Travis Waggoner

Southern New Hampshire University

CS-370

10/04/2022

Reinforce and A2C

Reinforce algorithms allow for the predictive model to stabilize and reward each step by storing the probabilities and rewarding along the way. This allows for better results where the program can find solutions faster and with more regularity. With the cartpole problem this looks like the program being able to balance the pool for longer amounts of time. Fundamentally utilizing logarithmic formulas allows for the neural network to able to save the previous result with in the iteration allowing for the overall solution to be found within the arithmetic. You can find a good reference to this in Chris Yoons pseudocode:

Function REINFORCE

Initialize θ arbitrarily

For each episode {S1, A1, R2, …., sT-1, AT-1, rT} πθ do

For t=1 to T – 1 do

Θ<-θ+α Δθlogπθ(Sr, at)Vt

End for

End for

Return θ

End function

With this function it shows how the initialization of theta in the reinforcement what will allow of the previous sections of the problems get solved. This also shows how the equation will allow for each episode to flow between saving each iteration and how it will be utilized with in the same equation. As well this shows how many steps are completed within each iteration and what value is returned.

The A2C algorithm can be implemented to help solve the cartpole problem by allowing for better stability within the program. This is done by reducing the variance the problem gives within the problem. One implementation is having the acters and critic parameters to lower the gradient policy. This can also be described in pseudocode described by Yoon with the following:

Def a2c (env):

Number of inputs

Number of outputs

Acter critic equals number of inputs, outputs and hidden size

Optimizer equals critic parameters and learning rates

For steps in range of number of steps:

# Normal steps

If done == number of steps – 1:

Quality value = critical forward into new state

Detach numpy at 0, 0

Append all sums and steps

Print

A policy gradient approach allows for smaller changes in the problem in order to make a complete solution. This also means that it can find more than one solution to the problem as it approaches the best answer. This is different from value-based approaches because of how different the reward system works with in the policy. An actor critic method changes how different the changes are over all and creates a more stable function that in theory find the answer at a faster more consistent rate.

Work Cited

Beysolow, T. (2019). 3. In *Applied Reinforcement Learning with python: With Openai Gym, tensorflow, and keras*. essay, Apress.

Gulli, A., & Pal, S. (2017). 8. In *Deep learning with keras: Implement neural networks with Keras on Theano and tensorflow*. essay, Packt Publishing.

Yoon, C. (2019, July 17). *Understanding actor critic methods*. Medium. Retrieved October 5, 2022, from https://towardsdatascience.com/understanding-actor-critic-methods-931b97b6df3f

Yoon, C. (2019, May 23). *Deriving policy gradients and implementing reinforce*. Medium. Retrieved October 5, 2022, from https://medium.com/@thechrisyoon/deriving-policy-gradients-and-implementing-reinforce-f887949bd63