Teachable ML in Retrospect

16/01/2023
Data Challenge

Mateusz Mierzjek
Simona Orzan & Qin Zhao

In Short

Teachable machine learning is a rapidly growing field that seeks to make the process of training machine learning models more accessible to non-experts. Using user-friendly interfaces and intuitive algorithms, teachable machine learning allows individuals with little or no programming experience to train their own machine learning models to perform specific tasks.

This ability to democratize the process of training machine learning models has the potential to greatly expand the number of people who can use and benefit from this powerful technology. This final report serves as a mini overview of the whole project.

Diving Deep

Without any previous knowledge on this subject, I chose the topic of reinforcement learning because games playing themselves always fascinated me. While it looks simple on one hand, it quickly became extremely complicated. All the way up to the point that I had to retire from using Windows and enter the world of Linux.

It all started out relatively easy, experimenting with the gym library and training my first model. Quickly I came to the realization that this is going to take more than just research. I started building my own library to perform the task at hand. I wanted to be able to train and run algorithm for various environments easily.

My journey into researching the various models was an interesting task, which took up much time. I resorted to selecting a smaller number of working models but focus on the environment implementation.

This proved to be a difficult task too, the environments required many libraries. And many of these libraries were not working properly. Requiring me to rewrite them myself from stack trace information.

In total the first 4 weeks of the challenge were spent on defining a proper baseline of knowledge within the field, which has been documented in the Project Document. The remainder of the project involved creating the various interfaces and training loops for environments. Lastly a small part has been dedicated to creating the Demo application, which shows off the trained environments.

Results

The results are satisfying to look at that's for sure. Many environments could be trained with the a2c algorithm, this delivered pretty good results. But the more sophisticated models could improve largely on the results.

The final product delivered is a full CLI implementation of training, evaluating, and enjoying RL models, research documentation on the topic of RL and code.

Conclusion

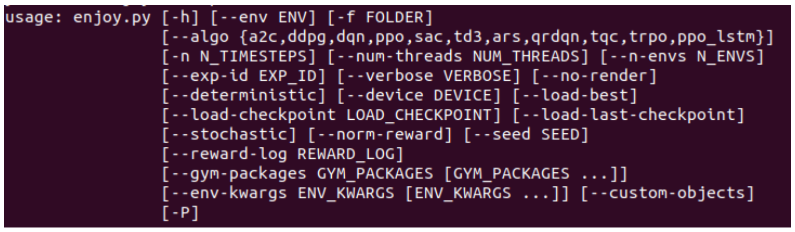
Sadly, it is obvious that a large part of my intended project is missing. Where is the teachable part? It proved to be a tough job to gather all the required knowledge and resources to manually create an app like this. The basis of the project is now set. Models and environments have been implemented. All that is left is creating a nice UI, which takes a long time.

My personal development goals are to strive towards technical ML improvement. And I do not see UX/UI design being an important piece in that.

Training RL models and learning how to interface with python libraries like Gym, PyTorch, OpenCV and an OS like Linux have taught me how to overcome issues and bugs which I have never experienced before. And I proudly managed to solve them on my own.

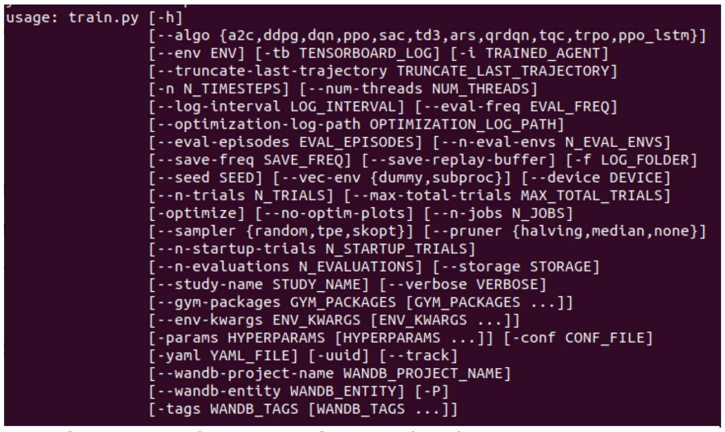
For the future I would recommend myself to take on a smaller project. But where is the fun in that? By taking on a challenging project I had something amazing to work towards. It was not easy, but it was worth the effort.

The results speak for themselves.



```
usage: enjoy.py [-h] [--env ENV] [-f FOLDER]
                [--algo {a2c,ddpg,dqn,ppo,sac,td3,ars,qrdqn,tqc,trpo,ppo_lstm}]
                [-n N_TIMESTEPS] [--num-threads NUM_THREADS] [--n-envs N_ENVS]
                [--exp-id EXP_ID] [--verbose VERBOSE] [--no-render]
                [--deterministic] [--device DEVICE] [--load-best]
                [--load-checkpoint LOAD_CHECKPOINT] [--load-last-checkpoint]
                [--stochastic] [--norm-reward] [--seed SEED]
                [--reward-log REWARD_LOG]
                [--gym-packages GYM_PACKAGES [GYM_PACKAGES ...]]
                [--env-kwargs ENV_KWARGS [ENV_KWARGS ...]] [--custom-objects]
                [-P]
```

Figure 2 Parmeters for Terminal Command 'enjoy' - Play environment with trained model



```
usage: train.py [-h]
                [--algo {a2c,ddpg,dqn,ppo,sac,td3,ars,qrdqn,tqc,trpo,ppo_lstm}]
                [--env ENV] [-tb TENSORBOARD_LOG] [-i TRAINED_AGENT]
                [--truncate-last-trajectory TRUNCATE_LAST_TRAJECTORY]
                [-n N_TIMESTEPS] [--num-threads NUM_THREADS]
                [--log-interval LOG_INTERVAL] [--eval-freq EVAL_FREQ]
                [--optimization-log-path OPTIMIZATION_LOG_PATH]
                [--eval-episodes EVAL_EPISODES] [--n-eval-envs N_EVAL_ENVS]
                [--save-freq SAVE_FREQ] [--save-replay-buffer] [-f LOG_FOLDER]
                [--seed SEED] [--vec-env {dummy,subproc}] [--device DEVICE]
                [--n-trials N_TRIALS] [--max-total-trials MAX_TOTAL_TRIALS]
                [-optimize] [--no-optim-plots] [--n-jobs N_JOBS]
                [--sampler {random,tpe,skopt}] [--pruner {halving,median,none}]
                [--n-startup-trials N_STARTUP_TRIALS]
                [--n-evaluations N_EVALUATIONS] [--storage STORAGE]
                [--study-name STUDY_NAME] [--verbose VERBOSE]
                [--gym-packages GYM_PACKAGES [GYM_PACKAGES ...]]
                [--env-kwargs ENV_KWARGS [ENV_KWARGS ...]]
                [-params HYPERPARAMS [HYPERPARAMS ...]] [-conf CONF_FILE]
                [-yaml YAML_FILE] [-uuid] [--track]
                [--wandb-project-name WANDB_PROJECT_NAME]
                [--wandb-entity WANDB_ENTITY] [-P]
                [-tags WANDB_TAGS [WANDB_TAGS ...]]
```

Figure 3 Parameters for Terminal Command 'train' - Train model on environment

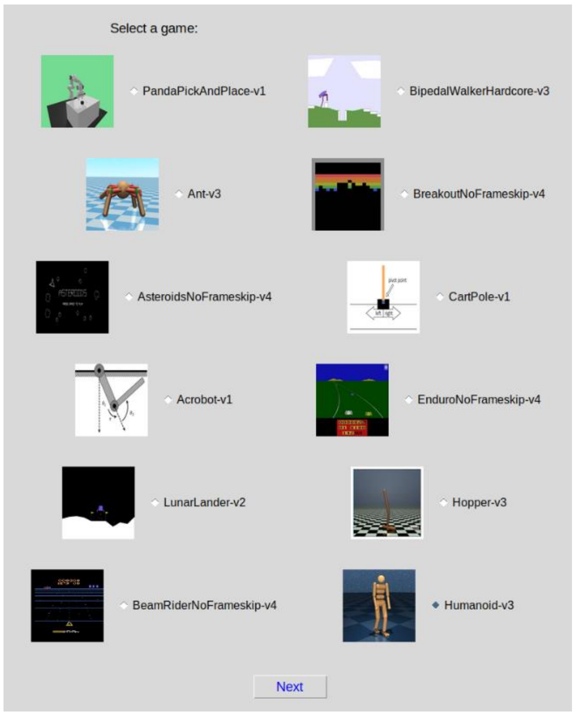


Figure 1 Demo app created implementing various environments

Algorithms & Environments

Playing
Training
Learning