Example: Resetting constant parameters based on the selected environment

Aditya Mohan[1,2]

[1]Leibniz University Hannover
[2]`https://www.AutoML.org`

Method. In this example, we set the parameters of a Reinforcement Learning agent that trains using Proximal Policy Optimization Schulman et al. (2017) based on the environment that is presented to the agent. The parameters are based on the default values defined in RL Baselines3 Zoo Raffin (2018) for the given environment. The assumption is that the default parameters, which are supposed to perform well in the vanilla environments, could provide competitive performance on simpler environments without a lot of degradation in performance, when the original environment is modified by changing certain features with a small deviation around their default.

The changing features of the environment are called contexts, and the resulting environment is a variation of the same environment, both belonging to the same class of Contextual MDPs Hallak et al. (2015). The algorithm implementation used in this example is from `stable-baselines3` (Raffin et al., 2021), and it is trained for a total of trained it for $1O^5$ steps. This training time is divided into 10 epochs, each of 10000 steps, where all other parameters of the agent are constantly reset to the same values, as defined in zoo. Additionally, the tests were performed in a setting where 2 context features were varied at the same time with a standard deviation of 0.5 around the mean values. However, once sampled, these were kept fixed throughout the training time.

Limitations. The biggest limitation of this approach is that it does not perform uniformly well across the instance due to the hyperparameters remaining static. While the agent is still able to solve the easier environments, it struggles heavily with the more complicated ones even though the hyperparameters are supposed to be optimal. Thus, going for a more dynamic approach of configuring hyperparmeters using Dynamic Algorithm Configuration (DAC) Biedenkapp et al. (2020) can potentially alleviate this issue.

Reproducibility. We provide the code and the command to reproduce the model in `README.md`. The runs take approximately 55-75 seconds per instance.

References

Biedenkapp, A., Bozkurt, H. F., Eimer, T., Hutter, F., and Lindauer, M. (2020). Dynamic algorithm configuration: foundation of a new meta-algorithmic framework. In *ECAI 2020*, pages 427–434. IOS Press.

Hallak, A., Di Castro, D., and Mannor, S. (2015). Contextual markov decision processes. *arXiv preprint arXiv:1502.02259*.

Raffin, A. (2018). Rl baselines zoo. `https://github.com/araffin/rl-baselines-zoo`.

Raffin, A., Hill, A., Gleave, A., Kanervisto, A., Ernestus, M., and Dormann, N. (2021). Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.