EL2805 Reinforcement Learning

Computer Lab 2

December 2nd, 2024

Division of Decision and Systems Control
School of Electrical Engineering
KTH Royal Institute of Technology

Debugging and Hyperparameters Selection

Introduction
Understanding whether your model has not been properly implemented, or either an hyper-parameter has not been chosen accurately, is hard. This is already difficult in Deep Learning architecture, and worsens in the case of Deep Reinforcement Learning, which is at the intersection of both Deep Learning and Reinforcement Learning. This requires Deep RL users to be not only skilled in programming and deep learning architecture, but also to be proficient in the theoretical concepts of RL.

Unstructured problems
Having said that, one thing you should learn from the lab is how to deal with this kind of problem (it will be a major skill when you will start your post-uni career). This is about answering unstructured questions, i.e., questions or problem that at first glance seem quite hard, if not impossible, to answer without additional data or structure. I could ask you to roughly estimate (right now) what is the number of devices connected to the internet, without letting you look up at the data anywhere. What would you do? Or I could ask you to roughly estimate the number of of gas stops in Stockholm. In the former case one could try to split the questions in several small questions, such as: what is the number of devices connected in Sweden? In the latter case you can try to think of the number of gas stop stations you can find in a radius of 1-2 km where you live and use this as a rough estimate of the density of gas stops.

My model does not work!
Well, I think you get the idea. You need to give structure to the question. What about Deep RL? My model does not work, what should I do? Start with a simple binary hypothesis testing:

1. Either your implementation is wrong
2. Or your hyperparameters are wrongly chosen

If you manage to assess that your implementation is correct, then you not need to worry anymore about this issue. Therefore making sure that your implementation is correct is of the foremost importance. How can you then assess if your implementation is correct? Again, the answer is structure! Let's take as an example DQN: what does it mean that my implementation is wrong? That either one of the components in DQN is incorrectly implemented or that some of the components do not correctly communicate with each other.

Assess each component of your model individually! Test if that component is behaving as expected, and then assess if the components are correctly communicating with each other! If that is the case, then most likely your model is correctly implemented, and we can move on to the problem of choosing the hyperparameters!.

Manual selection of the Hyperparameters
Ok, your model is correctly implemented, great! But it is still not solving the problem... what should you do?

We could start by giving some structure: find first which are the hyperparameters that depend on another one. For example, in DQN the capacity of the buffer L is roughly $L = CN$ where C is the period to update the target network and N is the batch size. In this way you have 1 less parameter to worry about. Is there any other correlation? Is there any parameter that has a very low importance ?

Remember that overall the learning rate is the most important hyperparameter, and that increasing the batch size allows for a larger learning rate, since the quality of the gradient improves. Remember also that a larger network requires more training!

Once you understand what kind of correlations there are, try to sort out the effects of the hyperparameters on the learning process. Some problems are:

1. The agent is forgetting what has been learnt so far. Is this an RL or DeepL problem?
2. The agent is not learning at all. Is this an RL or DeepL problem?
3. The agent sometimes is able to solve the problem, other times it is not. Is this an RL or DeepL Problem?

For each of those problems, try to think which parameters are the most important. If the problem can be related to both RL and DeepL parameters, can you do some hypothesis testing to sort out the answer?

Automatic selection of the Hyperparameters
In case you are still wandering in the dark, another way is to automatically choose the parameters. There are two common approaches: *grid search* or *random search* (you can also try Bayesian optimization...). In the first case you are essentially trying all the parameters to see which one works best. In the second case you randomly sample values for each hyperparameter. In both cases, I suggest you to define a set of possible values for each parameter in order to limit the search size.