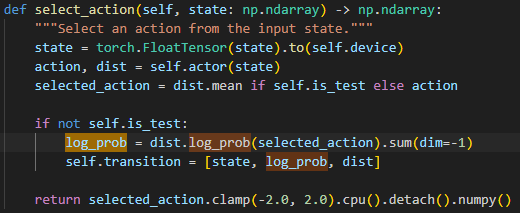
1. Introduction

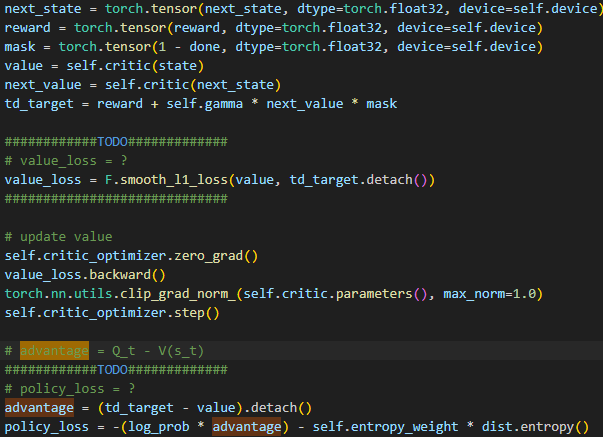
本實驗探討在強化學習中使用A2C和PPO演算法的實現與效果，並將其應用在Pendulum與Walker2d環境進行實驗。在Task1當中為基本的A2C實作，而Task2和Task3詳細說明PPO在Pendulum和Walker2d環境中的實現，同時包含GAE估計等關鍵技術。並且在最後顯示所有的實驗模型表現。

1. Implementation
   1. Stochastic policy gradient and the TD error



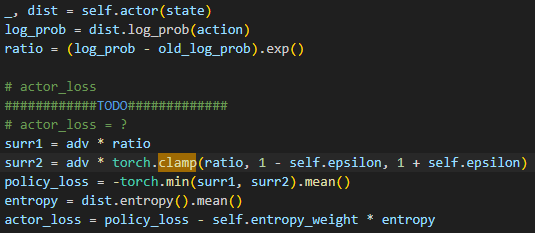


Stochastic policy gradient通過Actor輸出平均值和標準差，並採樣動作以計算log\_prob作為策略梯度的一部分。



TD error透過Critic預測的V(St)和 reward + gamma \* V(St+1)之間的誤差計算得來。

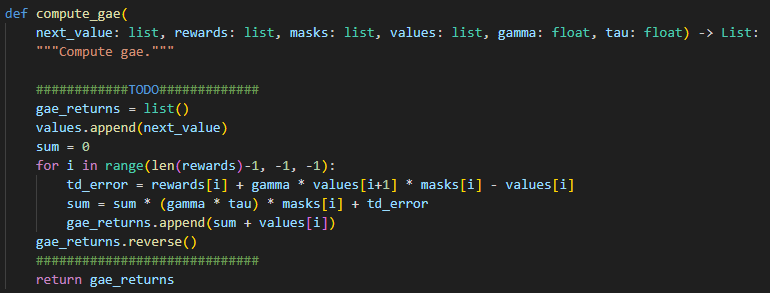
* 1. Clipped objective in PPO



使用torch.clamp將新舊策略比率限制在[1-ε, 1+ε]範圍內，然後乘上advantages

並與未做clamp的的版本取min再取負號，作為policy\_loss用來更新Actor。

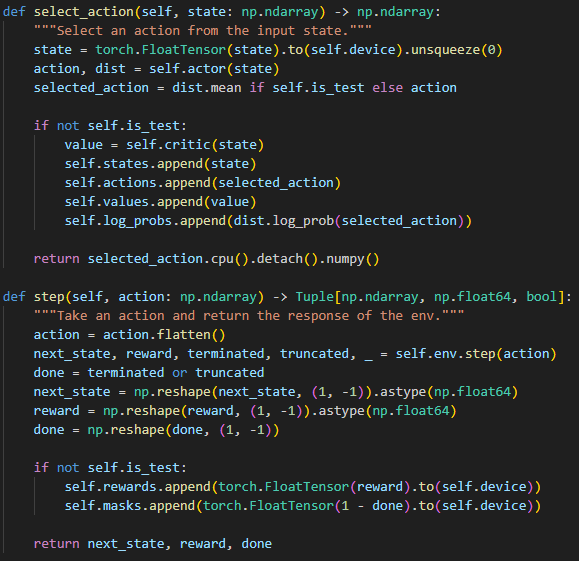
* 1. Estimator of GAE



透過compute\_gae來取得其估計值，公式為GAE = δt + gamma \* τ \* mask \* GAEt+1

其中δ為TD error。

* 1. collect samples from the environment



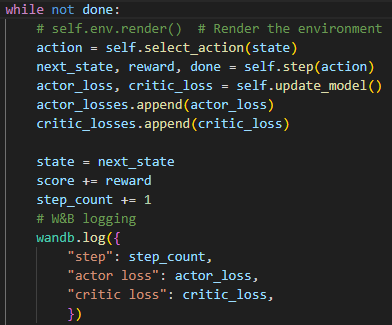
使用step取得環境的所有資訊，並透過Action和Critic模型取得當前的action和value並全部記錄起來以方便做PPO的模型更新。

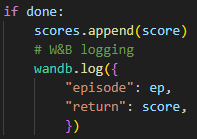
* 1. enforce exploration

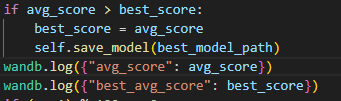


在Action模型的loss計算中，多增加了entropy的loss，-entropy\_weight \* dist.entropy()，以此來鼓勵Action輸出的std不要太小，藉此保持搜索的力道。

* 1. Weight & Bias







使用wandb.log 記錄每步的actor loss、critic loss、reward、episode數等指標。除此之外，在每一次episode之後執行test並且也將test的分數和過往20場的平均分數記錄在wandb當中以便觀察。

1. Analysis and discussions
   1. training curves
      1. Task 1

一張含有 文字, 筆跡, 行, 字型 的圖片

AI 產生的內容可能不正確。

一張含有 文字, 螢幕擷取畫面, 行, 繪圖 的圖片

AI 產生的內容可能不正確。

一張含有 文字, 行, 螢幕擷取畫面, 繪圖 的圖片

AI 產生的內容可能不正確。

* + 1. Task 2

一張含有 文字, 圖表, 行, 繪圖 的圖片

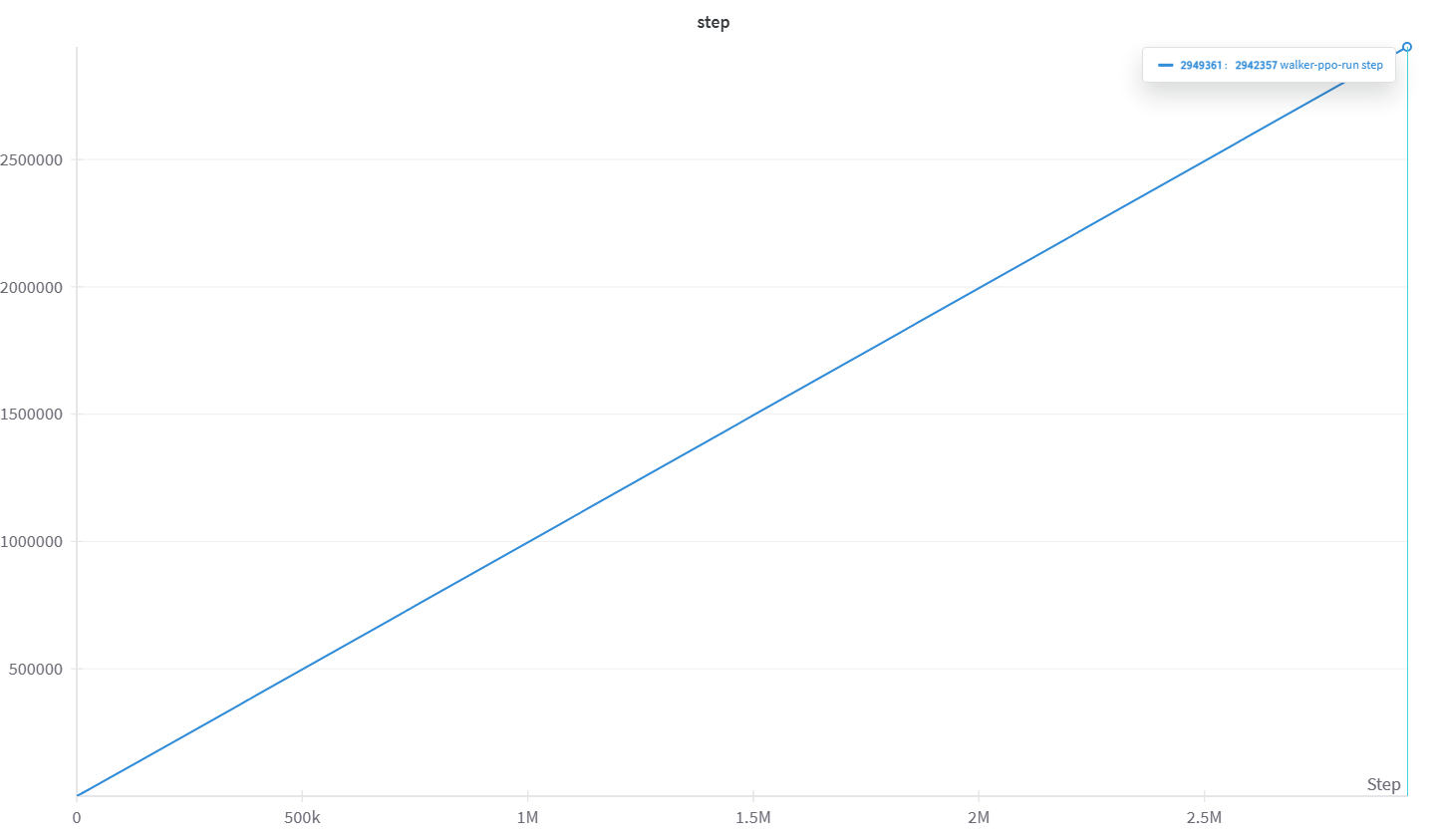
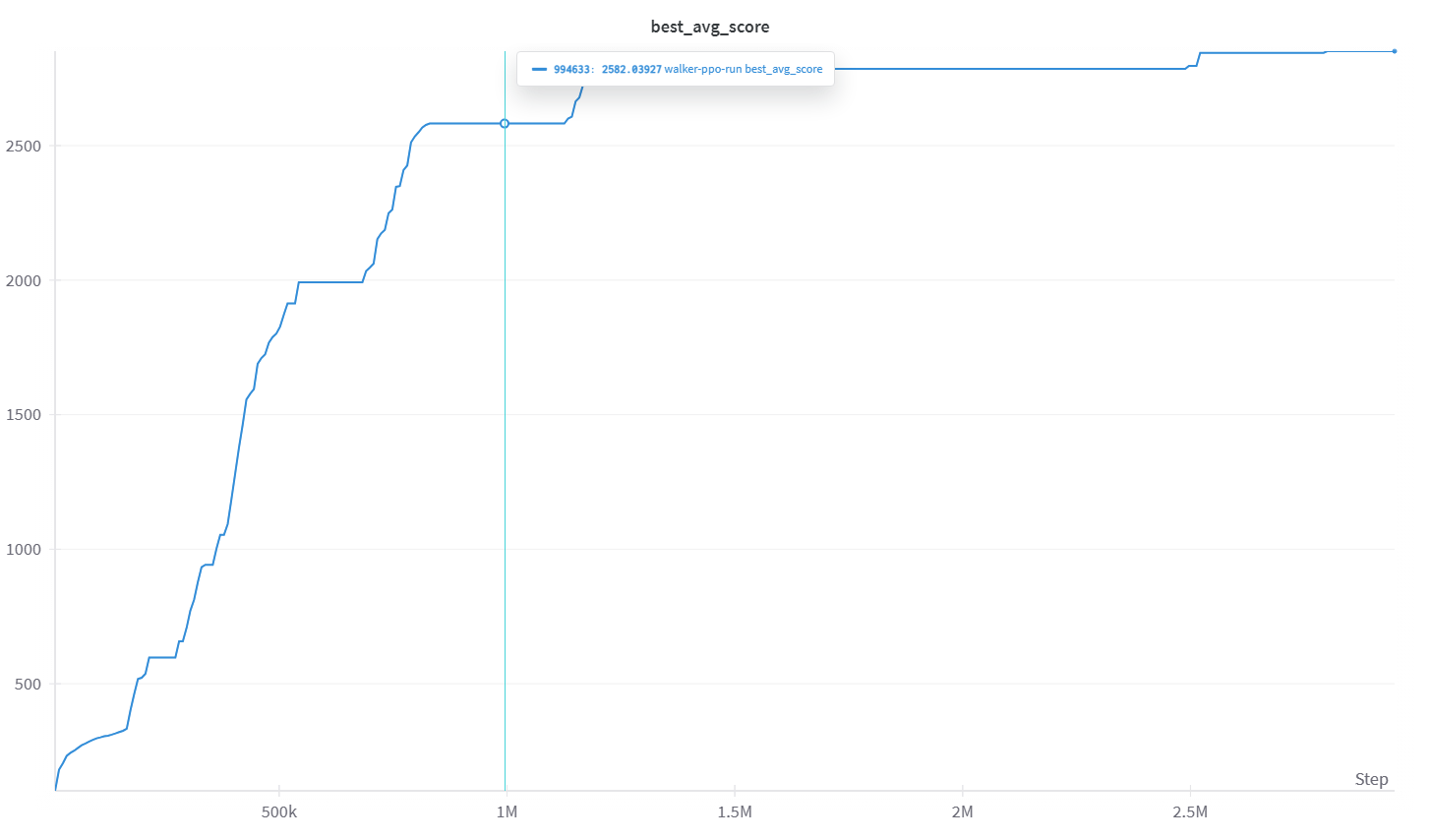
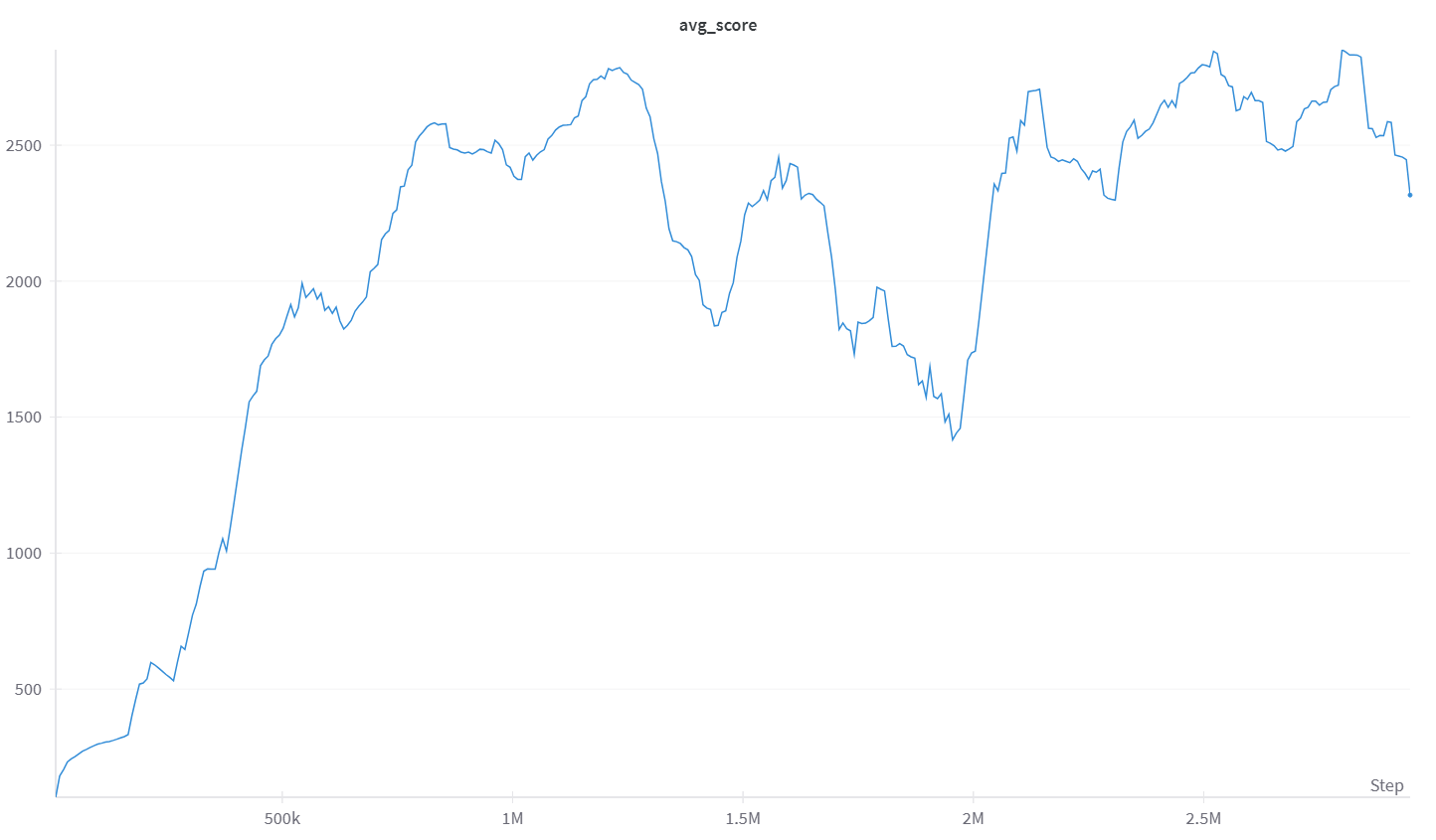
AI 產生的內容可能不正確。

一張含有 文字, 螢幕擷取畫面, 行, 繪圖 的圖片

AI 產生的內容可能不正確。

一張含有 行, 螢幕擷取畫面, 文字, 繪圖 的圖片

AI 產生的內容可能不正確。

* + 1. Task 3
    2. Conclusion

Task 1、Task 2可以在 200K 的environment step之內達到平均-150分以上。

Task 3可以在1M之內達到平均2500分以上。

所有Task所使用的seed:

[4140, 5339, 3232, 3940, 3164, 1885, 4789, 7802, 9140, 3896, 2383, 9107, 8202, 39, 4586, 464, 8145, 2829, 3133, 8311]

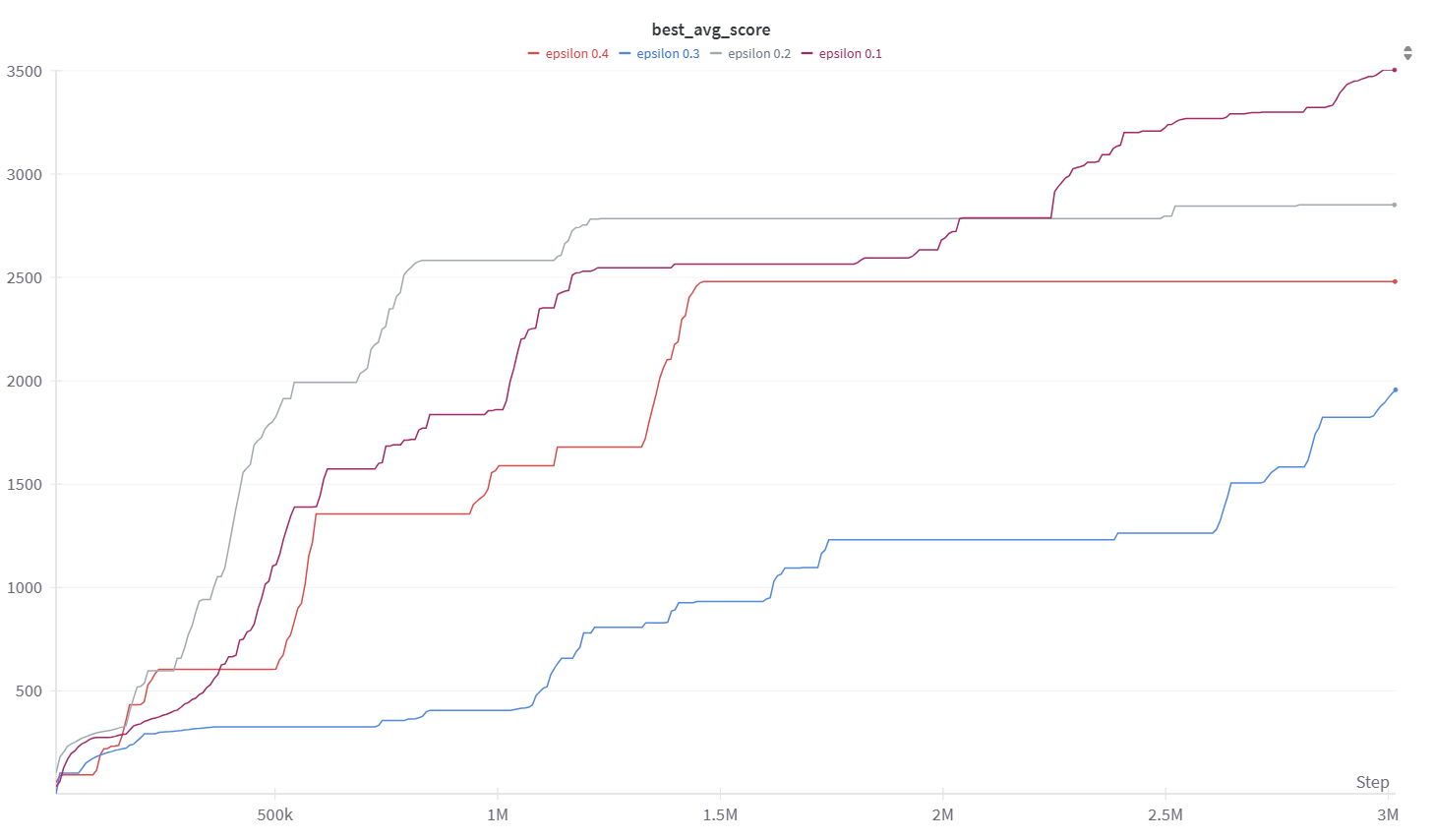
* 1. Compare the sample efficiency and training stability of A2C and PPO
     1. Sample efficiency

PPO透過多次更新update\_epoch=10以重複利用sample，尤其在Walker2d較難的環境中表現更好。A2C即時更新因為樣本利用率低，每次只會使用一次用來更新模型，適合較簡單環境，但效率較差。

* + 1. Training stability

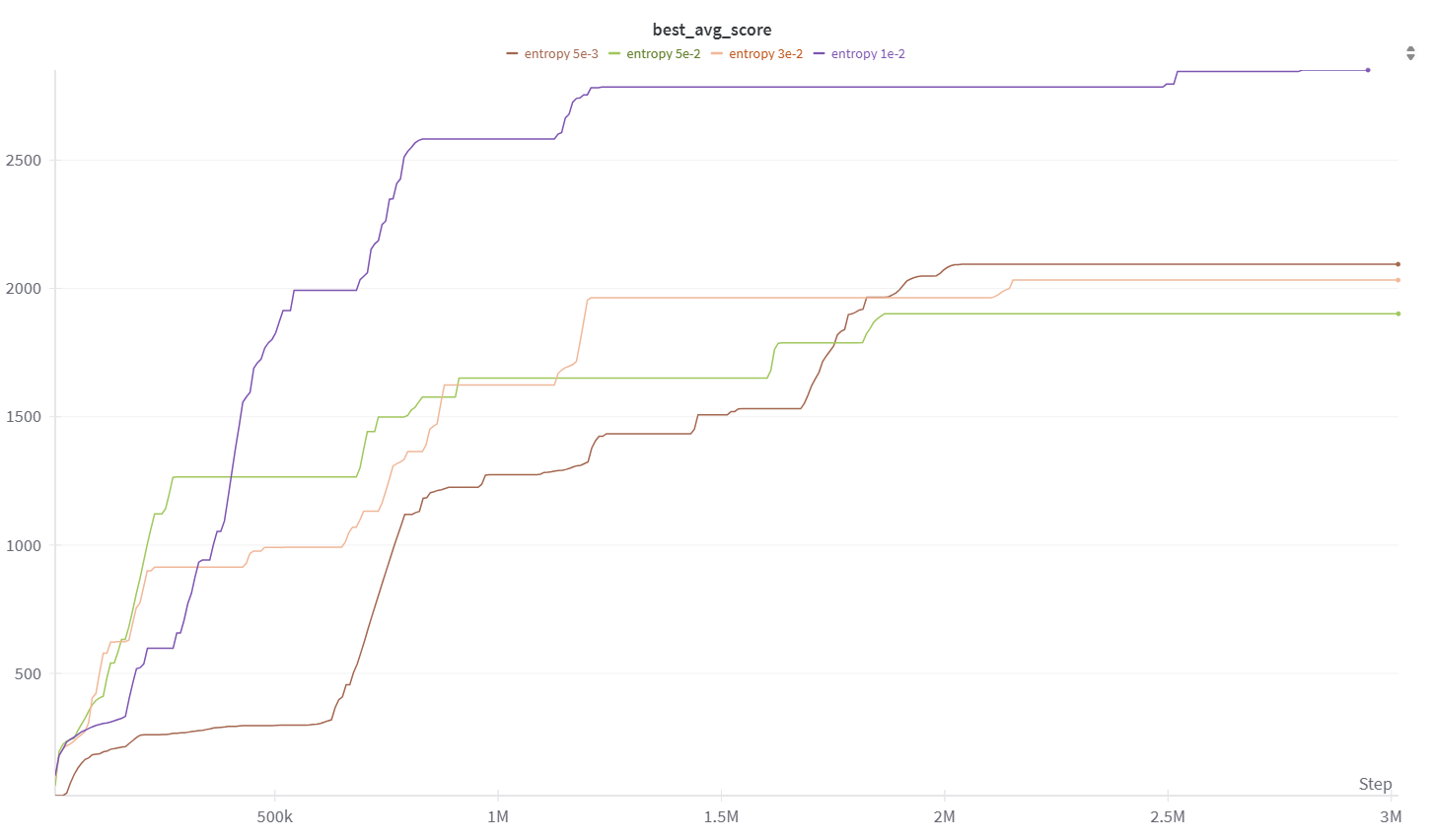
PPO的訓練穩定性優於A2C，通過限制新舊策略比率在[1-ε, 1+ε]之間和GAE估計，控制模型的更新幅度，因此表現會較於平滑。而A2C因為每次會直接使用剛剛取得的sample做更新，在模型的更新上波動較大。

* 1. Empirical study
     1. Clipping parameter



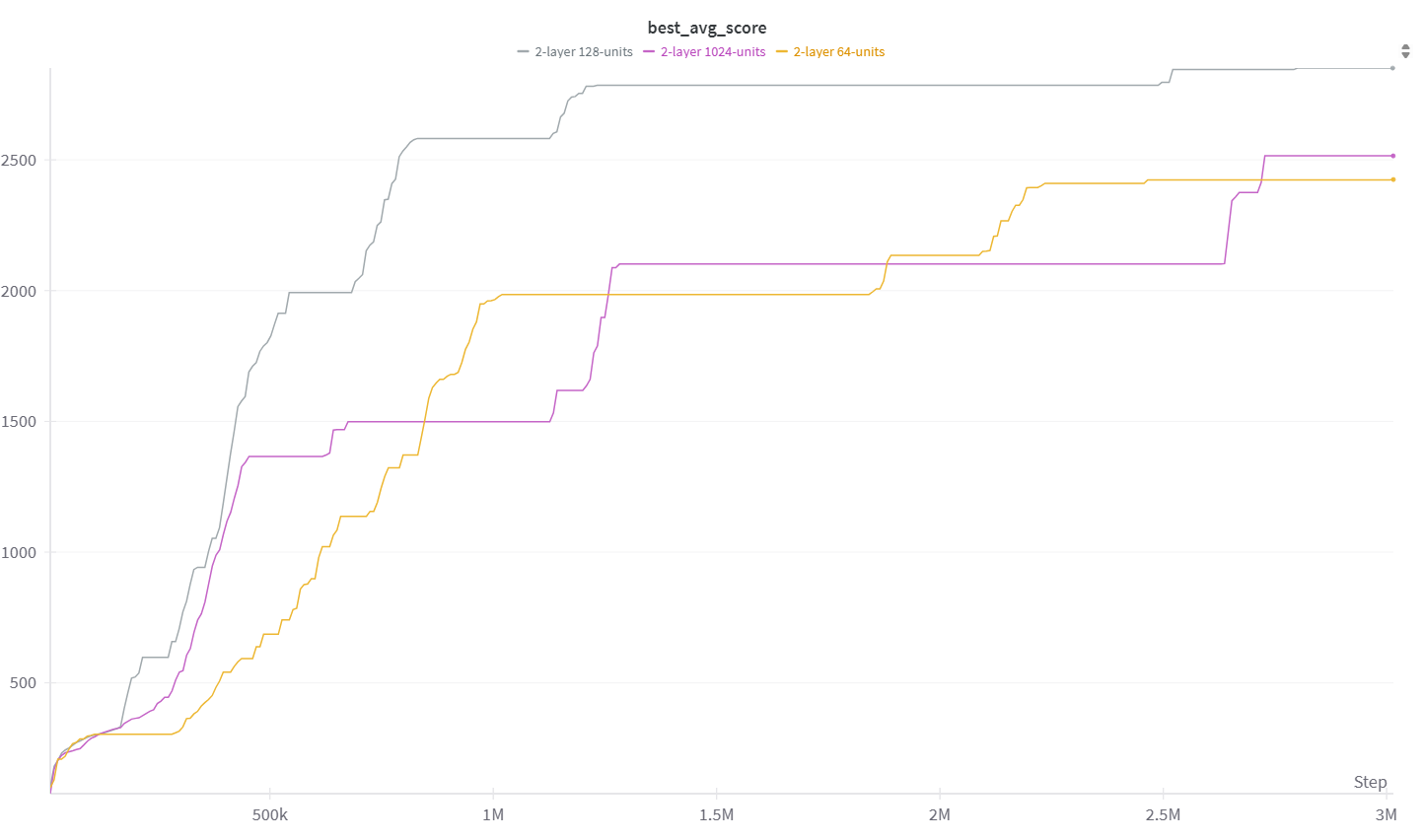
在所有的hyper parameter都不變的情況下，只更改epsilon (Clipping)的參數，可以發現對於Lab的要求，使用epsilon為0.2的參數在1M的environment step之前表現最好，因此選用此超參數作為最終模型的訓練，但是epsilon為0.1的實驗中，在後期能夠達到3500的高分，顯示若只論模型表現的情況下，將epsilon設為0.1對此環境較為適合。

* + 1. Entropy coefficient

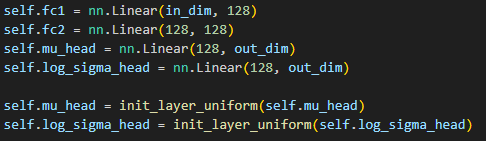


根據先前Clipping parameter的實驗，選用epsilon為0.2為基礎，其他參數皆不便的情況下，只更改entropy weight可以發現，將entropy weight設為1e-2的表現效果遠高於其他版本。

* 1. Additional analysis



在Task3中調整超參數的時候，考慮過是否是模型的複雜度太小造成初期的實驗效果不佳，後來經過比對後發現，若模型太過複雜如1024個units，表現上會非常不好，極有可能是overfitting的問題，在未來可以考慮加上dropout的設計，反之模型太過簡約也會影響效能。透過實驗比對之後才選定以2層基礎layer(fc1,fc2)各128個units當作模型主要架構。



1. Execute code
   1. Training

python a2c\_pendulum.py

python ppo\_pendulum.py

python ppo\_walker.py

* 1. Testing (and generate video)

python a2c\_pendulum.py -p “model path”

python ppo\_pendulum.py -p “model path”

python ppo\_walker.py -p “model path”