EVOLUTIONARY COMPUTING: STANDARD ASSIGNMENT - TASK I

August 31, 2020

Read this whole document before starting your assignment!

1 Introduction

The objective of this assignment is to gain experience with using Evolutionary Computation. To this end you will develop, apply, and compare Evolutionary Algorithms for the task of video game playing using a python framework called EvoMan.[1] The assignment is to be done in groups of four (or three, though less preferable). **The deadline –not negotiable– is as follows:**

week 4 of the course, 25-09-2020, Friday 23:59

Each day of delay in the submission will result in a deduction of 0.5 points from your Task grade.

2 Resources

All the resources (files) you need to install/use the EvoMan framework and start your experiments are available on GitHub, including code, manual, and baseline paper.

[1] `https://youtu.be/ZqaMjd1E4ZI`

3 Preparation

- *git clone https://github.com/karinemiras/evoman_framework.git*
- Install the framework and run the demo controller_specialist_demo.py (only to test the installation)
- Play the games yourself using the keyboard or joystick, with the purpose of understanding the nature and challenge of the problem. For this you can use the human_demo.py

4 Supplementary material

- Manual of the framework - evoman1.0-doc.pdf (on GitHub)
- Baseline paper: multi_evolution.pdf (on GitHub)
- Reporting Research (on Canvas)

5 Task

- WHAT: Implement two EAs that use the simulation mode "Individual evolution" of the framework to train a **specialist** agent.
- HOW: Use 3 enemies (games) to experiment with your algorithms. For each enemy, make an independent experiment evolving an specialist agent, and for all cases use exactly the same algorithm parameters. You are allowed, but not required, to use the code of the neural network (demo_controller.py) available in the scripts of demonstration.
- WHAT TO HAND IN: The report as specified below and your code.
- HOW TO HAND IN: Submit a single compressed file named *groupnumber.zip* containing a folder with the same name, including: the report named as groupnumber.pdf and your code (include the evoman folder, but no demos).

6 Rules and guidelines

- You are NOT allowed to simply copy and paste the code available in the scripts of specialist demonstration. That is, you need to implement your own code, building from the dummy demo. You may use any EA framework if you wish[2].
- You are allowed, but not required, to use the default fitness functions of the framework.
- You are not allowed to make changes to the source code of the framework (anything in the 'evoman' folder). Making such changes would be qualified as fraud, leading you to FAIL the assignment.
- For the "two EAs", you can use two different EAs, or the same EA with differences such as, the evolutionary mechanisms, parameter tuning, etc.
- With both algorithms (and for each enemy) you must repeat your final experiments 10 times (independently) and your report should present the statistics based on these 10 runs.
- Compare your algorithms by enemy, making a line-plot across the generations, with the average/std (for the mean and the maximum) of the fitness. Note that you need to calculate the average (over the 10 runs) of the mean and maximum (over the population in each generation). Do one plot by enemy, thus, separately.
- Compare your algorithms by enemy, testing 5 times your final best[3] solution for each of the 10 independent runs, and present the *individual gain* in box-plots. Note that you need to calculate the means of the 5 times for each solution of the algorithm for the enemy, and these means are the values that will be points in the box-plot. In summary, it is a total of 3 pairs of box-plots (so 6 boxes), being one pair per enemy. Additionally, do a statistical test to verify if the differences in the average of these means are significant between the groups of best solutions, when comparing two algorithms of an enemy.[4]
- The *individual gain*, i.e., the gain measure when playing against one individual enemy is calculated as

[2] One example is DEAP, which provides multiple resources for evolutionary algorithms. http://www.jmlr.org/papers/volume13/fortin12a/fortin12a.pdf

[3] Best means highest energy level for the player, or individual gain.

[4] There are R packages that plot multiple box-plots and automatically test them.

$individual\_gain = player\_energy - enemy\_energy$. [5]

- The environmental parameter of *level* of difficulty of the game should be 2 for all experiments, as well as the parameter *contacthurt* should be set as "player" (both these values are set by default).

- As for the other environmental parameters, you are free to choose their values, but required to report any changes to their default values.

7 Report structure

Your report should have a maximum of 3 pages (containing everything, e.g., text, figures, references, etc). Report format must follow the GECCO19 template. For Word see `https://gecco-2019.sigevo.org/index.html/dl1241`; for Latex, see `https://gecco-2019.sigevo.org/index.html/dl895`. Reports exceeding the maximum number of pages will automatically FAIL the group on the given Task.

You need to include the following info at the BEGINNING of the assignment: Name of the course; Number and name of the Task (e.g., Task 1: specialist agent); Number (and possibly name) of the team (e.g., Team 12, Team Jacob); Name and student ID nr. of the team members; Date. It is allowed to use an extra page for that, but this cover page should not contain anything beyond this identification info.

The pages describing the actual content should be as follows:

- Introduction (make sure to define a clear research question or goal, cf. Reporting Research document).

- Methods: explain how your algorithms work and its motivation, parameter settings, experimental setup, fitness function, budget, etc. Make sure everything is reproducible with the information presented!

- Results and discussion: discuss the differences between the results of your EAs; do they outperform each other? Comment on a possible explanation for that; Discuss the differences between your results and the results of the baseline paper; are they better/equal/worse? Comment on a possible explanation for that.

[5] These 5 repetitions are important because the environment is not exactly the same every time you play, and thus, sometimes your algorithm finds solutions that sometimes win and sometimes loose. Although it was not feasible to perform such repetitions during the evolutionary runs, in the final test we can afford it.

- Conclusions
- Literature list / bibliography: cite all works you are using in your project.

8 Tips

- Note that the demos are not part of the framework, but only a demonstration of how one specific user could use the framework. **The cleanest example, and from which you should start is** *dummy_demo.py*. We emphasize that **all you need to know to start your own experiments is in the manual (evoman1.0-doc.pdf), and it does not depend on the demos**.
- Add legends with large letters to the plots, so that you can reduce the plot size while still making it readable, saving a lot a space.
- Test your algorithm(s) at first using a small population for few generations. This also applies for tuning.
- Remember to count the time of some of your experiments and take this into consideration to plan carefully your budget.
- Be organized concerning the output files of your experiments.
- It might be interesting to normalize the inputs for the controller (as done in the default controller).
- Do a literature review using the references of the baseline papers. One key author to have in mind is Julian Togelius.
- Try to make a group with people of diverse scientific background, e.g. an AI student, a CS student, and a biology student.
- Apples and oranges: Evoman is a game playing framework, and not an EA framework (like for instance, DEAP).

9 Grading

Each Task is worth a maximum of **10** points, and the final assignment final grade is calculated as: **40%** (Task I) + **60%** (Task II). Task II is described in another document.