An Exploration of Ensemble Models in Evolutionary Algorithms

David Earnest
College of Engineering and Applied Science
University of Cincinnati
Cincinnati, USA
earnesdm@gmail.com

***Abstract*—In this paper, we present a method for improving the performance of Evolutionary Algorithms. Our method works by treating the population of the EA as an ensemble model. We find that our method substantially outperforms our baseline models at almost no additional computational cost.**

***Index Terms*—Evolutionary Algorithms, Evolutionary Strategies, Ensembles, Genetic Algorithms**

I. Introduction

Ensemble models (which work by using multiple diverse models to predict an outcome) have shown great success in solving optimization problems by reducing overfitting, variance, and model uncertainty. In theory, we assume each model in the ensemble to be independent and unbiased, but in practice, we can often see benefits from less principled approaches that do not meet these assumptions. For example, Bootstrap aggregating works by fitting each model in the ensemble to a dataset $\mathcal{D}_i$ sampled uniformly with replacement from $\mathcal{D}$. With neural networks, it has been shown that we can see benefits by not even training the models from scratch and instead just retraining the last layer of the neural network.

In this paper, we examine ensemble-style approaches for Evolutionary Algorithms (EAs). While a theoretically principled approach to ensembles in EA would require training n different EAs and using one of many possible approaches to combine the best solutions from each different EA, this is costly. It requires training n different EAs. We notice that the population of a single EA already looks somewhat like an ensemble of models, where each member of the EA population represents its own model. Maybe we can treat the EA population itself as a type of ensemble model. If this works, it would be possible to train any EA, with no restrictions, and then afterward combine that population into an ensemble model improving performance. This could potentially improve performance at almost no cost and without enforcing any constraints on the EA. Another method would be to try using the population as an ensemble but promoting diversity in the population through methods like Fitness Sharing [1], Crowding [2], or Islands of Fitness [3]. (We know from experimental results that ensembles tend to work best when members of the ensemble are diverse and typical EAs populations tend to converge, so diversity-promoting methods could offer some benefits). These diversity-promoting methods should improve the ensembles without adding substantial computation costs or restrictions on the EA, by using the population of an EA as an ensemble.

We suspect that it is possible to see many of the benefits of using ensemble models in EAs – i.e., reducing overfitting, variance, and model uncertainty – without the extra computation of training n different EAs. Possibly we will be able to see improvements by just treating the population as an ensemble, but we might need to use additional diversity-preserving techniques to ensure our ensemble models are not too similar. Either way, if this works, then we would have a simple and low-cost way to improve the performance of EA models.

II. Background

A. Ensembles

Ensembles are a method for combining multiple diverse models in order to make a prediction. They have shown strong performance in reducing overfitting, variance, and model uncertainty. The intuition behind ensemble models is that many mediocre models can outperform an exceptional model assuming that the mediocre models are unbiased and diverse enough. One refrain (which doesn't provide the theory but is instructive nonetheless) is the story of statistician Francis Galton, who observed a 1906 county fair competition to guess the weight of an ox. Galton observed that the average guess (1,197 lbs.) was very close to the true weight of the ox (1,198 lbs.) [4]. (NPR repeated the experiment in 2015 and found similar results) [5].

There are many methods for implementing ensembles the most theoretically principled of which is the Bayes Optimal Classifier, i.e., the ensemble of all the hypotheses in the hypothesis space, but because of computational costs approximations are usually used. One such approximation is Bootstrap Aggregating (bagging) [6]. In bagging, each model is trained on a bootstrap dataset $\mathcal{D}_i$, created by sampling with replacement from the original dataset $\mathcal{D}_i$, and predictions are made with the ensemble of models using voting. Another common approximation to use is boosting [7]. In boosting, each model is trained on a dataset with each datapoint weighted proportional to how many previous models misclassified the datapoint. So, data points that are misclassified more often are assigned larger weights in the new dataset. Inference is done through voting.

There are many more approaches to ensemble models, but the above should provide sufficient background to understand this paper.

B. Evolutionary Algorithms

Evolutionary Algorithms (EAs) are a broad approach to solving optimization problems loosely based on biological evolution. While there are many different types of EAs, they all have the same structure. Namely, an initial population of individuals is created and then the following is repeated until a termination condition is met: parents are selected from the population, the parents are recombined to create offspring, the offspring are mutated, the quality of individuals is evaluated, and the best individuals are selected for the next generation.

The procedure is given in algorithm 1 [8].

Algorithm 1 Evolutionary Algorithm

```
INITIALIZE population with random candidate solutions
while TERMINATION CONDITION is false do
    SELECT parents;
    RECOMBINE pairs or parents;
    MUTATE the resulting offspring;
    EVALUATE new candidates;
    SELECT individuals for the next generation;
end while
```

Each of the steps in algorithm 1 above are intentionally vague, as there are many different choices for how they can be specified.

C. Evolutionary Strategies

Evolutionary Strategies (ES) are a particular class of EA. Corresponding to particular choices for representation, recombination, mutation, parent selection, and survivor selection.

1) Representation: In ESs, individuals in the population are represented as real-valued vectors. This is in contrast to other EAs like genetic algorithms (GAs) where individuals are represented as binary strings.

2) Recombination: ESs use either discrete or intermediate recombination. In discrete recombination, each allele of the offspring is chosen to be one of the two parent alleles randomly with equal probability for each parent. In intermediate recombination, the parent allele vectors are averaged.

3) Mutation: ESs use Gaussian perturbation to mutate population individuals. In this paper, we use a specific variant of Gaussian perturbation called uncorrelated mutation with n step sizes [9]. In this scheme, each individual $\langle x_1, \ldots, x_n \rangle$ is extended with n step sizes $\langle x_1, \ldots, x_n, \sigma_1, \ldots, \sigma_n \rangle$ and the mutation is specified as follows:

$$\sigma_i' = \sigma_i \times e^{\tau' \mathcal{N}(0,1) + \tau \mathcal{N}_i(0,1)} \tag{1}$$

$$x_i = x_i + \sigma_i' \times \mathcal{N}_i(0,1) \tag{2}$$

Where $\tau' \propto 1/\sqrt{2n}$, and $\tau \propto 1/\sqrt{2\sqrt{n}}$.

4) Parent Selection: ESs use uniform random parent selection so that each individual in the population has an equal probability of being selected.

5) Survivor Selection: ESs use deterministic elitist replacement either by (μ, λ) or $(\mu + \lambda)$ selection. Deterministic elitist replacement means that the best μ individuals are kept and the remaining are discarded. In $(\mu + \lambda)$ selection, the best μ individuals are chosen from both the parents and the offspring. In (μ, λ) selection, the best μ individuals are chosen to keep are picked just from the offspring; this means that parents are discarded after one generation.

D. Diversity Preservation in Evolutionary Strategies

Since ESs use a population-based approach they have the potential to evolve many quality solutions, but because of genetic drift and other factors, it is common for an ES's population to converge around one solution. Luckily, there has been much research examining approaches to preserve diversity in ESs. One such approach is Islands of Fitness.

Islands of Fitness works by dividing a population into sub-populations that evolve parallel, with some kind of communication structure between the sub-populations. For instance, the sub-populations can be arranged in a ring, where adjacent sub-populations can exchange members every n generations. The hope is that each sub-population will evolve different solutions. Achieving this hope requires careful population initialization as well as carefully regulating how many population members are exchanged. Too similar initialization or too much communication can result in all of the sub-populations evolving similar solutions.

E. Iris Dataset

The iris dataset is a dataset containing measurements from 150 different iris flowers. The dataset contains measurements corresponding to sepal length, sepal width, petal length, and petal width for each flower. There are 3 different classes of flowers in the dataset: Setosa, Versicolor, and Virginica. Each of the 3 classes of flowers has 50 data points.

III. Related Work

While there has much work on both ensembles and evolutionary algorithms over the years, as far as I know, there is no work looking at ensembles specifically applied to EAs or work looking at using the population of an EA as an ensemble of models.

IV. Experiments

A. Experiment 1

The first thing that I wanted to test was if I could get any improvement over a standard approach, by simply treating the population of the EA as an ensemble, but leaving everything else the same. The strength of this approach is that I would require no restrictions on how the EA was trained and require very little additional computation and no additional computation in training. To test this approach I evolved an ES to the iris dataset with the following hyperparameters:

Hyperparameter	Value	Hyperparameter	Value
μ	15	λ	100
τ	0.5	τ'	0.34
Mutation Rate	0.05	Recombination Rate	0.0
Mutation Type	uncorrelated mutation with n step sizes	Survivor Selection	(μ, λ)
		Recombination Type	Intermediate

TABLE I: EA Hyperparameters

To classify the iris dataset, split the dataset into 125 data points for evolving the ES and 25 data points for testing the ES. I was trying to use the ES to evolve a matrix W. So that if X is n by k matrix where n is the number of data points we are using to evolve the ES – 125 in this case – and k is the number of features in our dataset – 4 in this case: sepal width, sepal length, petal width, and petal length – and W is an k by c matrix, where c is the number of classes dataset. Then we can predict the classes y by calculating:

$$y = \underset{c}{\text{argmax}}(XW) \tag{3}$$

where the argmax is taken over the classes. Then y is a n dimensional vector containing the prediction for each data point. For the fitness function, I use classification accuracy.

Even though this model is a fairly simple linear model, it is powerful enough to evolve good classification accuracy on the training data. To demonstrate this we present Figure 1, showing the max and average fitness during the evolving process. We present these results averaged over 1000 runs to reduce variance.

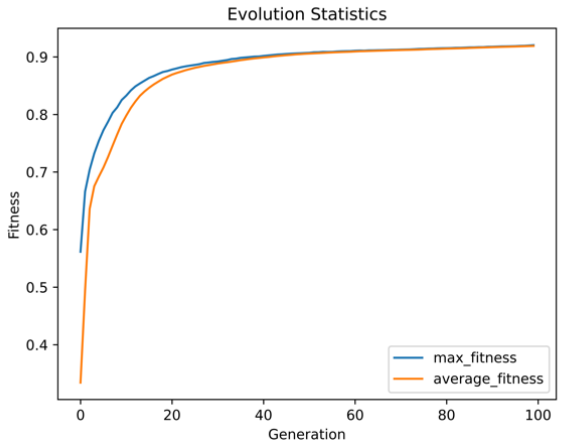

Fig. 1: max and average fitness during evolution process on iris dataset, averaged over 1000 runs

We can see that even though the model is a simple linear model, it is able to achieve good classification accuracy.

We ran this ES 1000 times for 100 generations each run. For each run, we compared the classification accuracy on the test data of the best-performing model vs an ensemble model, using the entire population of 15 for voting. We then calculated the performance on the test set. The average performance of the baseline model over 1000 runs was 0.886; the average performance of the ensemble model of 1000 runs was 0.884. We present the plot of the performances in each run in Figure 2.

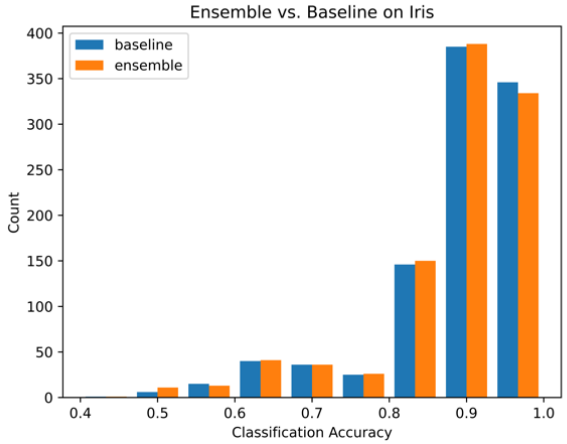

Fig. 2: classification accuracy of baseline vs. ensemble

We believe the reason that there wasn't much difference between the baseline model and the ensemble model is that the models in the ensemble were not sufficiently different. We suspect that one of the reasons for the lack of diversity is that problem we evolved a solution for is convex, and in many of the runs, it is likely that there is only one possible line that can be evolved to achieve the highest classification accuracy on the training data. To test this hypothesis, we will

look at evolving a classifier for only the first two classes for the iris dataset. Since the first two classes of the iris dataset are linearly separable, there will be an infinite number of lines that can be evolved by our ES with the same classification accuracy, i.e., with no selection pressure between them. This should increase diversity somewhat. To test this hypothesis we will evolve a model to only the first two classes of the iris dataset.

B. Experiment 2

For the second experiment, we trained to evolve a classifier to classify the first two classes in the iris dataset. This means that the dataset now only has 100 total data points. (Since the first two classes are linearly separable, it is fairly easy to achieve 100% classification accuracy on training data). We split the data into 75 training points and 25 testing points, and we ran the ES, with the same hyperparameters as in experiment 1, for 1000 runs. The average testing accuracy that we got for the baseline was 0.998 and the average testing accuracy for the ensemble was 0.999. While the actual increase from 0.998 to 0.999 is not that large, our ensemble approach decreases the number of misclassified points by half from 0.002 down to 0.001.

We present Figure 3 showing the classification accuracy on the test data over the 1000 runs.

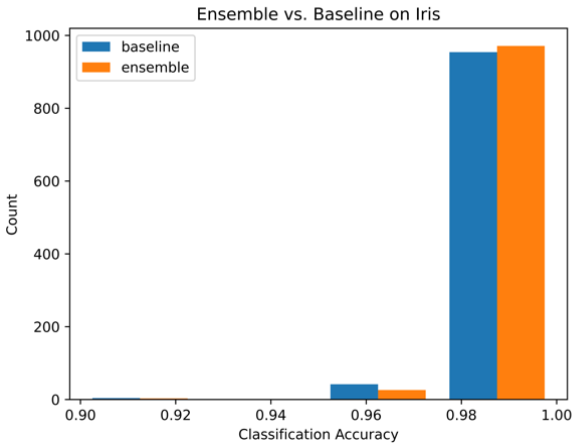

Fig. 3: classification accuracy of baseline vs. ensemble

In Figure 3, we can see the performance of the baseline and ensemble approaches. While both approaches have good classification accuracy, never dropping below 0.92, or 2 of 25 points misclassified, the ensemble approach does noticeably better, misclassifying half as many points. But since the values are fairly close together, we decided to do a one-sided t-test. The one-sided t-test returns a p-value of 0.03. For $\alpha = 0.05$, this means that we can reject the null hypothesis $baseline \geq ensemble$ for the alternative: $ensemble > baseline$.

C. Experiment 3

For the next experiment, we would like to use the same approach as experiment 2, but this time with noise and/or shifts added to the testing data. This would simulate distributional shift, i.e., a situation where the test data comes from a slightly different distribution than the training data. We believe that distributional shift could be interesting to test since the world is rarely static and distributional shift is widespread. For example, a model built to detect scam emails will likely experience distributional shift as scammers try to change their emails to fool the model. Also, we would expect to see more variation between the standard and ensemble approaches under distributional shift if the ensemble approach is learning a more robust classifier.

For this experiment, we run three-thousand runs on the iris dataset. The first 1000 runs will have noise added to the test data sampled from a normal distribution, the second 1000 runs will have a shift applied to the test data sampled from the normal distribution, and the final 1000 runs will have both noise and shift sample from a normal distribution added to the test data.

For the first 1000 runs, we added noise sampled from a Gaussian with a standard deviation of 0.5 to the test data and compared the standard and ensemble approaches. We saw a mean classification accuracy of 0.928 for the baseline approach and 0.939 for the ensemble approach, averaged over 1000 runs. We present a plot comparing classification accuracies in Figure 4. For a two-sided t-test, the p-value is 0.00026.

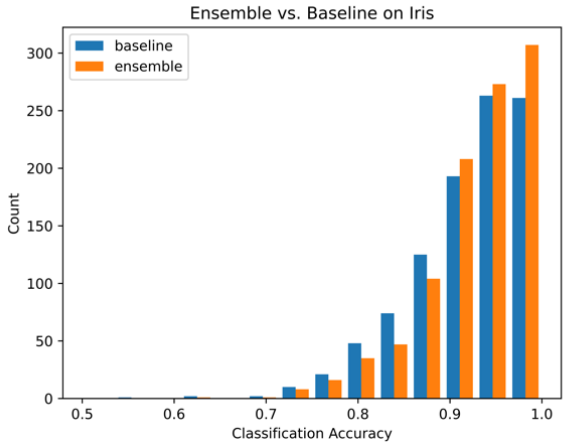

Fig. 4: classification accuracy of baseline vs. ensemble with noise

For the second 1000 runs, we added a shift sampled from a Gaussian with a standard deviation of 0.5 to the test data and compared the standard and ensemble approaches. We saw a mean classification accuracy of 0.930 for the baseline approach and a classification accuracy of 0.937 for the ensemble approach. For a two-sided t-test the p-value is 0.29. We present a plot comparing the classification accuracies in Figure 5.

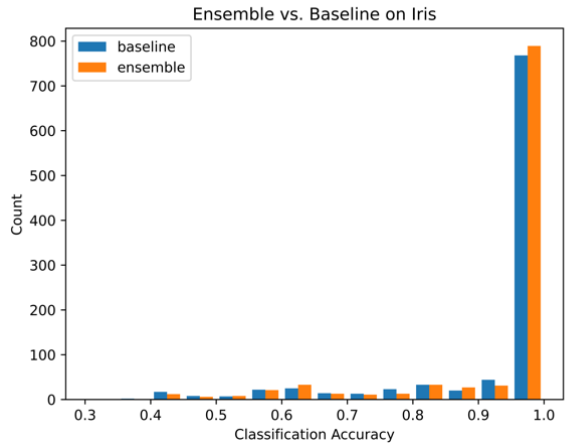

Fig. 5: classification accuracy of baseline vs. ensemble with shift

For the third 1000 runs, we added both shift and noise sampled from a Gaussian with a standard deviation of 0.25 and compared the standard and ensemble approaches. We saw a mean classification accuracy of 0.967 for the baseline approach and 0.973 for the ensemble approach. For a two-sided t-test the p-value is 0.032. We present a plot comparing the classification accuracies in Figure 6.

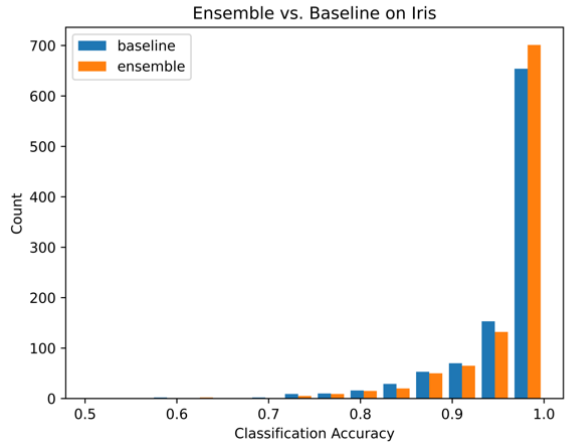

Fig. 6: classification accuracy of baseline vs. ensemble with shift and noise

Compared to the results from experiment 2, the results, from experiment 3, with distributional shift improved more in classification accuracy over the baseline but didn't decrease the percentage of misclassified points by the same amount. We believe that the reason ensembles didn't decrease the percentage of misclassified points by as much as under distributional shift (only $\sim 10\%$ under distributional shift compared to 50% without) is that since the first two iris classes fairly easily classified, a good classifier can classify all of the test points perfectly if unless we get a very unfortunate train-test split, but under distributional shift, there are many points that even a great classifier will misclassify.

Overall, the results have been promising, as we have seen the ensemble approach can achieve some improvement over the baseline with almost no additional computational overhead and no additional restrictions on the EA.

Going forward, we will see if it is possible to further improve the performance of our approach by using diversity-preserving techniques.

D. Experiment 4

Inspired by the increase in performance that we saw going from experiment 1 to experiments 2 and 3, we will try to further increase diversity. This time we will use a diversity-preserving technique to increase the diversity of the models in the ensemble. We hope that this will further improve performance.

For this experiment, we will use an Islands of Fitness approach over ES sub-populations. Each ES sub-population will have the same hyperparameters as those in Table I. We will evolve three different sub-populations arranged in a ring so that adjacent populations can exchange 1 population members every 25 generations. We initialize the sub-populations uniformly at random between non-overlapping intervals. The first sub-population is initialized between -30 and -10; the second sub-population is initialized between -10 and 10; the third sub-population is initialized between 10 and 30. We will compare the approaches to an Island of Fitness model baseline, using the same hyperparameters, but picking the best performing population individual instead of an ensemble. (Note: since each subpopulation is the same size as the population in experiments 1 through 3, we have three times as many members in the total population for the Islands of Fitness approach, because of this it is dangerous to directly compare results with the first three experiments since the computational cost to evolve these models is higher. We chose to increase the population size, and therefore the computational cost, because we felt if we keep the total population size at 15, then the subpopulations would be too small.)

We start by running this algorithm on the same problem as experiment 1. Namely, we evolve a classifier for the Iris dataset with 125 datapoints used for the training set and 25 datapoints used for the testing set. For this experiment, we don't have any distributional shift. The results of 1000 training runs are presented in the histogram in Figure 7.

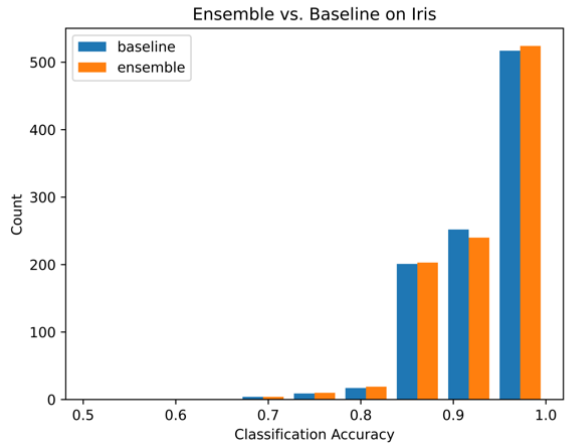

Fig. 7: classification accuracy of baseline vs. ensemble

The mean classification accuracy of the baseline model is 0.933 and the mean classification accuracy of the ensemble is 0.933. So, there does not appear to be an impact from using ensembles in this strictly convex problem. This matches what we observed in experiment 1. The new information that we gained is that diversity preserving techniques don't seem to help for strictly convex problems, as expected. Although the classification accuracies are higher in this experiment than in experiment 1, this is explained by the use of the more computationally expensive Islands of Fitness model.

In the next experiments, to make the problem not strictly convex we will replicate what we did in experiments 2 and 3 with the Islands of Fitness model. Namely, we will evolve a classification model over the first two classes of Iris. This makes the problem convex, but no longer strictly convex.

E. Experiment 5

For the first part of this experiment, we run the Islands of Fitness model with and without the ensemble approach to classify the first two classes of the Iris dataset. The results are presented in Figure 8.

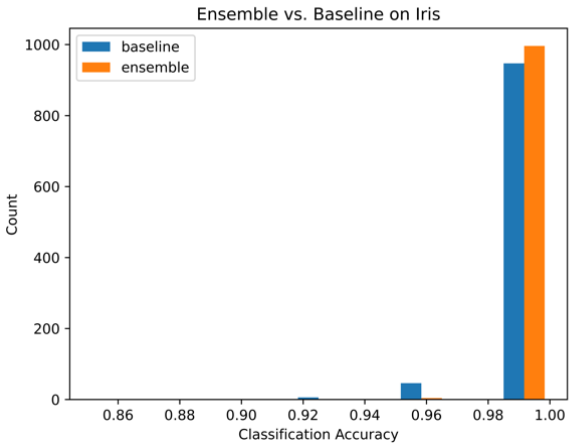

Fig. 8: classification accuracy of baseline vs. ensemble

The mean fitness for the ensemble model is 0.9998 and the mean fitness for the baseline model is 0.9976. And the p-value for a two-sided t-test is $1.94 * 10^{-10}$. So, even though the mean classification accuracies don't look too different from what we saw in experiment 2 (an increase of 0.001 for experiment 2 versus an increase of 0.0022 for experiment 5) this is a much stronger result as demonstrated by the p-value for the t-test. In this experiment, we were able to reduce the average percentage of misclassified points by 92% from 0.0024 down to 0.0002 by the use of the ensemble model. This result reinforces our belief that diversity-preserving techniques help improve our ensemble method.

F. Experiment 6

For the sixth experiment, we will mirror what we did in Experiment 3. Namely, we will repeat the results of Experiment 5, but this time adding distributional shift to the test data, through Gaussian noise and shift.

First, we add Gaussian shift to the test data sampled from a Gaussian with a standard deviation of 0.5. We ran 1000 different runs of our ES. The results can be seen in Figure 9. In this test, the mean classification accuracy for the ensemble model is 0.965 and the mean classification accuracy for the Islands of Fitness baseline is 0.932. When we run a two-sided t-test, we get a p-value of $2.05 * 10^{-36}$.

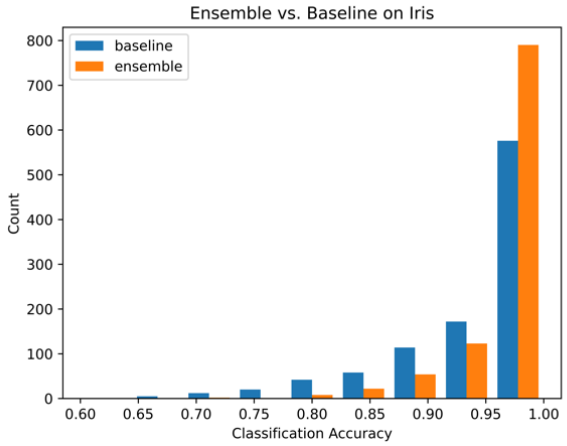

Fig. 9: classification accuracy of baseline vs. ensemble with noise

The results from this experiment mirror the experiment shown in Figure 4, where we had a mean classification accuracy of 0.928 for the non-Islands of Fitness baseline and 0.939 for the non-Islands of Fitness ensemble model. Notice that the non-Islands of Fitness ensemble model from 4 outperforms the baseline Islands of Fitness model despite using a population of one-third the size baseline Island of Fitness model and therefore substantially less compute. Additionally, we can see that the performance increase of using an ensemble is greatly improved by using it in addition to an Island of Fitness model.

Next, we will examine the results of adding a shift to the test data sampled from a Gaussian with a standard deviation

of 0.5. We again run 1000 runs of the algorithms. The results are in Figure 10.

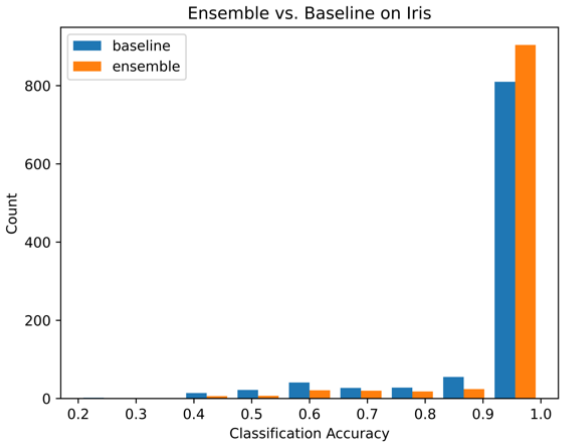

Fig. 10: classification accuracy of baseline vs. ensemble with shift

The mean classification accuracy of the ensemble model is 0.966 and the mean classification accuracy of the baseline model is 0.934. We ran a two-sided t-test and got a p-value of $3.23 * 10^{-9}$. This experiment mirrors what we saw in the previous experiment. The Islands of Fitness baseline model is outperformed by the non-Islands of Fitness ensemble model from Figure 5 despite using one-third of the population size. Additionally, we can see that the improvements from using our ensemble approach is greater when paired with the Islands of Fitness approach.

For the final 1000 runs, we will compare the Islands of Fitness approach with and without our ensemble approach. This time adding both noise and shift sampled from a Gaussian with a standard deviation of 0.25 and a mean of 0.0.

The results are presented in Figure 11.

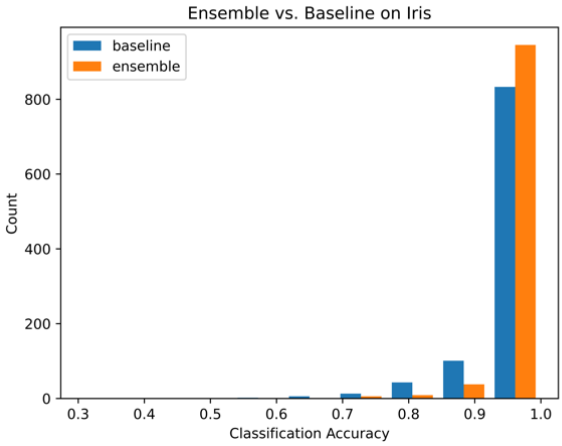

Fig. 11: classification accuracy of baseline vs. ensemble with noise and shift

Here the Islands of Fitness ensemble has an average classification accuracy of 0.988 and the Islands of Fitness baseline has an average classification accuracy of 0.969. A two-sided t-test returns a p-value of $5.28 * 10^{-16}$. Here we results mirror what we saw before, the non-Islands of Fitness ensemble model (shown in Figure 6) outperforms the Islands of Fitness baseline despite using substaintually less compute. Additionally, we see that same theme, namely that diversity preservation techniques seem to help performance for our ensemble method. This is demonstrated by the much larger improvement in classification accuracy compared to the improvement we saw in Figure 6.

The key takeaways from this experiment are that in all runs the non-Island of Fitness ensemble model performed better than the Island of Fitness baseline despite using much less compute and the diversity-preservation techniques like Islands of Fitness seem to improve performance for our ensemble style model.

For the next experiment, we will run our ensemble model on a non-convex problem.

G. Experiment 7

For this experiment, we will evolve an ES to classify an XOR dataset. The XOR data consists of two features, if both features are positive or both features are negative then the datapoint has class 1, otherwise the datapoint has class 0. The dataset contains 200 datapoints and can be seen in Figure 12

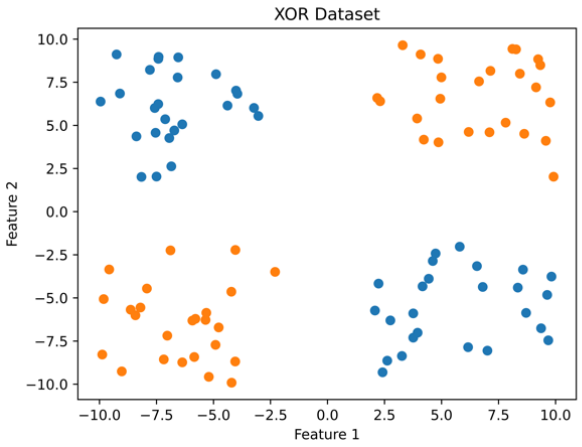

Fig. 12: XOR Dataset

To evolve an ES to classify the XOR dataset we will evolve the parameters of a simple neural network. The network will have one hidden layer with three hidden units and ReLU activations. So, that we can calculate the logits using the equation below:

$$logits = relu(XW_1 + b_1)W_2 + b_2 \tag{4}$$

Where X is the training data and W_1, b_1, W_2, and b_2 are the parameters of the neural network that we evolve. We will use an Islands of Fitness EA using the same hyperparameters

as we used in Experiments 4, 5, and 6. We include the hyperparameters in Table II below:

Hyperparameter	Value	Hyperparameter	Value
μ	45	λ	100
τ	0.5	τ'	0.34
Mutation Rate	0.05	Recombination Rate	0.0
Mutation Type	uncorrelated mutation with n step sizes	Survivor Selection	(μ, λ)
		Recombination Type	Intermediate
Diversity Preservation	Islands of Fitness	Subpopulation Size	15

TABLE II: EA Hyperparameters

We ran our EA for 1000 runs and got a mean classification accuracy of 0.982 for the baseline Islands of Fitness model and a classification accuracy of 0.989 for the ensemble Islands of Fitness model. The p-value from a two-sided t-test is $3.01 * 10^{-6}$. This is a clear increase from the ensemble approach unlike what we saw on the strictly convex problems in experiments 4 and 1. We can see the results in Figure 13.

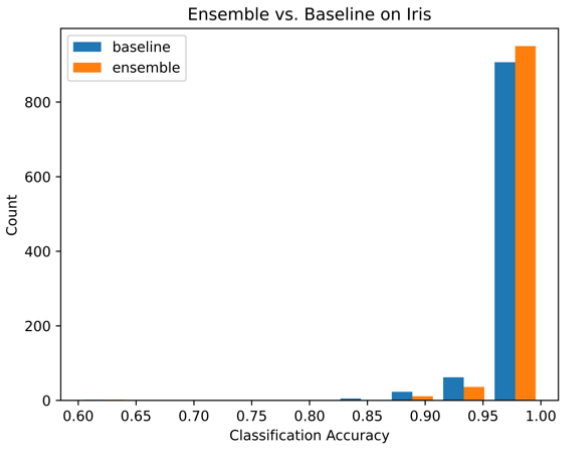

Fig. 13: classification accuracy of baseline vs. ensemble

For the next 1000 runs, we will run our ES with noise sampled from a Gaussian with a mean of 0.0 and a standard deviation of 0.5. The mean classification accuracy for the Islands of Fitness baseline is 0.980 and the mean classification accuracy for the Islands of Fitness ensemble is 0.987. A two-sided t-test returns a p-value of $4.90 * 10^{-9}$. We present the results in Figure 14

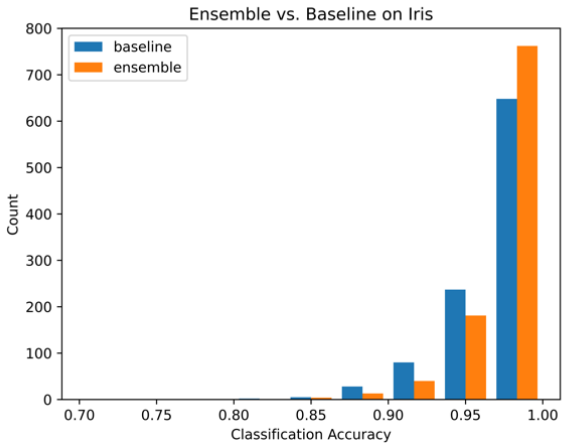

Fig. 14: classification accuracy of baseline vs. ensemble with noise

For the next 1000 runs, we will run our ES with a shift sampled from a Gaussian with a mean of 0.0 and a standard deviation of 0.5. The mean classification accuracy for the Islands of Fitness baseline is 0.981 and the mean classification accuracy for the Islands of Fitness ensemble is 0.987. A two-sided t-test returns a p-value of $5.75 * 10^{-6}$. We present the results in Figure 15.

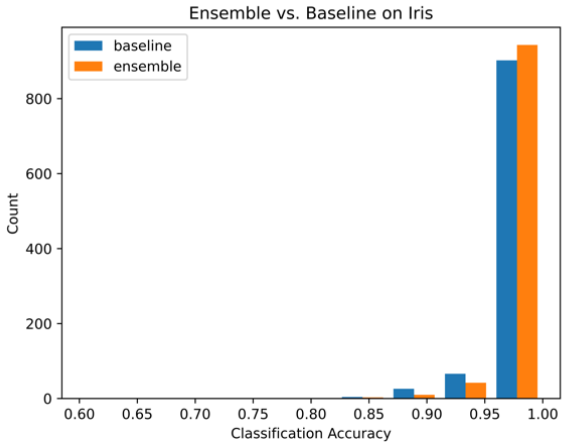

Fig. 15: classification accuracy of baseline vs. ensemble with shift

For the final 1000 runs, we run our ES with shift and noise both sampled from a Gaussian with a mean of 0.0 and a standard deviation of 0.5. The mean classification accuracy of the Islands of Fitness baseline is 0.977 and the mean classification accuracy for the Islands of Fitness ensemble is 0.984. A two-sided t-test returns a p-value of $1.49 * 10^{-6}$. We can see the results in Figure 16.

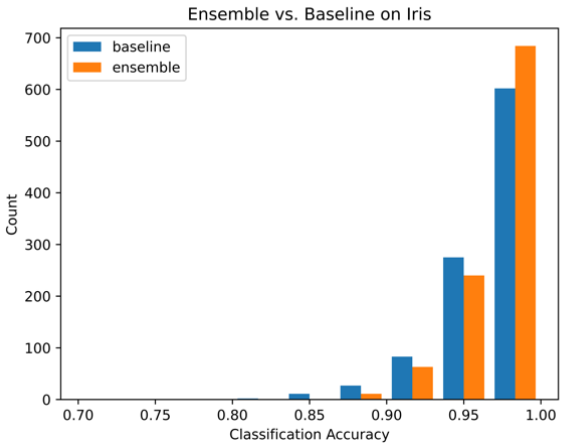

Fig. 16: classification accuracy of baseline vs. ensemble with noise and shift

In this experiment, we saw similar results to the previous experiments. Namely, we saw that the ensemble model improves classification accuracy. The new thing that we learned is that the ensemble model seems to help substantially in non-convex problems even without distributional shift.

V. Discussion

In this paper, we tried to treat the population of an EA as an ensemble. Our approach aimed to see if we could improve the performance of EAs at almost no cost. In Experiments 1, 2, and 3 we saw that our ensemble approach was able to improve the performance on the Iris dataset at almost no cost. In Experiments 4, 5, 6, and 7 was saw that we could get more improvements from our method by adding diversity-preserving techniques like Islands of Fitness. The only negative result we saw was that our method doesn't seem to help much when the problem is strictly convex. This was expected since ensembles need diversity to work. Additionally, this negative result isn't very consequential since strictly convex problems are rare, out method is almost no cost, and strictly convex problems are not best solved by EAs in the first place.

A. Future Work

With the time and computing resources that we had, we didn't get to run every experiment that we would have liked to. It would be interesting to try our approach with different EAs. In our work, we only tried our approach with Evolutionary Strategies, but our approach should extend to any EA: Genetic Algorithms, Evolutionary Programming, Genetic Programming, etc. Additionally, it could be interesting to try our approach with different diversity-preserving techniques. We used Islands of Fitness models, but we could also use Fitness Sharing or Crowding. Finally, there are many different more diverse problems that we could run our algorithm on. In this work, we focused on classification, but there is no reason that we need to use classification.

References

[1] A. Eiben and J. Smith, *5.5.3 Fitness Sharing*, p. 92–93. Springer, 2nd ed., 2015.

[2] A. Eiben and J. Smith, *5.5.4 Crowding*, p. 93–95. Springer, 2nd ed., 2015.

[3] A. Eiben and J. Smith, *5.5.6 Running Multiple Populations in Tandem: Island Model EAs*, p. 95–97. Springer, 2nd ed., 2015.

[4] F. GALTON, "Vox populi," *Nature*, vol. 75, pp. 450–451, Mar 1907.

[5] Q. Bui, "17,205 people guessed the weight of a cow. here's how they did.," Aug 2015.

[6] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, pp. 123–140, Aug 1996.

[7] R. Maclin and D. W. Opitz, "Popular ensemble methods: An empirical study," *CoRR*, vol. abs/1106.0257, 2011.

[8] A. E. Eiben and J. E. Smith, *3.1 What Is an Evolutionary Algorithm?*, p. 26–26. Springer, 2015.

[9] A. Eiben and J. Smith, *4 Representation, Mutation, and Recombination*, p. 60–60. Springer, 2nd ed., 2015.