE and size, respectively.

Figure 8.1 – BackPropagation, GA and ES training, validation and test time.



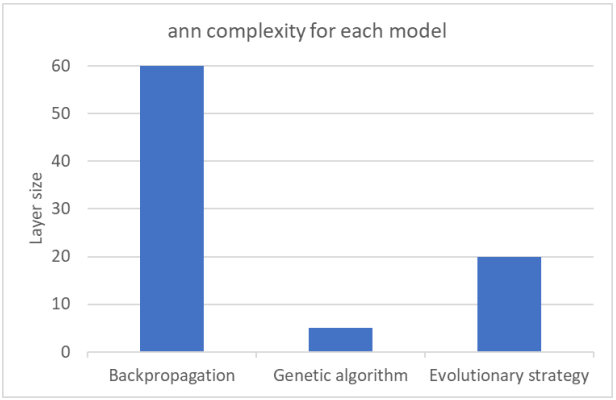
In terms of training time, the discrepancy between BackPropagation and the remaining is significant, particularly to ES. This occurs because, unlike GA, ES uses a continuous optimization approach, updating the model's parameters based on the gradient of the objective function. Nonetheless, when the prediction times (validation and test) are considered the ES approach seems to be the fastest, which can be justified by the decrement in the model's complexity, as shown in figure 8.3. Moreover, even though the GA has the lowest complexity among the three, as will be seen, it is the worst one in the prediction phase time.

Figure 8.2 – BackPropagation, GA and ES validation and test MSEs.



The discrepancies between the accuracy of the optimization techniques are considerable, but not extremely significant, notably in the test set. In fact, as said previously, a more refined adjustment on the parameters, especially for the ES could lead to a similar MSE of the backpropagation.
Nonetheless, regarding the accuracy, backpropagation seems to be the most suitable technique for this problem.

Figure 8.3 – BackPropagation, GA and ES layer size.



Finally, the layer's size is much bigger in a backpropagation approach, because this corresponds directly to the complexity of the problem or dataset. In the remaining strategies, the inclusion of populations or offspring, in which the best solutions are iteratively chosen allows for a lower need for learning capacity. Therefore, the resulting model is simpler and has fewer memory restrictions. Furthermore, in the validation or test sets the number of operations to perform is lower, resulting in a quicker model at these stages, as seen previously.

Thus, overall, the best optimization technique for the current problem is backpropagation.

To close the analysis procedure, the main succinct premises of each model are the following:

- **BackPropagation**:
  1) Faster training process than GA and ES
  2) Achieves the most complex model
  3) Sensitive to the choice of initial weights and may require careful tuning of hyperparameters
- **GA**:
  1) Lower number of iterations to converge than ES
  2) Achieves the lowest complexity in the final model
  3) The lowest complexity corresponds also to the most accurate solution (this is problem-specific)
  4) Can find solutions to complex and non-differentiable, which can be a problem for backpropagation.
- **ES**:
  1) A High number of iterations to converge
  2) Slower training process than backpropagation and GA
  3) Fastest model in the prediction process
  4) Useful when the objective function is noisy or the gradients are difficult to compute

9. CONCLUSION

In conclusion, three different approaches were taken to optimize the training of an artificial neural network for a classification problem. The first approach was using backpropagation, the second approach was using a genetic algorithm, and the third approach was using an evolution strategy, also an evolutionary method.

Overall, it can be concluded that both the genetic algorithm and evolution strategy approaches were able to produce good results in terms of accuracy, with the evolutionary approach being slightly more accurate but significantly slower than the genetic algorithmic approach. However, in this particular

case, the backpropagation approach was able to produce the most accurate results and was also the fastest. The choice of which approach to use will depend on the specific needs and constraints of the problem at hand, such as the metrics used, complexity and differentiability of the search space.

10. APPENDIX A

Table 10.1 – Backpropagation results.

Layer size	alpha	momentum	training time	validation time	test time	mse_valid	mse_test
5	0.01	0.7	0.043896	0.00036	0.000304	0.006669	0,017973
5	0.01	0.8	0.04424	0.000295	0.000243	0.006693	0,017905
5	0.01	0.9	0.041351	0.00037	0.000314	0.006698	0,017928
5	0.1	0.7	0.045488	0.000358	0.000287	0.00723	0,020933
5	0.1	0.8	0.050528	0.000362	0.000316	0.007227	0,020938
5	0.1	0.9	0.049231	0.000584	0.000487	0.007227	0,020938
10	0.01	0.7	0.092463	0.000371	0.000322	0.006676	0,018015
10	0.01	0.8	0.088632	0.000588	0.000576	0.006704	0,012245
10	0.01	0.9	0.112479	0.000881	0.000858	0.00691	0,012167
10	0.1	0.7	0.14385	0.000415	0.000405	0.007112	0,020988
10	0.1	0.8	0.107466	0.000317	0.000283	0.007125	0,020984
10	0.1	0.9	0.097793	0.00074	0.000591	0.007116	0,02098
20	0.01	0.7	0.233619	0.000344	0.000428	0.006738	0,011924
20	0.01	0.8	0.227115	0.000607	0.000737	0.00685	0,011981
20	0.01	0.9	0.209475	0.000428	0.000443	0.006707	0,011878
20	0.1	0.7	0.203284	0.000411	0.000429	0.007056	0,021023
20	0.1	0.8	0.215423	0.000432	0.000438	0.007049	0,021028
20	0.1	0.9	0.201298	0.000454	0.000548	0.007054	0,021027
30	0.01	0.7	0.365289	0.000467	0.000426	0.006769	0,011815
30	0.01	0.8	0.319204	0.000436	0.000466	0.006759	0,011831
30	0.01	0.9	0.37877	0.00054	0.000633	0.006553	0,018339
30	0.1	0.7	0.356559	0.000505	0.000493	0.007024	0,02105
30	0.1	0.8	0.332313	0.000635	0.00072	0.007029	0,021048

30	0.1	0.9	0.333181	0.000426	0.000441	0.007029	0,021048
40	0.01	0.7	0.471006	0.000437	0.000527	0.006662	0,01809
40	0.01	0.8	0.446409	0.000539	0.000612	0.006598	0,012182
40	0.01	0.9	0.47186	0.000499	0.000528	0.006658	0,0181
40	0.1	0.7	0.458109	0.009413	0.000678	0.007016	0,021066
40	0.1	0.8	0.459926	0.000468	0.000503	0.007012	0,021071
40	0.1	0.9	0.482231	0.000432	0.000504	0.007015	0,021064
50	0.01	0.7	0.599098	0.000594	0.000742	0.006736	0,011998
50	0.01	0.8	0.609601	0.000437	0.000505	0.006396	0,012261
50	0.01	0.9	0.655457	0.000442	0.000476	0.006539	0,018324
50	0.1	0.7	0.595778	0.000556	0.000591	0.007006	0,021070
50	0.1	0.8	0.605217	0.00049	0.000565	0.007005	0,021069
50	0.1	0.9	0.635561	0.000466	0.000535	0.007003	0,021077
60	0.01	0.7	0.792652	0.000769	0.000725	0.006303	0,012565
60	0.01	0.8	0.795614	0.000529	0.000605	0.006743	0,011905
60	0.01	0.9	0.790319	0.000662	0.000760	0.006391	0,012947
60	0.1	0.7	0.875428	0.000638	0.000813	0.006998	0,021083
60	0.1	0.8	0.900343	0.000715	0.000855	0.006999	0,021081
60	0.1	0.9	0.871422	0.000526	0.000597	0.006997	0,021080

Table 10.2 – Genetic Algorithm results.

Layer size	**population size**	**mutation rate**	**training time**	**validation time**	**testing time**	**generations**	**mse_valid**	**mse_test**
5	50	0.1	0.61125	0.00036	0.00035	10	0.06116	0.04065
5	50	0.2	0.79650	0.00044	0.00055	10	0.05826	0.03826
5	100	0.1	1.34291	0.0003	0.00062	10	0.05728	0.03747
5	100	0.2	1.33844	0.00032	0.00032	10	0.05731	0.03752
5	200	0.1	5.31322	0.00036	0.00037	19	0.05743	0.03761

5	200	0.2	2.92369	0.00039	0.00035	10	0.05772	0.03785
5	400	0.1	5.61416	0.00068	0.00076	10	0.05734	0.03750
5	400	0.2	6.26046	0.00037	0.00037	10	0.05720	0.03743
10	50	0.1	9.58975	0.0004	0.00046	100	0.05744	0.03760
10	50	0.2	6.81083	0.00048	0.00054	68	0.05775	0.03782
10	100	0.1	19.9807	0.00054	0.00075	100	0.05811	0.03814
10	100	0.2	2.00376	0.00043	0.00044	10	0.05743	0.03754
10	200	0.1	3.83844	0.00040	0.00042	10	0.05749	0.03757
10	200	0.2	39.4189	0.00046	0.00043	100	0.05736	0.03753
10	400	0.1	8.14046	0.00048	0.00047	10	0.05760	0.03761
10	400	0.2	82.2546	0.00050	0.00045	100	0.05737	0.03755
20	50	0.1	13.6287	0.00043	0.00050	100	0.05908	0.03880
20	50	0.2	1.87587	0.00044	0.00047	13	0.05799	0.03787
20	100	0.1	28.0575	0.00118	0.00062	100	0.05778	0.03775
20	100	0.2	28.3278	0.00047	0.00056	100	0.05803	0.03796
20	200	0.1	5.52863	0.00043	0.00050	10	0.05776	0.03771
20	200	0.2	56.9485	0.0007	0.00079	100	0.05750	0.03757
20	400	0.1	110.876	0.00056	0.00106	100	0.05763	0.03763
20	400	0.2	11.6072	0.00045	0.00049	10	0.05807	0.03787
30	50	0.1	18.6749	0.00051	0.00056	100	0.05843	0.03818
30	50	0.2	18.3303	0.00054	0.00067	100	0.05973	0.03925
30	100	0.1	36.9226	0.00051	0.00053	100	0.05804	0.03785
30	100	0.2	3.72198	0.00053	0.00067	10	0.05829	0.03794
30	200	0.1	70.1783	0.00048	0.00050	100	0.0583	0.03808
30	200	0.2	7.40644	0.00056	0.00052	10	0.05824	0.03788
30	400	0.1	142.935	0.00048	0.00054	100	0.05793	0.03776
30	400	0.2	149.167	0.00072	0.00079	100	0.05806	0.03790
40	50	0.1	22.1525	0.00067	0.00070	100	0.05824	0.03790

40	50	0.2	22.4362	0.00060	0.00069	100	0.05976	0.03918
40	100	0.1	44.7064	0.00052	0.00072	100	0.05835	0.03798
40	100	0.2	45.4174	0.00064	0.00078	100	0.05956	0.03895
40	200	0.1	91.3405	0.00058	0.00089	100	0.05818	0.03788
40	200	0.2	90.8099	0.00054	0.00090	100	0.05827	0.03793
40	400	0.1	18.7910	0.00057	0.00068	10	0.05851	0.03803
40	400	0.2	18.7451	0.00059	0.00057	10	0.05863	0.03811
50	50	0.1	2.69575	0.00069	0.00098	10	0.06033	0.03927
50	50	0.2	26.9195	0.00113	0.00120	100	0.05894	0.03831
50	100	0.1	5.25356	0.00062	0.00074	10	0.05970	0.03889
50	100	0.2	54.3730	0.00059	0.00065	100	0.05981	0.03903
50	200	0.1	107.896	0.00098	0.00076	100	0.05859	0.03805
50	200	0.2	10.8900	0.00059	0.00081	10	0.05882	0.03816
50	400	0.1	215.430	0.00068	0.00087	100	0.05861	0.03807
50	400	0.2	222.822	0.00057	0.00063	100	0.05861	0.03808
60	50	0.1	31.1139	0.00064	0.00076	100	0.06113	0.03998
60	50	0.2	32.1350	0.00073	0.00168	100	0.05943	0.03856
60	100	0.1	62.4778	0.00071	0.00100	100	0.05939	0.03857
60	100	0.2	66.1091	0.00056	0.00070	100	0.05881	0.03815
60	200	0.1	125.422	0.00082	0.00076	100	0.05902	0.03832
60	200	0.2	13.1685	0.00070	0.00076	10	0.05965	0.03873
60	400	0.1	255.215	0.00088	0.00093	100	0.05931	0.03853
60	400	0.2	259.402	0.00128	0.00137	100	0.05899	0.03831

Table 10.3 – Evolution Strategy results.

Layer size	**offsprings**	**step_size**	**training time**	**validation time**	**test time**	**mse_valid**	**mse_test**
5	50	0,02	8,487615	0,000689	0,000925	0,057039	0,031551

5	50	0,01	8,866993	0,000644	0,000901	0,055592	0,028358
5	100	0,02	18,80563	0,000624	0,000869	0,056053	0,030195
5	100	0,01	16,37045	0,00066	0,00094	0,057856	0,030854
5	200	0,02	36,93768	0,000764	0,001151	0,057342	0,030031
5	200	0,01	35,94596	0,000679	0,000932	0,058036	0,029069
5	400	0,02	69,23101	0,00133	0,002165	0,054564	0,03131
5	400	0,01	68,24075	0,000745	0,001084	0,057974	0,034723
10	50	0,02	15,37558	0,001482	0,001919	0,060423	0,031184
10	50	0,01	16,05714	0,000922	0,001306	0,056784	0,032843
10	100	0,02	28,68113	0,000884	0,001133	0,061776	0,027313
10	100	0,01	26,16552	0,000861	0,00139	0,056885	0,030526
10	200	0,02	45,72203	0,000952	0,001287	0,05076	0,029255
10	200	0,01	48,01608	0,000853	0,00123	0,051335	0,030644
10	400	0,02	103,4745	0,00081	0,001108	0,051974	0,028599
10	400	0,01	94,02857	0,000824	0,001136	0,052836	0,028177
20	50	0,02	33,81045	0,001676	0,00224	0,059252	0,03093
20	50	0,01	31,92075	0,001162	0,001475	0,055626	0,026247
20	100	0,02	55,78333	0,001866	0,002436	0,057956	0,027753
20	100	0,01	50,26874	0,001327	0,001991	0,056005	0,026371
20	200	0,02	83,90705	0,002255	0,002191	0,064341	0,025011
20	200	0,01	88,15794	0,001209	0,001475	0,059213	0,029432
20	400	0,02	161,3366	0,001315	0,00172	0,043318	0,029605
20	400	0,01	161,8226	0,001279	0,001903	0,045759	0,028241
30	50	0,02	77,78714	0,001487	0,001889	0,064684	0,031629
30	50	0,01	72,2408	0,001465	0,001961	0,055085	0,030137
30	100	0,02	101,1298	0,001577	0,00225	0,068146	0,027379
30	100	0,01	106,718	0,001497	0,001912	0,065	0,033806
30	200	0,02	147,0636	0,001551	0,002328	0,067745	0,029539

30	200	0,01	147,4354	0,001445	0,001834	0,059365	0,028117
30	400	0,02	291,4187	0,00209	0,002793	0,080684	0,05804
30	400	0,01	271,838	0,001541	0,001981	0,043925	0,024878
40	50	0,02	148,316	0,002494	0,003343	0,057145	0,029
40	50	0,01	134,5488	0,001749	0,002224	0,047469	0,02709
40	100	0,02	163,9221	0,001702	0,00213	0,06059	0,033665
40	100	0,01	167,1657	0,005004	0,00668	0,061566	0,033577
40	200	0,02	242,375	0,001674	0,002757	0,065485	0,031097
40	200	0,01	233,1855	0,001787	0,002364	0,05694	0,026741
40	400	0,02	364,321	0,001942	0,002448	0,348844	0,036059
40	400	0,01	382,5492	0,002018	0,002406	0,340793	0,036104
50	50	0,02	268,8996	0,002385	0,002956	0,061609	0,031112
50	50	0,01	262,5028	0,002119	0,002645	0,057461	0,028344
50	100	0,02	305,2765	0,002229	0,003086	0,060281	0,025451
50	100	0,01	284,5821	0,002122	0,002785	0,061744	0,035147
50	200	0,02	373,7222	0,002087	0,002587	0,069182	0,032666
50	200	0,01	364,4796	0,002197	0,002734	0,058423	0,028724
50	400	0,02	534,8464	0,002162	0,003153	0,053962	0,041384
50	400	0,01	589,5997	0,005414	0,008723	0,064638	0,030067
60	50	0,02	426,4624	0,002675	0,003759	0,05828	0,030991
60	50	0,01	413,4521	0,00307	0,004177	0,056316	0,034031
60	100	0,02	514,9841	0,002725	0,003409	0,065132	0,027687
60	100	0,01	487,1465	0,002594	0,003587	0,065033	0,031067
60	200	0,02	539,9343	0,002528	0,00313	0,062427	0,034962
60	200	0,01	540,0908	0,00261	0,003848	0,064783	0,022861
60	400	0,02	728,5393	0,00285	0,003531	0,068865	0,035616
60	400	0,01	908,4294	0,00368	0,003931	0,185215	0,196802