Natural Computing 2020/2021: Assignment 2

This is an exercise worth 40% of the course mark. The deadline is 4pm on 19/11/20. You will work on this assignment in teams of two students. You are asked, not to share your solution with other students outwith your team, but you can discuss your approach and any technical problems.

The assignment requires you to write or modify a program, and to use it in order to carry out a number of tasks. You can choose any common programming language, but advise can be given only related to the provided JavaScript package. The use of other existing software packages is allowed, if you give credit to the sources. Feel free to use also other resources and literature to prepare your work and to support your argument in the report, but make sure to cite all the papers and resources that you have used for this assignment. Write about your investigations in a report with a length of maximally six pages. Credits and references to not count towards the page limit.

More details are given below. In the tutorialsand in the classroom Q&A sessions will be time to clarify your questions related to the assignment. For urgent questions, please contact the lecturer.

Tasks

Task 1[1]

Optimize the weights of a neural network by PSO for the two-spirals dataset. You can use either the *Spiral* data produced in the Neural Networks Playground https://github.com/tensorflow/playground or the provided dataset *two_spirals.dat*. Use the half of the data for training and half for testing.

1. Design a fitness function based on the testing error.
2. Define a search space for the problem, based on a fixed network structure. The problem is more easily solvable if non-linear input features are used, i.e. in addition to x and y, also sin(x), sin(y), as prespecified in the Neural Network Playground. You can then work with one hidden layer of 4-8 neurons, which corresponds to a search space of 24 to 48 dimensions. You can also add the features x^2, y^2, but rather not xy.
3. Run a PSO on this task and show your results.
4. Consider for comparison also a different architecture with only linear input features.
5. Discuss the effect of the PSO parameters.

Task 2

Use a Genetic Algorithm to optimise the shape of a neural network. The shape is a set of numbers specifying how many neurons there are per layer, e.g. [1, 2, 3, 1] means the network will have one input node, two nodes in first hidden layer, three nodes in second hidden layer and one output node. Decide based on your experience in the previous question, whether you want to continue with PSO or whether you rather use a standard gradient-based neural network learning rule.
In practical application of machine learning, a validation set is often used in addition to the training and testing dataset. The validation set, plays the role of the test set during training, while the actual test set in used in the end to evaluate the whole procedure. Here, you can use half of the data for neural network training, and use the results for both halves for the fitness in the GA. This can help to save time: e.g. if either of the losses is not decaying quickly from its stating value, if the test set loss is much larger than the training set loss, or if the losses are strongly fluctuating, you may consider to break the fitness evaluation early.

1. Evolve the structure of the network by a GA. Based on your choice of the problem encoding, design a neural network creator that specifies the shape of the neural network given a binary (or integer) string that is evolved by the GA.

1 This task is based on question B3 from the first (unmarked) coursework.

2. An initialisation function for the weights can be evolved as well, but do not evolve also the individual weights from the GA, as this will be left for the low-level algorithm. Also a learning rate or batch size may be subject to the evolution, it is, however, perfectly acceptable to prespecify these design choices for the neural network.
3. Discuss the operators and parameters of the GA, and performance of the GA.
4. Comment also, how you control the complexity of the evolving network, and how the best one of the evolved individual neural network works.

Task 3

In this part, a GP is to be applied to the same problem as in task 2. You will now also control the also the structure of the network. Try to reuse as much of the code of the previous task as possible.

1. In addition to the quasilinear (i.e. linear sum and sigmoid activation function) units, also other options for the function that is represented by a node, are to be provided for selection by the GP, such that all nodes can realise functions similar to the input features in the Neural Networks Playground. In addition, you may consider to provide an option for the GP to select whether a link is present or absent ("isDead").
2. Discuss the operators and parameters of the GP, and the obtained performance.
3. When either starting from scratch or from the best network found in Task 2, do you find better networks here? Discuss also the criteria that you apply to tell which network is better.
4. Your algorithm will be similar to a Cartesian Genetic Program (CGP). Among the observations that have been made for CGPs one can find, e.g., that long-thin architectures are better than short-thick ones, or that populations of only 4-5 individuals are best, or that cross-over is not needed. Can you confirm any of these?
5. Add a brief outlook on any other tasks that you would consider as interesting the present context.

Natral Computing 2020/21 Assignment 2: Notes and Hints

Particle swarm optimisation (PSO), Genetic algorithms (GA), and Genetic programming (GP)

- This is a natural computing assignment, and neural networks can be understood as biologically inspired computational methods, so they do fit in the course. However the emphasis in your work and your report should be on PSO, GA and GP.
- In the present case, we know that the global fitness optimum is at zero (unless you add noise to the data). It will not be necessary to reach this optimum and in some cases this may not even be possible. If your best results are not very close to the optimum, make sure that you can show that your algorithm works by considering question such as: Does the fitness show any improvement over time? How do the parameters of your algorithm affect the performance? Does your algorithm perform better than random search?
- If you work with neural networks of different shape as individuals, you may find that crossover is difficult to implement or that it does not lead to improvements. You can choose a maximal network and perform the crossover in this space, or to work with specific operators that are less general, but more meaningful than the single-cut cross-over. It is also possible to avoid crossover and to work only with mutation, where perhaps different types of mutations are useful.
- We have mentioned standard values for population sizes in the course. You will like to work with populations that are smaller then these recommendations. It is important that you see an improvement of the fitness, but not that you find the *global optimum*. Nevertheless, make sure do discuss such choices in your report.

- For PSO,, the number of optimal neural network solutions increases for higher search space dimension, but each generation takes longer and also the exploration may require more time. As the complexity of the task depends mainly on the complexity of the dataset, it is (apart from the extremes) not immediately clear which choice is better, and the result may also depend on the PSO parameters. Any comment about this in your report will be appreciated!

Neural networks

- Zero test loss is often not achievable in the two-spirals problem, because the two categories are very close in the centre, such that there the test data may not be discriminable based on the information from the training data. This become worse when noise is added. It is still a good idea to add a bit of noise to the data, to test the robustness of the results.
- If you find that your implementation is too slow to solve the two-spiral problem, you may use the circle data set instead of the two-spirals data set (these data set can be found in the neural network playground).
- As mentioned in the tasks the two-spiral problem is quite hard, if only linear input units are used. Feel free to use also the other available input units, e.g. sinus-like features or multiplicative inputs. It may be tempting, but it is strongly discouraged to use spiral-like input features for this two-spiral problem, because this would be a heuristic, but not a metaheuristic approach.
- The maximal size of the networks in the Neural Networks Playground is quite limited, but do not hesitate to limit it further to be able to search efficiently in a smaller search space. Consider, however, that very small networks cannot solve the two-spirals problem.
- Regularisation has not been mentioned above, but is obviously an option that can be considered here as well.

Implementation

- There are three options for the implementation, choose one of the following:
    1. Download the neural networks playground https://github.com/tensorflow/playground and adapt it to the tasks as necessary.
    2. Use the code that will be provided at the learn page, which is somewhat adapted from the github version of the neural networks playground towards task 1, but will require further modification.
    3. Use the implementation from a neural network package of your choice or write the code yourself.
- Please make clear in your report which option you have selected. If you have selected option 1 or 2, submit a zip file in the same format as the package you started from. If you have used your own (option 3) code, explain in the report how your code can be tested and make sure this is possible without much effort.
- It will be helpful, if you comment your code, and if you include a README file how your code can be executed, but there will be no marks for good code (you may lose marks if neither the text nor the code is clear).
- It may be useful to write some data to a file for data representation in your report. In Java-Script this is a bit tricky, but is not a problem either. You can find examples online.

Report

- The maximum length of your report will be 6 pages (in a 12pt font). If you report is longer than this page limit, the marker will not be required to read beyond the sixth page, so you may loose marks. However, you can add an appendix that the marker may choose to look up for evidence of any statements in the main part of the report. A shorter report is possible, but it may be difficult to represent the answers to the questions in a suitable form in less than four pages.

- Make sure that your design choices are well explained, and that you give at least a brief interpretation of your results.
- Use graphical elements to demonstrate your approach and to show your results.
- Cite all sources you have used, expect the slides from this course or your report on the first coursework. Bibliography and credits do not count towards the page limit.

Marking criteria

- Marking will be based on the description of your results in the report.
- This coursework is worth 40% of your course mark.
- Marking of this coursework will be out-of 100 points. You can get up to 50 marks for task 1, 25 marks for task 2, and 25 marks for task 3
- Each of the following aspects will account for 20% of the marks
  - Understanding of the problem and justification of any design choices
  - Completion of the task
  - Quality of the results
  - Analysis and discussion of the results
  - Quality and clarity of the description and representation in the report
- If you did not perform any numerical experiments for one of the tasks, you can still get marks for this task, e.g. by describing a potential design and discussing any issues you had in your attempts etc.

Teams

- You are expected to work in teams of two students. You will submit only one report per team.
- If you have been working in a team with one other student for the unmarked coursework, you can continue in this team.
- You can choose to work with another team mate for this assignment, but please make sure that any former team mates are aware of this change.
- If you are not sure how to find a team mate, please send an e-mail to the lecturer (michael.herrmann@ed.ac.uk). You will then be paired up with another student who also sends such an e-mail.
- If you are working alone, your report will be marked by the same standards as reports submitted by teams, unless there is only one student submitting a single-authored report, in which case there will be some compensation for this student.

Submission

Submit your report and a zipped archive of your code via the NAT Learn page by

4pm of Thursday, 19. November, 2020.