Tamás Csilling [UP1105444]
Aritifcial Intelligence (CEID1060)
June 16, 2023

Genetic Algorithms Assignment

https://github.com/OnionKiller/AI_Homework_GA-HAR-PUC-Rio

B1. GA design

Preprocessing. With the help of *ydata-profiling* python library, I already assesed the dataset. It showed, that the data mainly consist two separate part: the parameters about the users which is very highly correlated, and the measured sensor values. There was a value error in the Z4 data, which was improperly exported from the csv file. The errorous live was dropped. Also the non US compatible floating point format resulted in custom data exporting functions. Because all of the user data is highly self correlated, and is irrelevant from the robot position, these values should be dropped from the dataset. Otherwise these data is not highliy correlated to any other measured values, so it is not a clear issue to keep them in the trainig dataset.

Question 1. a) Coding

Because the data will be used in a GA (Genetic Algorithm), well defined boundaries for the variables are very important. To achive this, the values are transformed to a $[0, 1]$ interval. It is done with simple min-max scaling, but it is errorprone for outlayers, which is the case for some of the truncated data noise. The data otherwise shows a clear gaussian distribution, with ocassional multi gauss distributions. Only centerig the data is not really nescessery, most of the data is relatively centered. Altough this can be solved with standardisation I choose to use the original data boundaries clamped to $[0, 1]$ to get all possible sensor settings. Also did an alternative solution, where the data is standardized, and then shifted to have only positive values. In this case, the boundaries are defived by the minima and maxima values of the data. Since the dataset is now a $[0, 1]$ bound floating point for all of the 12 sensor data, the gene is constructed to be a 12 variable gene, where the variables are coded as floating points. It requires some specialised mutation, and crossover functions to be implemented, but they have generally a better performance, than quantitaztion.

Question 2. b) Surplus prices

Since the whole dataset was standardized, unless there was an extreme difference between the median values of the dimensions, the data is expected to be unique in all variables of the gene. This is because there is no saturation (neither from normalisation, nor variable boundaries, and due to the data shape, there are minimal outlyer values).

Question 3. c) Initial Population

To create an initial population, I choose to uniformly sample the problem space, which is in this case, for the normalized values, a $[0, 1]^{12}$ hipercube.

Question 4. d) Similarity calculations

The goal of the similarity caluculation is to determine if two configuration vecotrs are simimilar or not in the 12 diemnsional problem space. This high dimensional problem space rules out the Euclidean and Manhattan distance, as these two distance functions would result a very high isosurface size. (12 dimensional hipersphere and hipercube both have a proportionally very high surface.) Further drawback of these functions are, that they are sensitive to absolute value differences, so they are not good for comapring similar vectors, with different magnitudes. Cosine similarity measures the similarity between vectors disregarding the length of the vectors. This measure calcualtes the angle between the two vectors. This is especially practical, as the problem is coded as 12 dimesional vectors, thus defining a measure over vector angles

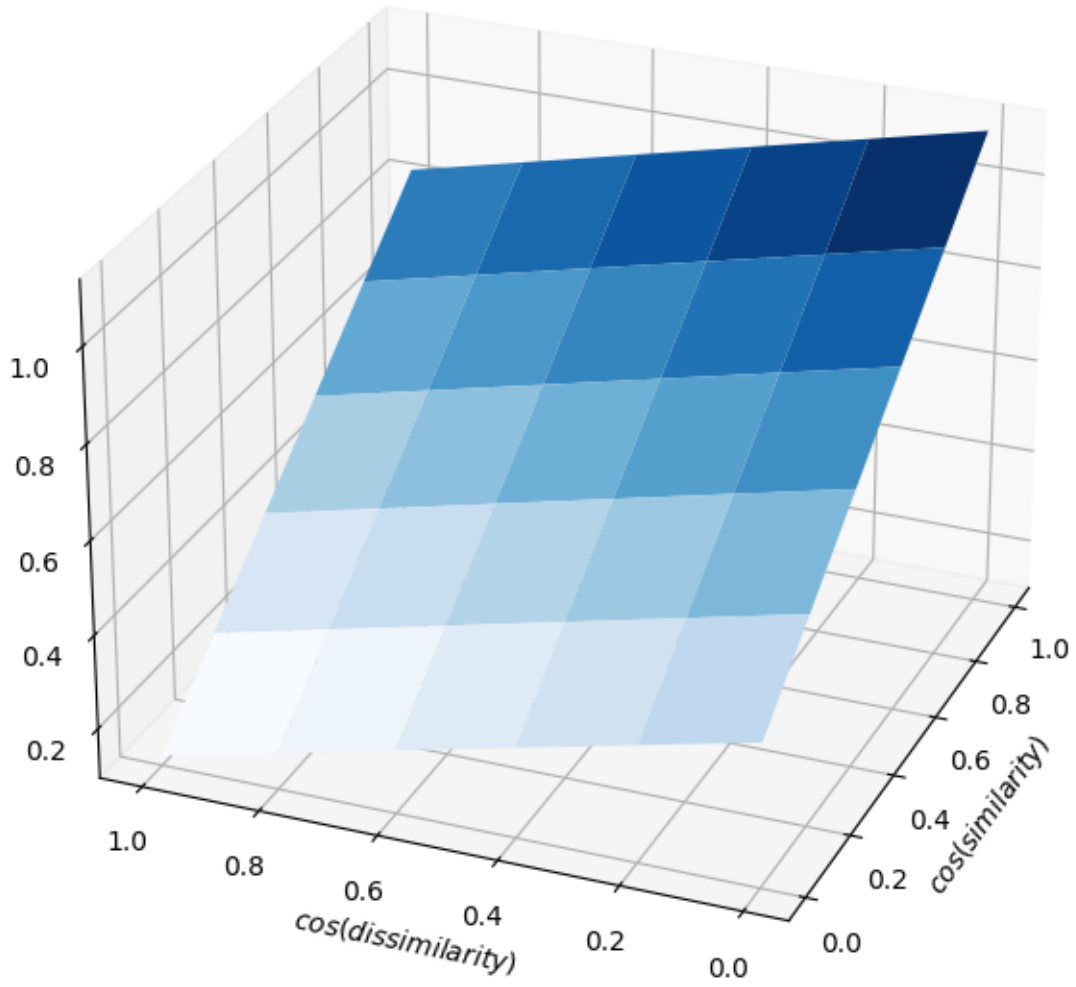

FIGURE 1. Fittness function value in cosine parameter space

is a very efective fitness function. We want to have as small angel between the teaching vecotr, and the proposed position, as possible. Because the dataset is standardised

Question 5. e) Fittness function

$$F(v) = \frac{cos(v, t_s) + c(1 - \frac{1}{4}\sum_{i \neq s} coa(v, t_i)}{1 + c} \tag{1}$$

In this fittness function the cosine measure has a value between $[0, 1]$ beacuse the vector values are bound to $[0, 1]$. Assuming that the c values is also bound to $[0, 1]$ the fitness value will be between $[0, 1]$ due to the normalization with $1 + c$. If the vector values would not be bound to $[0, 1]$ the F value could be negative, but this is not the case in this setup. With this setup the formula is a suitable fitness function as it will have it's maxima at the case where the vector v is parallel with the sitting M.O. and orthogonal with all of the other cases, and it's minima is wehere v is orthogonal, and parallel with all the other classes M.O.

For the c value 0.4 is suggested, as the second term is normalized, and thus the two factors can have equal proportion in the fitness function. To have the main effect of beign similar, the dissimlarity has only 40% effect. This vlaue at this point is only heuristicly setted. It can be tweaked appropriatley in presence of actual data, and results.

Question 6. f) genetic Operations

The optimisation setting were tested before sparsly, some of them were not validated by tests.

i) For selectioon over the population roulette base, ranking based and tournament methods are conssidered. I selected roulett based selection, because the population sizes are relatively small, thus sorting the dataset is not a significant overhead, and they peprform better selection than rank based selections, allowing lower fitness results back to the population, resulting in better global convergence. Touurnament selection would be a great alterative to the roulett based method, I only choose roulett, due to the low population size.

ii) Because the gene a vector of dependent data, but these data doesn't have any sequencial information at them, the simple and multi point corssovers are lass practical than uniform crossover, because we don't need to keep the sensor values together to have a good result. May some of the sensor data are dependent enough, that a multi point crossover could bring extra value to the process, generally uniform crossover is better here.

iii) Because the top genes may be close to the optimal solution, mutating the top gene is beneficial, to exploit the gene to have a better fit to the local best where it is currently. This is only because the gene is based on floating points, thus micro optimalizations of the top gene are almost all the time available and can introduce better results to the gene pool.

B2. Implementation of the GA

For the most performance insead of implementing a highly optimalized and vectorized version of the GA, I used an external framework, which implements GA. It has almost all the feautres required by the assignment, only the biased uniform crossover implenetation was missing, which was really easy and convinient to implement.

B3. Evaluation and Impact of Parameters

Question 7. a) Matrix testing of the parameters

	Population Size	Likelihood of crossing	Likelihood of mutation	Average score	Average iteration count
1	20	0.6	0.0	0.7055	33.3
2	20	0.6	0.01	0.7129	105.1
3	20	0.6	0.1	0.7135	108.7
4	20	0.9	0.01	0.7117	101.4
5	20	0.9	0.01	0.7132	161.7
6	200	0.6	0.0	0.7140	60.7
7	200	0.6	0.01	0.7150	272.0
8	200	0.6	0.1	0.7149	187.1
9	200	0.9	0.01	0.7150	213.0
10	200	0.9	0.01	0.7149	236.2

From this it's clear, that small population stopped more earlier, but the larger population has better results. Most of the solutions are really good from the start, the fittnes function shows minor improvement.

Question 8. b) Plotted results The results for the different runs, with standard deviation error bands. The small population results are on teh 2 figure, while the 200 population results are on the 3 figure.

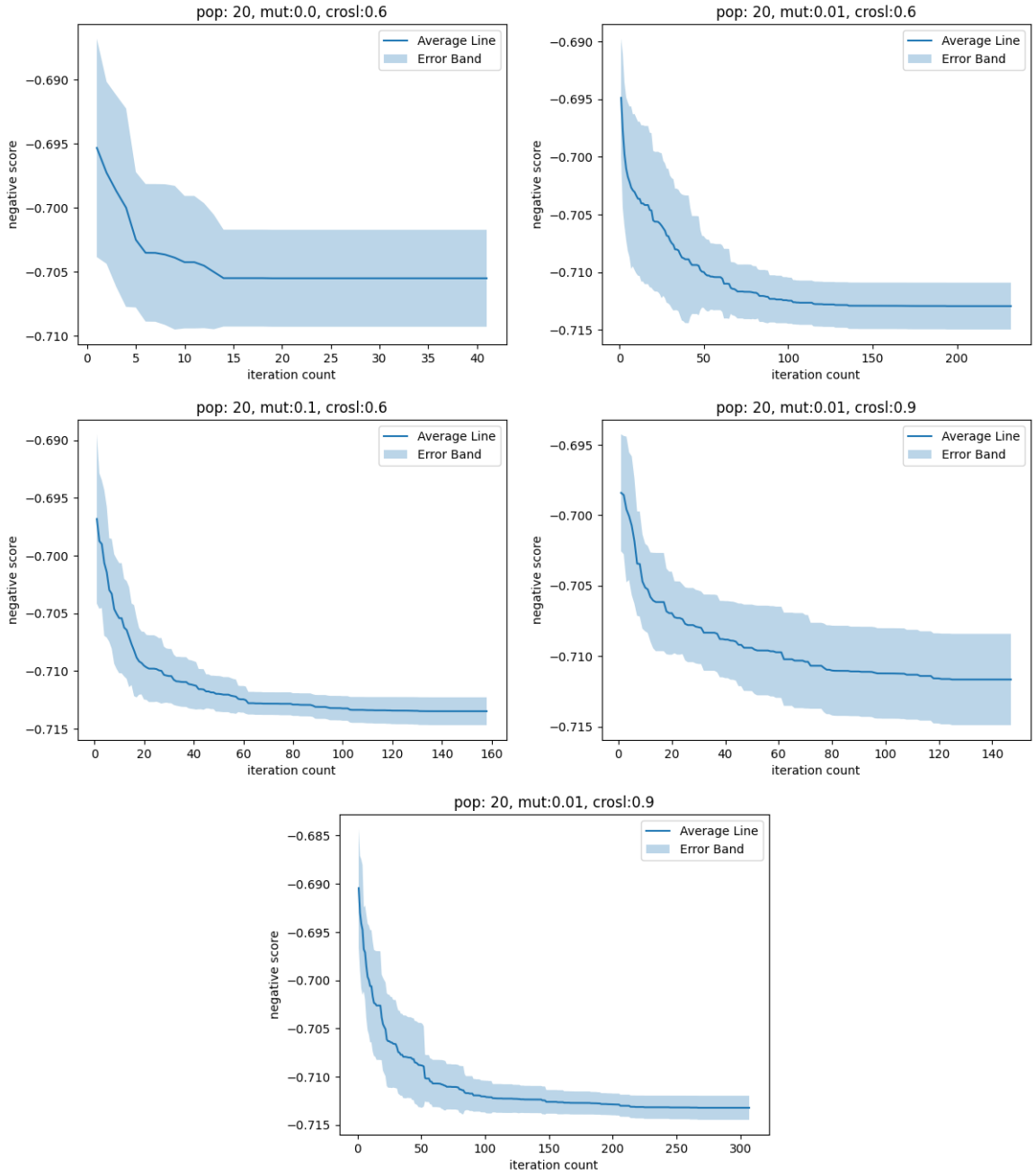

FIGURE 2. Average inverted GA scores over iteration with error where popultaion is 20

Question 9. c) Conclusion

It is clearly visible on the 4 figure, that higher popultaions create way better results. These results are siginificantly better. The oter paramteres doesn't have such a big impact, except if hte mutation value is 0, which has negative effect especiall on low population numbers. It is visible, that for low population numbers, hgher mutation values scored better results. If the crossover likelihood was high, the results were less good, because the ever changing genes, which resullted in to much changes between the genes. This is also clear for small population numbers, while for high population scores, the method run long enough to find a good solution in almost all cases. It is also apparent, that bigger mutation result in faster convergence in this scenario.

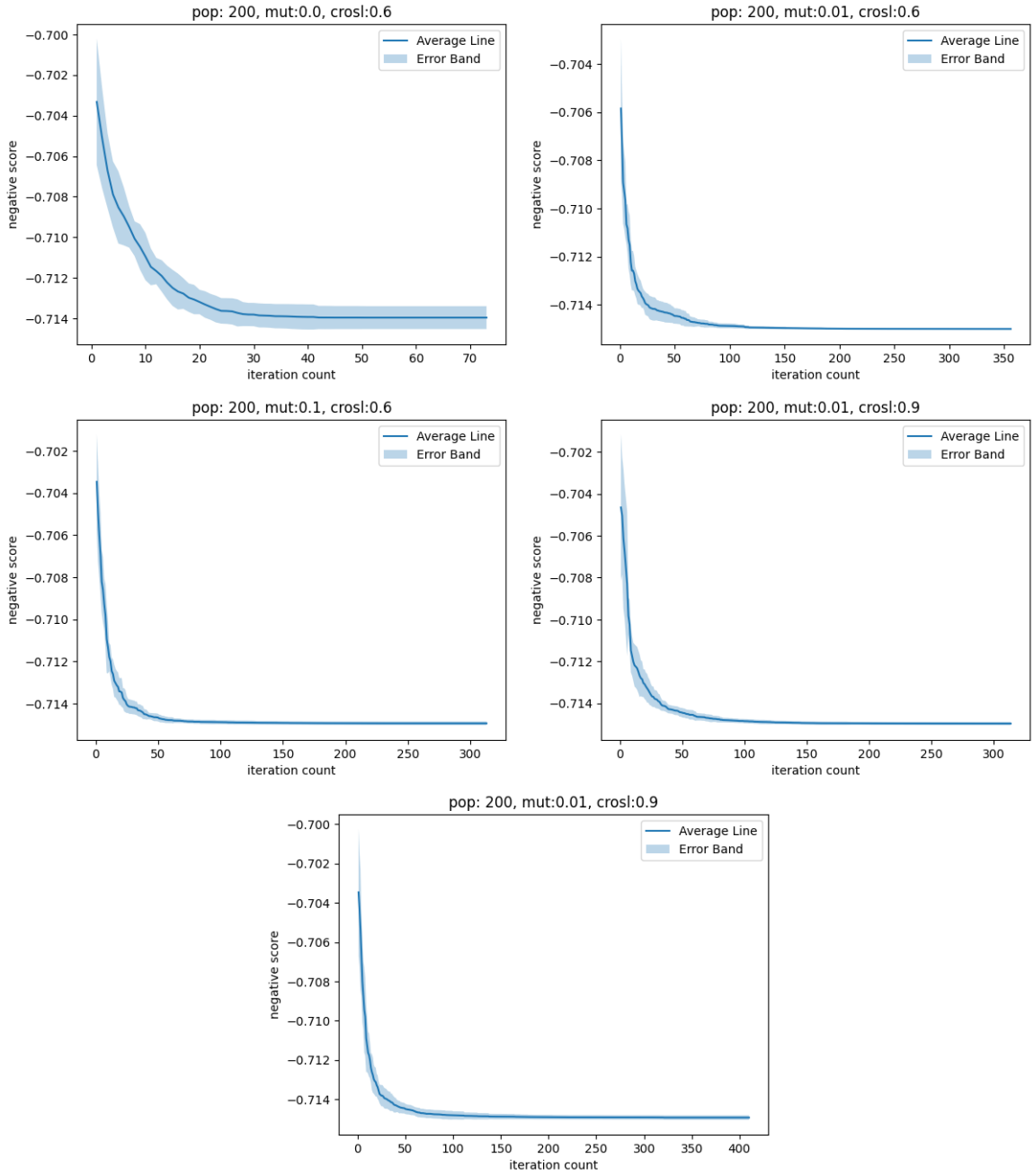

FIGURE 3. Average inverted GA scores over iteration with error where popultaion is 200

Question 10. d) Comparsion with NN

Sadly the trained NN used different parmeter layout, so it is not possible to put the result into the NN modell.

PATRAS

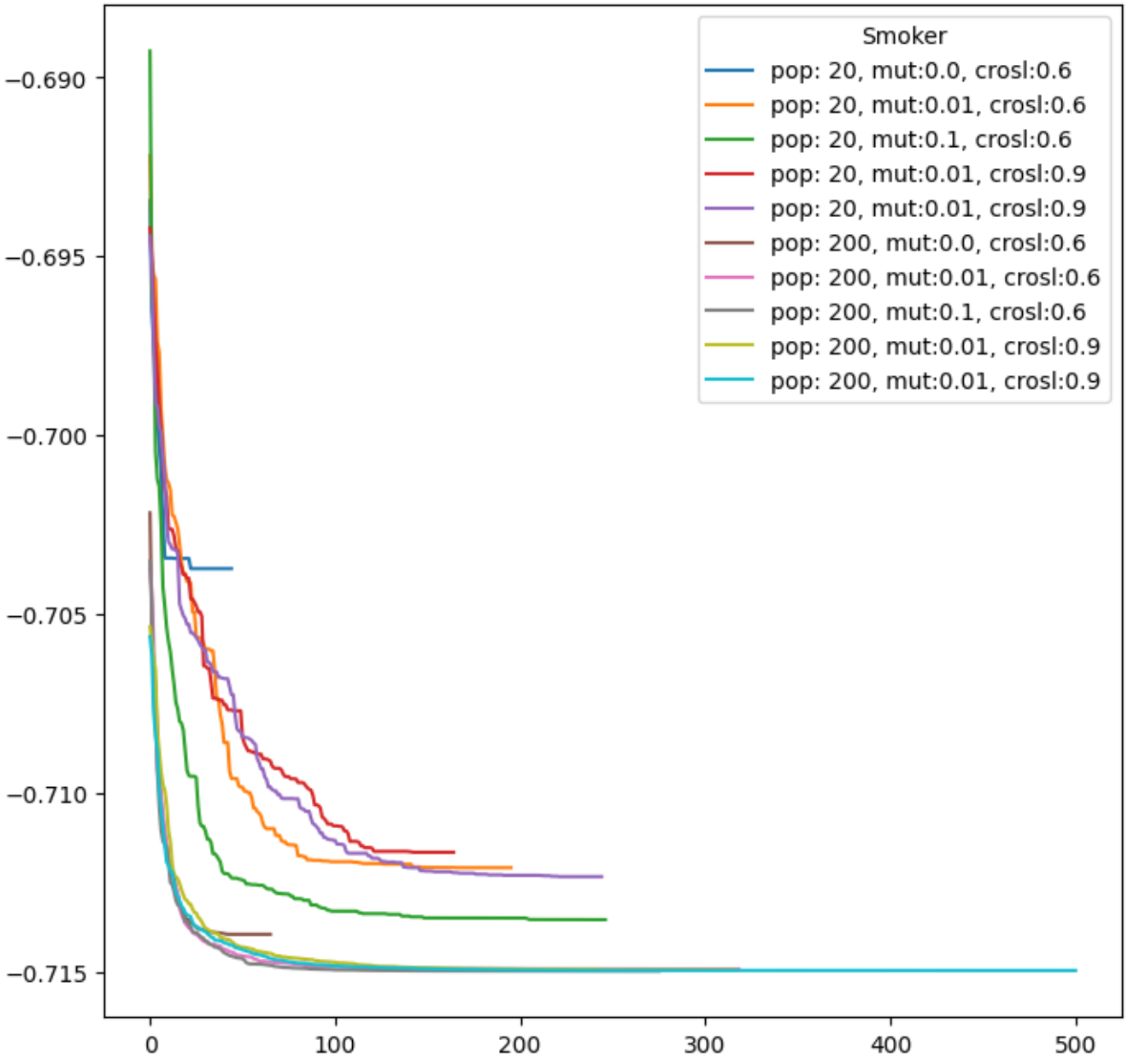

FIGURE 4. All of the average scores compared