Machine Learning Project 3: Final Paper

Brett Layman

Kirby Overman

Carsen Ball

Abstract

In this paper, we compare the performance of three different evolutionary algorithms: Genetic Algorithm, Evolution Strategy, and Differential Evolution, along side Backpropagation in their ability to train an effective feed forward artificial neural network for classification. All three evolutionary algorithms used the mean test error against a test set as the fitness measure. We worked with five different datasets, all of which had numeric data with less than 5 classes and 5 or more attributes. From our experiments, we found that Differential Evolution generally outperformed the three other learning algorithms Across all datasets.

1. Introduction

This section provides a brief overview of the project.

1.1 Algorithms Used

Here we detail all of our algorithms. We do note at the introduction here, that the fitness function used for all of the following evolutionary algorithms is the mean error calculated by forward propagating a set of test examples through each Multi-Layer Perceptron (MLP).

1.1.1 Genetic Algorithm

For the genetic algorithm (GA), we implemented tournament selection with k set to half of the population, generational replacement, and crossover before mutation. The selection of crossover before mutation was arbitrary because the ordering doesn't matter for Genetic Algorithm. In our mutation step, we applied a Gaussian mutation using the standard deviation of fitness with respect to the entire population. The reason for this is so that the mutations aren't extreme and, for all children, are based on the mean quality of the population. The chromosome for the GA we define now to be simply the weights of a given MLP in the population.

1.1.2 $(\mu + \lambda)-$Evolution Strategy

The main differences between our evolution strategy (ES) algorithm and GA is the manner in which the sigma values of the Gaussian operator are updated. For ES, we update the sigma values by a Gaussian-like function

$$\sigma = \sigma * exp(\frac{U(0,\sigma)}{\sqrt{|C|}})$$

where $|C|$ is the length of the given chromosome.

1.1.3 Differential Evolution

Differential Evolution (DE) is a fairly simple and efficient algorithm that utilizes a unique version of mutation. DE iterates over the entire population, randomly selects three different individuals (not including the current individual) then subtracts two and adds it to the third. The resulting vector is known as a trial vector. Then for crossover, we iterate over each allele of the trial vector and construct a new individual, taking either the allele of the trial vector or the current iterated individual (depending on crossover rate). The result of this algorithm is a global optimizer for continuous spaces. The chromosomes for DE in our experiments are the weight matrices for a given MLP.

1.1.4 Backpropagation

Our backpropagation algorithm is similar to what we saw previously. The difference this time being that we are training an MLP for classification rather than function approximation. With that in mind, we needed to change the output layers to use a logistic output function rather than a weighted sum. Along with this change, we altered the output node delta calculation to include a calculation of the derivative of the logistic function, based on the chain rule.

1.2 Hypothesis

Here we present our hypotheses for our four training algorithms on our five datasets. In the rest of the paper we will address the quality as of certain algorithms to be 'typically' better than others. We define this notion that the given algorithm will be both expectantly and empirically better to a statistical significance threshold of $p = 0.05$.

1.2.1 Genetic Algorithm

We expect both GA and ES to outperform the other two algorithms. We believe that backpropagation will perform worse because it's complexity requires that it be run for a more iterations to achieve high performance. We think that DE will perform worse because we think it may be an overly simplistic approach that's limited to adding vectors in the current population.

1.2.2 $(\mu + \lambda)-$Evolution Strategy

Due to the adaptive sigma values for ES, we expect ES to perform better than the other three algorithms across all datasets, in part, for reasons stated above. The reason we believe ES to be better than GA is because its sigma values adapt based on the fitness of the individual. We think this will allow for higher diversity in early generations, and for worse individuals, thus giving the ES more ability to explore early on and optimize once a good solution is found. Whereas the GA will be limited to a constant rate of exploration.

1.2.3 Differential Evolution

We anticipate that Differential Evolution will be the fastest of the evolutionary algorithms due to its simplistic nature. Additionally, given that the DE mutations are simply linear

combinations of individuals in the population, we expect that diversity will be an issue and thus DE will be more susceptible to local optimums relative to the other three algorithms.

1.2.4 Backpropagation

Given our previous experiments with Backpropagation, we recognize how expensive backpropagation is and so we expect that this algorithm will be the slowest of all four algorithms to be compared. Additionally, we think that given the constraints of our experiment, it will not have enough iterations to outperform the other algorithms.

2. Methods

In this section we discuss the various methods utilized in training, testing and comparing our algorithms

2.1 Training and Test Set Generation

To create a valid spread of data we performed 10 fold cross validation on the data sets. We also randomized the data points, as to increase the spread of expected outputs throughout the subsets. We implemented a package from scikit-learn to perform the 10-fold cross validation. The operations were performed in the Data class, which allowed us to easily use the same subsets for all the algorithms.

2.2 Tuning Parameters

For the Genetic Algorithm (GA) and the Evolutionary Strategy (ES) we tunned crossover rate, population size and the number of generations that the algorithm ran for. Crossover rate corresponded to the likely hood that a a weight from one parent was was given to a child versus gaining a weight from a different parent. Population size corresponded to how many networks the algorithm was evolving with. The final parameter, number of generations is the number of times that the algorithm selects, crosses over, and mutates over the whole population. For computational efficiency we used a MLP with 2 hidden layers and 20 nodes per layer. GA and ES had a population of 20 individuals and ran for 5 generations. A crossover rate of .7 seemed to allow for the best exploration of the networks.

For Differential Evolution (DE) we considered the recommendation in [5] for tuning our parameters but found that it was only a good starting point. The paper recommended 0.47 mutation factor and 0.88 crossover rate. Instead, we found through experimentation that the closer to optimal parameters were 0.5 for both. Additionally, we worked with a population size of 20 just like GA and ES, but due to the speed of DE, we ran it up to 100 generations.

2.3 Algorithm Analysis

We evaluated the performance of the network using percent correctly classified. We tested each algorithm on 10 test subsets of data, and calculated percent correct for each subset. We then created an normal distribution for the percent correct. This allowed us to perform

a two sided t-test on the percent correct to determine if there was a significant difference in percent correctly classified, by each network from the various training algorithms.

3. Results

In this section we provide the results of our experiments on the 5 data sets, D1 - D5

3.1 Backpropagation

D1	75.0%	79.0%	87.0%	77.0%	81.0%	77.0%	73.0%	79.0%	86.0%	77.0%
D2	19.05%	38.1%	33.33%	42.86%	23.81%	38.1%	33.33%	28.57%	42.86%	33.33%
D3	44.16%	45.45%	50.65%	45.45%	42.86%	42.86%	44.16%	29.87%	55.26%	46.05%
D4	22.08%	15.58%	22.08%	24.68%	20.78%	9.09%	14.29%	19.48%	21.05%	18.42%
D5	6.0%	12.0%	13.0%	12.0%	8.0%	8.0%	9.0%	10.0%	8.0%	10.0%

Table 1: Percent Correctly Classified By the Backpropagation MLP for 10 Instances

3.2 Genetic Algorithm

D1	79.0%	79.0%	78.0%	78.0%	79.0%	79.0%	79.0%	79.0%	80.0%	81.0%
D2	71.43%	64.94%	57.14%	61.04%	68.83%	71.43%	62.34%	68.83%	64.47%	61.84%
D3	71.43%	66.23%	57.14%	59.74%	68.83%	71.43%	62.34%	70.13%	64.47%	60.53%
D4	12.99%	20.78%	23.38%	18.18%	11.69%	16.88%	24.68%	15.58%	21.05%	22.37%
D5	89.0%	93.0%	92.0%	95.0%	90.0%	91.0%	86.0%	85.0%	94.0%	89.0%

Table 2: Percent Correctly Classified By the GA MLP for 10 Instances

3.3 $(\mu + \lambda)-$Evolution Strategy

D1	78.55%	78.83%	77.93%	78.0%	79.38%	79.24%	79.1%	79.17%	80.48%	80.97%
D2	66.67%	61.9%	47.62%	76.19%	61.9%	80.95%	57.14%	71.43%	71.43%	71.43%
D3	71.43%	66.23%	57.14%	59.74%	68.83%	71.43%	62.34%	70.13%	64.47%	60.53%
D4	20.7%	20.5%	21.0%	20.7%	20.2%	21.0%	20.5%	21.5%	20.5%	20.2%
D5	94.0%	93.0%	95.0%	95.0%	91.0%	92.0%	93.0%	92.0%	94.0%	89.0%

Table 3: Percent Correctly Classified By the Evolution Strategy MLP for 10 Instances

3.4 Differential Evolution

D1	79.1%	78.6%	79.1%	80.3%	79.6%	79.4%	78.8%	80.3%	79.8%	79.4%
D2	33.9%	35.1%	36.3%	33.9%	33.3%	35.1%	34.5%	34.5%	36.9%	34.5%
D3	64.5%	64.3%	64.8%	64.3%	64.7%	66.8%	65.6%	64.6%	62.7%	64.3%
D4	19.9%	19.7%	19.7%	20.0%	19.4%	19.5%	20.2%	20.0%	19.5%	19.4%
D5	90.1%	90.6%	89.5%	90.0%	89.5%	89.3%	89.3%	89.1%	90.1%	88.8%

Table 4: Initial Percent Correctly Classified by Differential Evolution

D1	20.7%	20.5%	21.0%	20.7%	20.2%	21.0%	20.5%	21.5%	20.5%	20.2%
D2	66.3%	70.3%	68.2%	66.9%	65.0%	67.9%	67.4%	67.4%	67.3%	70.0%
D3	71.43%	66.23%	57.14%	59.74%	68.83%	71.43%	62.34%	70.13%	64.47%	60.53%
D4	12.99%	20.78%	23.38%	18.18%	11.69%	16.88%	24.68%	15.58%	21.05%	22.37%
D5	97.1%	97.1%	96.6%	97.4%	96.6%	97.3%	95.6%	97.1%	96.8%	96.8%

Table 5: Final Percent Correctly Classified By the Differential Evolution MLP for 10 Instances

3.5 Comparison

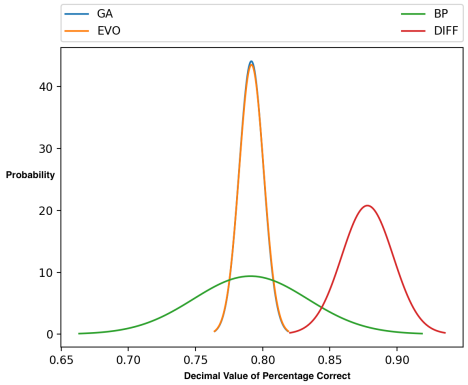

Figure 1: Normal Distribution of Percent Correct by Each Algorithm on D1

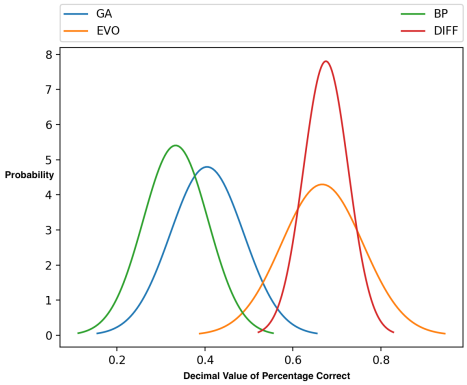

Figure 2: Normal Distribution of Percent Correct by Each Algorithm on D2

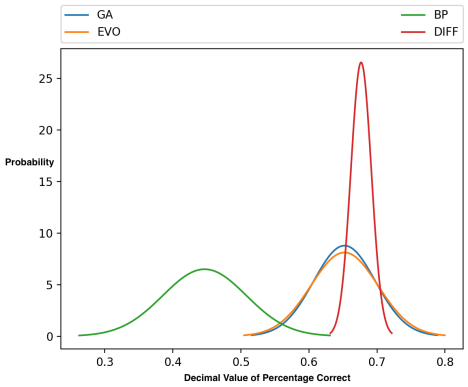

Figure 3: Normal Distribution of Percent Correct by Each Algorithm on D3

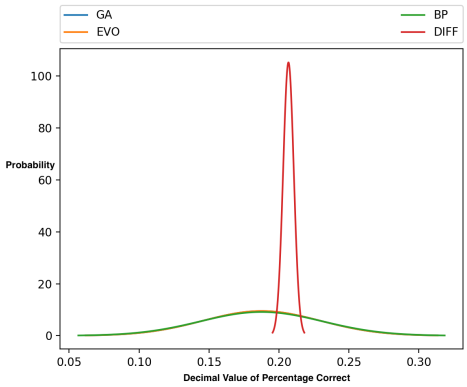

Figure 4: Normal Distribution of Percent Correct by Each Algorithm on D4

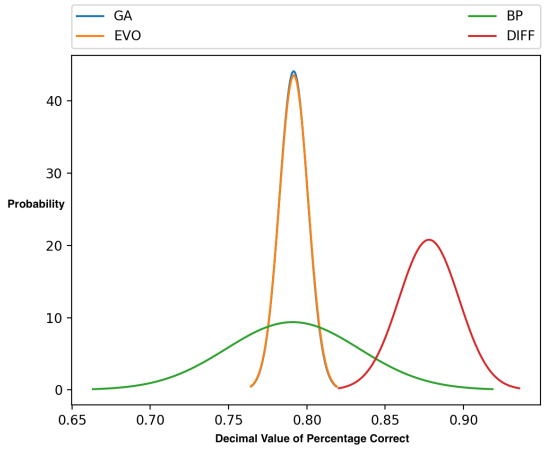

Figure 5: Normal Distribution of Percent Correct by Each Algorithm on D5

The information below is the results from the two sided paired t-test on the percent classified correctly for each algorithm for each data set. The format is True for a significantly different distribution, false for not, followed by the p-value of the statistical test. Figure 1,2,3,4,5 represent visualizations for the normal distributions of the percent correctly classified by each algorithm, on each data set.

3.5.1 D1 Statistics

Algorithm	GA	EvoStrat	DiffEvo	BackProp
GA	N/A	False, 0.98	True, 4.2e-07	False, 0.97
EvoStrat	False, 0.98	N/A	True, 4.2e-07	False, 0.96
DiffEvo	True, 4.2e-07	True, 4.2e-07	N/A	True, 4.3 e -0.7
BackProp	False, 0.97	True, 4.3 e -0.7	False, 0.96	N/A

Table 6: Two Sided T-Test Data Set 1

Algorithm	GA	EvoStrat	DiffEvo	Backprop
Mean	79.6%	79.16%	87.8%	79.1%
Variance	0.000082	0.000084	0.000369	0.001809

Table 7: Mean and Variance of the Percentage Correct for Each Algorithm

3.5.2 D2 Statistics

Algorithm	GA	EvoStrat	DiffEvo	BackProp
GA	N/A	True,9.42e-05	True, 1.07e-05	False, 0.07
EvoStrat	True,9.42e-05	N/A	False,0.81	True, 9.44e-06
DiffEvo	True, 1.07e-05	False,0.81	N/A	True, 4.3 e -0.7
BackProp	False, 0.07	True, 9.44e-06	True, 4.3 e -0.7	N/A

Table 8: Two Sided T-Test Data Set 2

Algorithm	GA	EvoStrat	DiffEvo	Backprop
Mean	40.46%	66.66%	67.5%	33.33%
Variance	0.0069	0.0086	0.0026	0.0054

Table 9: Mean and Variance of the Percentage Correct for Each Algorithm

3.5.3 D3 Statistics

Algorithm	GA	EvoStrat	DiffEvo	BackProp
GA	N/A	False, 0.99	False,0.14	True, 1.35e-05
EvoStrat	False, 0.99	N/A	False, 0.17	True, 1.71e-05
DiffEvo	False,0.14	False, 0.17	N/A	True, 1.11e06
BackProp	True, 1.35e-05	True, 1.71e-05	True, 1.11e06	N/A

Table 10: Two Sided T-Test Data Set 3

Algorithm	GA	EvoStrat	DiffEvo	Backprop
Mean	65.23%	65.23%	67.67%	44.68%
Variance	0.0021	0.0024	0.00023	0.0038

Table 11: Mean and Variance of the Percentage Correct for Each Algorithm

3.5.4 D4 Statistics

Algorithm	GA	EvoStrat	DiffEvo	BackProp
GA	N/A	False, 1.0	False, 0.18	False, 0.99
EvoStrat	False, 1.0	N/A	False, 0.18	False, 0.99
DiffEvo	False, 0.18	False, 0.18	N/A	False 0.20
BackProp	False, 0.99	False, 0.99	False 0.20	N/A

Table 12: Two Sided T-Test Data Set 4

Algorithm	GA	EvoStrat	DiffEvo	Backprop
Mean	18.76%	18.76%	20.68%	18.75%
Variance	0.0017	0.0017	0.000014	0.0019

Table 13: Mean and Variance of the Percentage Correct for Each Algorithm

3.5.5 D5 Statistics

Algorithm	GA	EvoStrat	DiffEvo	BackProp
GA	N/A	False, 0.063	True, 0.00011	True, 1.6e-13
EvoStrat	False, 0.063	N/A	True, 6.83e-05	True, 7.72e-15
DiffEvo	True, 0.00011	True, 6.83e-05	N/A	True,5.69e-16
BackProp	True, 1.6e-13	True, 7.72e-15	True,5.69e-16	N/A

Table 14: Two Sided T-Test Data Set 5

Algorithm	GA	EvoStrat	DiffEvo	Backprop
Mean	90.4%	92.8%	96.8%	96.00%
Variance	0.000964	0.00032	0.000024	0.00044

Table 15: Mean and Variance of the Percentage Correct for Each Algorithm

4. Discussion

In this section we discuss the results from our experiments and compare the performance of the algorithms in question.

4.1 Genetic Algorithm Performance

Both GA and ES didn't quite perform as expected. In fact, our GA performed worse than DE in datasets D1, D2, and D5. We contribute this to the fact the we only had five generations — a number we came to once we realized how complex both ES and GA were. Initially, we believed that backpropagation would have the longest running time, but we found experimentally that GA and ES took the most time to execute of the four algorithms. Further experimentation will be needed to determine if, given further generations, GA and ES will be able to perform as well as we'd initially hypothesized.

4.2 $(\mu + \lambda)-$Evolution Strategy Performance

We correctly hypothesized a comparison between GA and ES — ES would outperform GA, in other words, ES would produce a better MLP for classification than GA, although a statistically significant difference was only found in D2. In most cases the two algorithms performed comparably, which we attribute to their similarity. In D2 we think that ES may have been able to explore a space with a lower local minimum, because of it's ability to adapt sigma values for exploration.

4.3 Differential Evolution Performance

Differential evolution executed faster than all the other algorithms as expected; however, DE seemed to get caught in local optima. We believe this due to the lack of initial diversity

and the fact that DE's mutation method is simply taking linear combinations of current individuals, and thus less likely to introduce diversity that will help explore other areas of the solution space. This is empirically evident by the fact that, in trial runs, we noticed that DE very quickly reached a local optima then had very small changes — if any for the rest of the generations.

Some things that may have improved the final resulting MLP from differential evolution are greater population sizes, more generations, and initializing the weights at the start in such a way as to provide a much greater degree of diversity. This last part is due to the fact that the differential evolution began to converge within the 100 generations we used for testing. For our population size of 20, we were far under the recommended 75 from [5].

Regardless of these limitations, in our experiment DE was able to outperform all other datasets in D1 and D5.

4.4 BackPropagation Performance

Backpropagation performed worse that at least one of our evolutionary algorithms in all datasets except D4. Interestingly, it tended to have higher variance in it's results, leading us to believe that it was capable of achieving high performance, but that in some trials it was limited by the number of iterations, or possibly some of the tunable parameters. This is consistent with our hypothesis that Backpropagation requires more time and iterations to achieve optimal solutions than what we allotted for in our experiment. Although we cannot conclude, based on our limited results, that it would in fact become better. It's also possible that backpropagation had high variance because of getting stuck in local minima in some trials, and not it others. If this is the case, future experiments should explore using higher momentum parameters, since ours was fairly low.

4.5 Summary

Contrary to our expectations, differential evolution typically performed the best across our datasets. Even though we don't think DE explored as diverse of populations as the other two evolutionary algorithms, we think that it's speed allowed it to achieve better results given the time constraints of our experiment. It was able to iterate over many more generations that our other evolutionary algorithms, thus giving it a distinct advantage.

References

[1] Miroslav Kubat (2015) *An Introduction to Machine Learning* Springer. Chapter 5, Artificial Neural Networks.

[2] Dietterich, T (1997). *Approximate Statistical Test For Comparing Supervised Classification Learning Algorithms.* Oregon State University, Department of Computer Science

[3] UCI Machine Learning Repository (1987) archive.ics.uci.edu/ml/index.php

[4] Rudolf Kruse & Christian Borgelt & Christian Braune & Sanaz Mostaghim & Matthias Steinbrecher (2016) *Computational Intelligence* Springer. Second Edition.

[5] Magnus Erik Hvass Pedersen (2010). *Good Parameters for Differential Evolution* Hvass Laboratories

[6] Rainer Storn & Kenneth Price (1995). *Differential Evolution - A simple and efficient adaptive scheme for global optimization over continuous spaces* International Computer Science Institute.