Title

Investigating Emergent Strategies in an Iterated Prisoner’s Dilemma Environment With the use of Machine Learning.

Introduction

My exploration delves into the ethical dilemmas that emerge, within the realm of game theory. the one I chose is the Iterated Prisoner’s Dilemma (IPD). While the classic Prisoner's Dilemma (PD) involves a single round, the IPD extends the game to multiple normally random number of rounds, allowing for the evolution of strategies over multiple rounds. The Classic example of the Prisoner’s dilemma is two prisoners A and B are in separate rooms, they individually have the choose to either defect or cooperate. If prisoner A choses to defect while prisoner B chose to cooperate then prisoner A will only get a 1-year sentence and prisoner B will get an 8-year sentence and vice versa. If both chose to defect, they will get a harsher sentence of 5 years each. If they both choose to cooperate, they will get a 2-year sentence each. This dilemma is about the conflict between mutual benefit and individual gain.

There was a lot of research regarding IPD in 1980 – [Effective choice in the Prisoner’s Dilemma] which highlights different strategies. One strategy I will be focusing on is the Tit for tat strategy which never scores higher than its opponent but wins overall.

Reinforcement learning (RL) is a variant of machine learning that requires an agent to decide and observe in an environment, the agent will be rewarded or punished based on the decisions it makes. This allows the agent to create a strategy (policy) that will prioritise decisions that maximize the reward it receives. I will be using Proximal Policy Optimization (PPO) for this experiment. PPO is a type of RL that is simpler to implement and reliable – [Proximal Policy Optimization algorithm)]. This was the chosen algorithm since it was supported with Unity Machine Learning Agents- [Unity ml agents] which I used in an engine called Unity-[Unity] to create the environments.

In this paper I will be exploring the strategies that emerge using machine learning in the Iterated Prisoner’s Dilemma (IPD) environment. I will be comparing these strategies to already documented strategies with a focus on the Tit for tat strategy seeing how the agent will work against it and if we can force the agent to develop the Tit for tat strategy by changing its environment and establishing the requirements for this strategy to emerge. Changing the environment will also serve the purpose of observing if it will have any major impacts or create any patterns in the agent.

Tit for tat (TFT) strategy is thought to be one of the best strategies as on a very simplified version of IPD which is the version I will be making the environment into. When randomness is added to the strategies to simulate what would be a misunderstanding or mistake in the real world, the Tit for tat strategy does not do so well. TFT works by always choosing to cooperate unless the opponent chose defect on the previous turn. This means that TFT only works well with other cooperative strategies.

I will be conducting the experiment in three parts. Each part will have the AI go against a different opponent. The reason for this is to give the agents a lot of different chances to see if there are any difference between the opponents or if there will be one pattern to emerge that will top all three opponents.

The first opponent will be an opponent that will choose defect or cooperate at random with the odds being 50/50 for each. This will serve as a sort of introduction and help me get the hang of the Agent.

The second opponent will be itself I will have it train whilst its going against itself this one I have the most hope that it will produce unique results.

The third opponent will be the Tit for tat strategy. I attempted to and the environment to see if we can force the agent to develop the Tit for tat strategy, this did not work, however.

my exploration delves into the ethical dilemmas that emerge, within the realm of game theory. the one I chose is the Iterated Prisoner’s Dilemma (IPD). While the classic Prisoner's Dilemma (PD) involves a single round, the IPD extends the game to multiple normally random number of rounds, allowing for the evolution of strategies over multiple rounds. The Classic example of the Prisoner’s dilemma is two prisoners A and B are in separate rooms, they individually have the choose to either defect or cooperate. If prisoner A choses to defect while prisoner B chose to cooperate then prisoner A will only get a 1-year sentence and prisoner B will get an 8-year sentence and vice versa. If both chose to defect, they will get a harsher sentence of 5 years each. If they both choose to cooperate, they will get a 2-year sentence each. This dilemma is about the conflict between mutual benefit and individual gain.

There was a lot of research regarding IPD in 1980 – [Effective choice in the Prisoner’s Dilemma] which highlights different strategies. One strategy I will be focusing on is the Tit for tat strategy which never scores higher than its opponent but wins overall. It is a strategy that brings the out the highest score that both itself and its opponent strategy can obtain.

With AI having shown a lot of progress recently, I decided to look at dilemmas. There are many popular dilemmas but the one I chose is the Iterated Prisoner’s Dilemma (IPD) which normally is a one round game called the Prisoner’s Dilemma (PD) the IPD is a version that consists of multiple rounds of the same game. This means that strategies can be developed and evolved during the game. There was a lot of research regarding IPD in 1980 – [Effective choice in the Prisoner’s Dilemma] which highlights different strategies. One strategy I will be focusing on is the Tit for tat strategy which never scores higher than its opponent but wins overall. It is a strategy that brings the out the highest score that both itself and its opponent strategy can obtain.

Reinforcement learning (RL) is a variant of machine learning that requires an agent to decide and observe in an environment, the agent will be rewarded or punished based on the decisions it makes. This allows the agent to create a strategy (policy) that will prioritise decisions that maximize the reward it receives. I will be using Proximal Policy Optimization (PPO) for this experiment. PPO is a type of RL that is simpler to implement and reliable – [Proximal Policy Optimization algorithm)]. This was the chosen algorithm since it was supported with Unity Machine Learning Agents- [Unity ml agents] which I used in an engine called Unity-[Unity] to create the environments.

In this paper I will be exploring the strategies that emerge using machine learning in the Iterated Prisoner’s Dilemma (IPD) environment. I will be comparing these strategies to already documented strategies with a focus on the Tit for tat strategy seeing how the agent will work against it and if we can force the agent to develop the Tit for tat strategy by changing its environment and establishing the requirements for this strategy to emerge. Changing the environment will also serve the purpose of observing if it will have any major impacts or create any patterns in the agent.

Predictions/Hypothesis

Agent vs Random

My prediction is that the Agent will eventually reach a state where it will only defect. I believe this will be the case since the opponent chooses at coop or defect at 50/50 odds. If we assume it will just alternate between the two, the agents will score on average 1 point per turn if it chooses to cooperate. It will gain an average of 2 points per turn on average of it chooses to defect. So, it will default to defecting since that scores the higher point per turn on average.

Agent vs Agent

If the agents were rewarded on cumulative score rather than individual, then I predict it will reach a state of only cooperating. However, I will be rewarding the agents individually, this will already create a bias since I’m not asking the AI to weigh up the value of individual versus mutual benefit, I am simply telling it to gain the biggest induvial score. So, I believe this will take the longest before it reaches Nash equilibrium and that being both agents choosing to defect. I believe it will take a very long time since it will start of as random choosing defect or cooperate and like the agent vs random prediction it will realise that this will output more per average on a turn. I do believe that on very rare occasions the AI can be trained and will reward Cooperation for a while, but I do not believe that this will last.

Agent vs Tit for tat

I believe that this will quickly reach Nash equilibrium with its choice being to permanently cooperate. However, since there are always going to be 50 rounds, I can see the Agents remembering to defect on the final round. This is a flaw in the experiment.

Now I needed to setup the Agent. Its PPO RL agents so it needs to have observations. The observations I gave it was a float array with 50 elements this array represents the previous choices its opponent made being 0 for cooperate and 1 for defect, for all the undecided or yet to come rounds in the array they were displayed as -1. The -1 is a flaw in the experiment as its additional data I did not want it to have access to as it can see the number of rounds that will be played.

The Agent also needs to be told what decision it can and can’t make. Since it will be a round based game where the Manager will request a decision from the Agent, this makes my environment discrete rather than continuous.

The fourth step was to setup each of the experiments and use the UMLA package to train the agents in each of the environments. So I configured the PPO Agent algorithms parameters, The agent will have only two options it can make, cooperate or defect. I set this in the Behavior Parameters script that the UMLA package contains

I ran this experiment separately but had multiple of the same experiment training alongside itself to speed up the training process.

For the environment I used Unity Engine version 2022.3.2f1 – [Unity reference] for the PPO I used Unity ML agent’s version 21 – [Unity ML agents’ reference] the ML agents used python 3.10.12.

After getting all the software set up, I created my environment the structure was I would have a manager that would handle all the calculations and order that everything happens in, it will also calculate the reward that the agents should receive. I then had the Agent which is the bare minimum using the Unity ML agents’ package to function. This is where the decision would be accessible which then get sent to the manager. The last script I created is Opponent this was different for each of the three experiments.

(Insert UML chart)

Agent Parameters

The agent is given access to observe the opponents’ previous choices as an array of floats. 0 means it cooperated, 1 means it defected. For errors or decisions that haven’t been made yet the array represents it as -1.   
The agents’ choices can only be one of two outputs 0 or 1 which gets translated to Cooperate or defect respectively.

First Experiment Agent Vs Random

The structure for the AI and manager is the same as stated above but the opponent will Choose at random to defect or cooperate at 50/50 odds. Which is then sent to the manager where the score will be calculated, and the rewards given to the

Introduction (Sub)

In this paper I am investigating the use of Reinforcement learning in dilemmas the chosen dilemma is IPD this is a common dilemma that appears often in everyday life examples range from something as simple as grocery shopping with friends to Countries and their nuclear weapons. This experiment aims to see if there are common or re-emergent patterns that the RL produces. I will also be trying to force a certain pattern to emerge by changing the environment. I will be doing this by placing an AI against Three different opponents in the IPD and seeing what strategies emerge for each of the opponents.

Software Specifics (Sub)

To get started you need to acquire all the correct software Being Unity Engine version 2022.3.2f1 which is where the environment will be built. Unity ML Agent version 21 (UMLA) which is a package for unity that offers a lot of agent algorithms, but we will only be using PPO algorithm for this experiment, Unity ML agents require Python 3.10.12 to function.

Environment creation in Unity(sub)

An agent needs and environment to train in. I create this in Unity with a simple visual scene that has two blue cubes to represent the Agent and Opponent. Both cubes have two smaller red and green cubes attached to them these are invisible by default. These two smaller cubes are meant to represent their choices to cooperate (Green) and defect (Red). When the Choices are made these cubes become visible based on the choice that was made. That was all I created for the visual side of the experiment, this helps in identifying what choices both the agent and opponent chose.

(Possible image of scene)

Program and structure(sub)

The third step was to create the Scripts that will allow the UMLA package to train and communicate with our scene. I decided to go with a setup that will contain a manager that handles all the calculations of who won and the rewards respectively, The Manager also organises the order that everything is done, and when the AI should make a decision. The Agent Script which is simply just the bare minimum for the UMLA package to start training, it contains the ability to make a decision and collect observations. The last script is an opponent script which for the first experiment acted like an agent but just chose at random whether to cooperate or defect, the goal for the first experiment was to set a baseline and establish the best choice for the agent in an PD game where there is no indicator for what the opponent will chose. In the second experiment We simply just use the agent as the opponent so this will have two agents training simultaneously thanks to UMLA this was very simple to accomplish. The third experiment I recreated the Tit for tat algorithm as the opponent, this experiment is where I will be attempting to force the Agent to adopt a Tit for tat pattern.

(Insert UML diagram)

Setting the Parameters (sub)

The environment I create will decide what parameters I give the Agent.

Partially observable -due to the Agent not knowing what its opponent will chose in the future.

Multi agent – Since it has opponents that it will face off against.

Competitive- The agent will be trained to score its highest score rather than highest cumulative score.

Deterministic and stochastic – the first experiment will be stochastic due to its opponent’s randomness but second and third experiments are deterministic.

Static- Since the environment only changes every round.

Discrete- it’s a round based game, so it is all discrete rather than continuous.

For the Agents Observations I give it an Array of float with 50 elements this is array will represent the opponents’ previous choices being 0 for cooperate and 1 for defect, For the Choices that are yet to be decided I will make them -1 because my environment is meant to be partially observable since the agent should not have access to future or yet to be decided moves of its opponent. The -1 is a flaw in the experiment as its additional data I did not want it to have access to as it might be able to use this to learn the number of rounds it will be playing and defecting on the last round by default. We choose 50 since the Agent will be playing in games that contain 50 rounds.

The Agent needs to make decisions since the environment is discrete it can only choose between either cooperate or defect. This makes setting the behaviour parameter provided by UMLA easy I set the Discrete Branch to 1 since we only want it to make one decision and the branch size of the branch to 2 since we want it to choose between 2 options. The agent decision is now outputted in the Agent Script as either a 0 or 1 which I associate to cooperate and defect.

The Agent should also be rewarded to learn what it is doing correctly. I set my rewards as 0,1,2,3 To keep simplicity. The conditions for which rewards get applied when are down below in fig(2) which I obtained from this paper-[figure paper]

A white rectangular grid with black text

Description automatically generated

The PPO algorithm has a lot of other parameters stored as a .yaml the main parameter I changed in this is max timesteps or “max\_Steps” this means that once the Agent has reached the specified number in max timesteps then it will automatically stop and save what it has learnt. I use this as a training length on all three experiments and decided at 200,000 steps since that is all I had time for on my hardware.

Measurements

I will measure the agent’s strategy once it reaches a Nash equilibrium I will then check to see if it is just decided to repeatedly choose to cooperate or defect which is what I would consider a failure, or hopefully it develops a different strategy where the choices its opponent makes will influence its choice.

Method Conclusion (Sub)

Getting all the necessary software is important. Then creating an environment even if it is not a visual one. Making sure that the Agent Parameters are set correctly as if this is incorrect it will either give you skewed results or not work completely.

Background maths

The full equation for PPO stated in OpenAI’s Paper – [OpenAI PPO] is.  
LCLIP +V F +S t (θ) = ˆ Et [LCLIP t (θ) − c1LV F t (θ) + c2S[πθ](st)],

Where:

* LCLIP(θ) is the clipped surrogate loss.
* LV F t(θ) is the Squared-error loss for the value function.
* S[πθ](st) is the entropy bonus.
* ˆ Et denotes the empirical expectation over collected samples.
* c1 and c2 are coefficients.

Generalized Advantage Estimation formula in OpenAI’s paper is.

At = δt + (γλ)δt+1 + · · · + · · · + (γλ)T −t+1δT −1,

where:

* At is the estimated advantage at time step t.
* δt is the temporal difference error at time step t.
* γ is the Discount factor.
* Λ is a parameter that balances bias and variance in the advantage estimation.
* T is the total number of time steps

The formula for temporal difference error is.

δt = rt + γV (st+1) − V (st)

where:

* δt is the temporal difference error at time step t.
* rt is the reward at time step t.
* γ is the Discount factor.
* V (st+1) is the estimated value of the next state.
* V (st) is the value of the current state.

This study investigates the use of Reinforcement Learning (RL) in the context of the Iterated Prisoner’s Dilemma (IPD) and Prisoner’s Dilemma (PD). “The PD is one of the most fiercely debated thought experiments in philosophy”-[Prinsoner dilemma, martin peterson]. PD appears in everyday situations like grocery shopping to global issues such as nuclear war. The goal is to identify common RL-generated patterns and identify emergent strategies and their environments.

First Experiment: Agent Vs Random (Sub)

In Experiment One we have the agent go against an opponent that chooses to defect or cooperate at random. Since there is randomness involved, I decided that this experiment should be the one to be reran this helps with reducing the chance that probability skewed the results. From Figure [Graph fig] we can see that they both reached the best possible mean score of 2 with the first test being 0.015 behind which I have assumed is due to randomness. The first test is faster than the second test to reach its desired score. This means it developed an Always Defect strategy – [Axelrod].

Second Experiment: Agent Vs Agent (Sub)

The Second Experiment was the Agent against itself it will learn, both agents were getting rewarded separately but the training was being done together so they were each reward for individual reward rather than cumulative reward. The start of the experiment was a lot of back and forth between trying to figure out whether cooperate was good. After only around 20k steps the agent reach the conclusion that defect was the correct choice. This means it reached the Always Defect strategy.

Third Experiment: Agent Vs Tit for tat (Sub)

In the final experiment where it’s the agent against Tit for tat strategy It concluded quickly that cooperating was the best choice. scoring 99 in 50 rounds which is an average score per round of 1.98. this score is nearly the best it can achieve the highest score it can achieve is 101 if defects on only the last round. This means it developed the Always Cooperate Strategy.

Analysis(Sub)

First Experiment: Agent Vs Random(SubSub)

The First experiment was used as a test to set the bar for what to expect from the other two experiments. I reran this test twice to attempt to remove and chance of randomness from skewing the results. Ideally, I should rerun this more than twice so there is still a possibility that these results are due to randomness and the actual result could be a lot different. However, the results line up with my predictions that it should reach Nash Equilibrium where it only chooses to defect. I did assume it would reach this point much faster than it did.

So overall I would count the first experiment as a success. It served as an introduction to the other two experiments and tested that the environment works. It also reached the predicted outcome.

Second Experiment: Agent Vs Agent (Subsub)

The second experiment was to place two of the agents against each other and train them simultaneously. The highest rewarding outcome for this would be if both the Agents cooperated. Since I set the agents to be rewarded individually the outcome, I predicted that both the agents would reach a state where they only choose to defect. Which is exactly like the results showed. However different from my prediction the agents reached this conclusion quickly at around only 20,000 steps. This is much earlier than I thought and surprised me, I did not have time to rerun the test. So, these results I am not taking as 100% accurate. The agents were both rewarded on individual score rather than group score but even so would the highest scoring AI in this scenario would be on that Cooperates rather than both defecting. This I believe is due to them both being trained simultaneously as we learnt from the first experiment defecting will score higher on average so they will most likely already be leaning towards choosing defect early on. Once both the Agents are already choosing to defect, they will reach a state where they might try to experiment with both cooperating however if they do, they will get punished for it by the other agent. This will give the other agent even more points only reinforcing it to choose to defect rather than cooperate. For the agent that chose to cooperate it will end up with no points also reinforcing it to choose defect over cooperate because at least then it would score 1 point minimum. I do believe that eventually with enough chances the Agents might eventually both randomly try to cooperate again at the same time, and both learn that it rewards more than both choosing to defect. However, I believe the chance of this is very slim and would take a lot of training for it to occur.

Third Experiment: Agent Vs Tit for tat(subsub)

The third experiment I was hoping for the agent to adopt the Tit for tat strategy. This result would be very unlikely though since its only rewarded based on its score rather than what strategy it develops. The way Tit for tat functions is it will reward any opposing strategy that wants to cooperate with it. This means that the Agent will most likely develop the Always Cooperate strategy- [Alexrods]. With a flaw in the experiment, it could also possibly learn when the last round is, this means it could learn to defect only on the last round since its actions would have no consequences. However, it did not develop this strategy and only chose to cooperate. This result was expected it reached it incredibly quickly at around the 7,000-9,000 steps range, much faster than any other experiment. I wanted to see what strategy would reward developing a Tit for tat strategy rather than an always cooperate strategy, I concluded that the only way to reliably obtain the desired result would set up the environment as a tournament against a lot of other strategies rather than single games against three strategies.

conclusion

The results mostly follow my prediction with the second and third experiment reaching my prediction faster than I thought it would. This experiment I would consider a success in the fact that my result did produce Strategies, but they only produced the always cooperate and always defect strategy which are not the most desirable strategies I was looking for. In terms of the third experiment, I would consider the experiment a failure since it did not produce the desired results. The desired results would be a Tit for tat strategy or a variant of it like Tit for two tats strategy – [Axelrod’s]. I believe these results did not appear due to the incorrect environment; however, it did produce predicted results of adopting an Always Cooperate strategy. My experiment results will have a negligible impact on research in this field since they are not substantial enough.

LESPS

There are many issues relating to AI but for this I will focus only on issues relating to my experiment. First to address is legal. Depending on strategies that the AI develops it could lead to people following these strategies to avoid or get out of the Justice System. My experiment did not produce good enough results to make this an issue.

Social and ethical, issues since my experiment is using a dilemma which for human means deciding between two or more difficult choices. The PPO agents see this more as a challenge to get the highest score. So, the Agent is not working out how to solve the dilemma but rather how to get the most rewards. This means that the strategies will not always provide the right ethical or moral decision. This can be seen with the trolley problem, the AI will most likely be rewarded based on how many lives it saves, but we know from humans taking the experiment that we do not always come to the same conclusion especially when it requires the humans to take a life to save more.

Professional and sustainability, Job security and AI have been a large topic with a lot of people worried about their job being replaced by AI. For this experiment if my results were substantial there could be a concern with Jobs being replaced or replaced partly that require decision making, In extreme case a job like a judge. This would cause a lot of concern and link back to the Legal area of this experiment.

Lesps

My experiment revolves around the Prisoner’s dilemma and the application of Ai to make decisions. This application of AI raises multiple issues, one of which is the legal issue of responsibility. “A Circus tiger can learn tricks from the tamer, but nevertheless stays a predator. If such a creature would hurt or kill the tamer, it is not to blame, as it is a part of its nature.”- [Ethical and legal responsibility for Artificial intelligence]. This can be used as an analogy for AI as AI cannot be blamed for its decisions due to the nature of it. This prompts debate about where responsibility for AI decisions should lie.

Another legal concern is copyright and data protection. Many AI systems rely on datasets that require human fine tuning or creation to ensure dataset is accurate. However, the people working on the datasets often face precarious labour conditions and are paid as little as $1.46 per hour –[The exploited labor behind AI]. This Issue extends beyond legality and encompasses ethical, social and sustainability issues. The labour conditions and low wage are believe to contribute to the carbon effects-[Higher wager for sustainable development?]. The cost to train AI models is a considerable concern, with estimates for a BERT model ranging from $2.5k to $1.6m –[The cost of training NLP models].

Future works

My results will not have any impact on or use in the future. If my results were substantial enough, for example if a new strategy that is the best of the best in any situation for the prisoner’s dilemma emerged from my experiment; Then it might end up getting used to settle situations like prisoner’s dilemma which spans from something simple as grocery shopping with friends to countries de-arming their nuclear weapons. It could also end up having an impact on the Justice system and how the trials are gone about if humans are not required and the “best” decision can be made without them. However, these impacts and results are both very unlikely to happen in the foreseeable future.