ARTIFICIAL INTELLIGENCE IN VIDEO GAMES – MIMICKING HUMAN BEHAVIOUR

By Denzel Guma - 9804324

JUNE 3, 2022
COVENTRY UNIVERSITY
Bsc in Computer Science
School of Computing, Electronics and Mathematics

6001CEM Declaration of originality

I Declare that This project is all my own work and has not been copied in part or in whole from any other source except where duly acknowledged. As such, all use of previously published work (from books, journals, magazines, internet etc.) has been acknowledged by citation within the main report to an item in the References or Bibliography lists. I also agree that an electronic copy of this project may be stored and used for the purposes of plagiarism prevention and detection.

Statement of copyright

I acknowledge that the copyright of this project report, and any product developed as part of the project, belong to Coventry University. Support, including funding, is available to commercialize products and services developed by staff and students. Any revenue that is generated is split with the inventor/s of the product or service. For further information, please see www.coventry.ac.uk/ipr or contact ipr@coventry.ac.uk.

Statement of ethical engagement

I declare that a proposal for this project has been submitted to the Coventry University ethics monitoring website (https://ethics.coventry.ac.uk/) and that the application number is listed below.

Signed: denzel guma Date: 15/06/2022

First Name:	Denzel
Last Name:	Guma
Student ID number	9804324
Ethics Application Number	P130897
Supervisor	YingLiang Ma

Table of Contents

Table of Figures

Abstract

This project intended to examine whether it is possible to effectively implement machine learning techniques in video game Artificial Intelligence (AI) to improve the field of machine learning within game AI development. As to see whether an enhanced experience for the player/s of the game is created because of these implemented techniques. The intended implementation for the game AI in this project was the Asynchronous Advantage Actor Critic (A3C) which is a policy gradient-based Deep Reinforcement Learning algorithm (DRL). This algorithm was used to control the artificial agent/s within the 2D bomber-man-like game. With the objective of the learning algorithm being to mimic human behaviour rather than show inferior/superior behaviour.

Introduction

Project Aim

The aim of this project is to identify if machine learning techniques can be used effectively in video game development to mimic human behaviour in artificial agents.

Project Objectives

2D Bomber man game with A3C controlled AI that shows human like behaviour

Qualitative and quantitative results from Turing Test experiment that shows game AI was deemed human like and that the player/s gameplay experience was enhanced as a result.

Background and Context

AI within game development has been an active area research since the early seventies with the Atari games such as Pong and Space invaders being the earliest examples of rudimentary AI systems being implemented into video games.

Motivation for Project

AI in video games is yet to reach a level of realism where it can be deemed human-like as it is typically based on non-adaptive techniques such as Decision trees, Scripting etc. which are vulnerable to player exploits, diminishing user retention and game experience. By Investigating adaptive techniques such as machine learning within game development. The results produced will provide game developers with new findings that adds to the body of knowledge about machine learning implementation within game AI development. Allowing for developers to vastly improve the quality of their games and as a direct result improving the users game experience. This being a benefit for me personally as a future game developer and avid games enthusiast.

The Structure

The project will be structured in this format:

Literature Review, which Explores the current state of AI in video games particularly rule-based, and machine learnt approaches. As well as covering related works and gaps in the scientific body of knowledge; Method, which covers how the game was developed, the way in which the game was distributed and the way in which the Turing Experiment was conducted; Results and Evaluation, which covers the results of the Turing Test Experiments and gives a detailed analysis of what they mean; Project Management, which covers in detail how the project was conducted; and Conclusions which provides a conclusory statement on the projects research findings.

Literature Review

Introduction

This section will first introduce what video games are on a theoretical level. Followed by what artificial intelligence is, its rules and how its implemented within video games. Additionally, references to related works will be acknowledge and gaps in knowledge will be identified to highlight the key areas that this project will attempt to address.

What are Video Games?

According to Bergonse (2017) a video game is a "Mode of interaction between a player, a machine with an electronic visual display, and possibly other players, that is mediated by a meaningful fictional context, and sustained by an emotional attachment between the player and the outcomes of her actions within this fictional context.". These fictional contexts can be split into many categories which are referred to as "Game Genres" within the video games industry and encapsulates multiple game types such as:

Action Games: Action games focus on physically challenging player reflexes and precision. Examples of games in the action genre are shooter games (such as Halo 5), Platformers (such as Super Meat Boy), fighting games (such as Tekken 7) and driving games (such as Forza Horizon 5).

Adventure Games: Adventure games focus primarily on the narrative experience and/or exploring the generated world, solving problems. Examples of games in the adventure genre are Text adventure and interactive fiction games (such as Zork), Narrative adventure games (such as Life is Strange) and Puzzle adventure games (such as The Talos principle).

Role-Playing Games (RPG's): RPG's focus primarily on the narrative experience and allows the player to immerse themselves as the character/s in the game world through experiencing their perspective of the world. Examples of games in the RPG genre are Single Player RPG's (such as Final Fantasy X) and Massively Multiplayer Online Role-Playing Games (MMORPG's) Such as World of Warcraft.

Simulation Games: Simulation games focus primarily on realistically modelling the game to simulate hypothetical or real-world environments and situations. Examples of games in the Simulation genre are Life Simulation Games (such as The Sims 4), Vehicle Simulation games (such as Microsoft Flight Simulator).

Strategy Games: Strategy games focus primarily on player decision making and tactics to accomplish a set goal. Examples of games in the Strategy genre are Turn Based Games (such as XCOM 2) and Real Time Strategy Games (such as StarCraft 2).

Artificial Intelligence and Video Games

To define what AI is we first need to define what intelligence is. Intelligence is the ability to learn, process, comprehend, and store information gathered from the environment, with environmental adaptability being the emphasis as it shows the use of multiple cognitive processes indicating that a collection of abilities are being utilised (Good Therapy ,2017). With this clear definition of intelligence, it can be easily inferred that AI is the ability for artificial agents to demonstrate this intelligent behaviour.

Over the last two decades video games have gained popularity among AI researchers. As (Laird et al.,2001) suggests that they: are a cheaper alternative to robotics, have become more realistic with the advancement of graphics technology and are flexible enough to give the researchers more time to experiment. With this in mind (Spronck, 2005) suggested 7 rules that game AI developers should follow to provide a fair and entertaining challenge to the user to keep them engaged with the video game. If the game AI can accomplish these goals, then its overall quality will be elevated.

No Obvious Cheating: Artificial agent/s should not use information or execute actions that are principally unavailable to the user. If game AI does incorporate cheating, then it shouldn't be obvious to the player.

Unpredictable Behaviour: Artificial agent/s behaviour should be unpredictable to the user

No Obvious Inferior Behaviour: Artificial agent/s should not exhibit clear inferior behaviour to the human user.

Using the environment: Artificial agent/s should be able to explore and analyse an environment and then utilise the environment intelligently.

Self-correction: Artificial agent/s should be able to corrects its future behaviour to avoid repeating past mistakes

Creativity: Artificial agent/s should be able to come up with creative solutions to new unseen game situations

Human-like behaviour: Artificial agent/s behaviour should be indistinguishable to that of a human user.

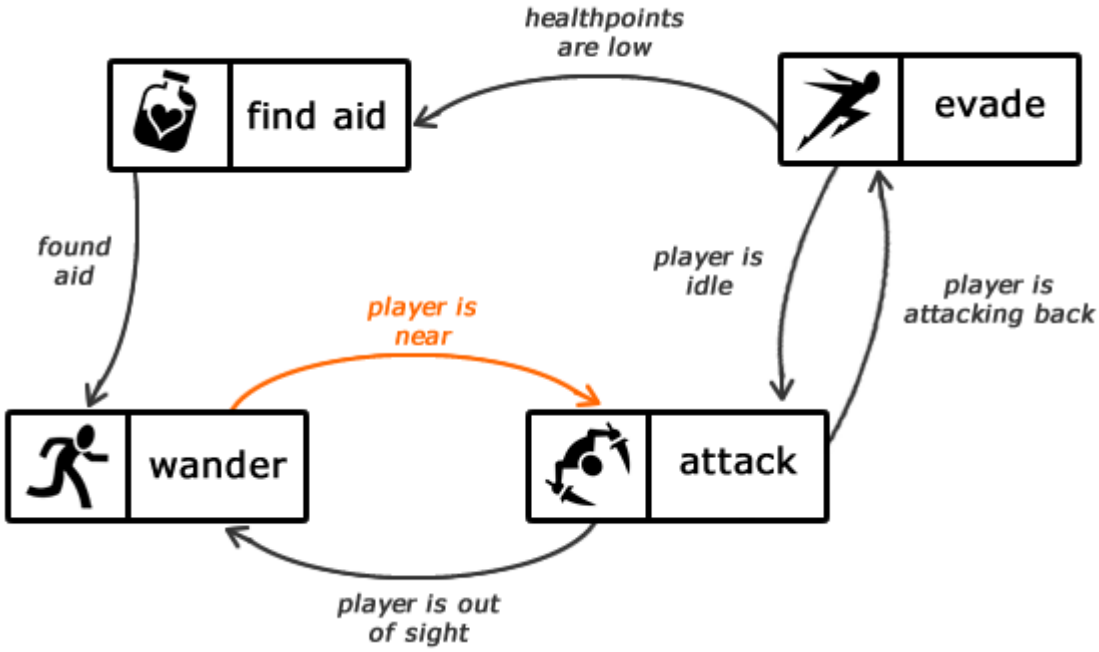
To achieve the Spronck rules game AI developers use a variety of different algorithms and approaches ranging from rule-based systems to more recently machine learning based systems.

Rule based Systems

Rule based Systems are systems in which artificial intelligence is accomplished through pre-determined rule sets and predefined outcomes. Currently among game AI developers this is the most popular approach as it a proven approach and the 3 most common techniques used within this approach are:

Finite State Machines (FSMs)

Figure 1: FSM example in a Video Game (Bevilacqua, 2013)



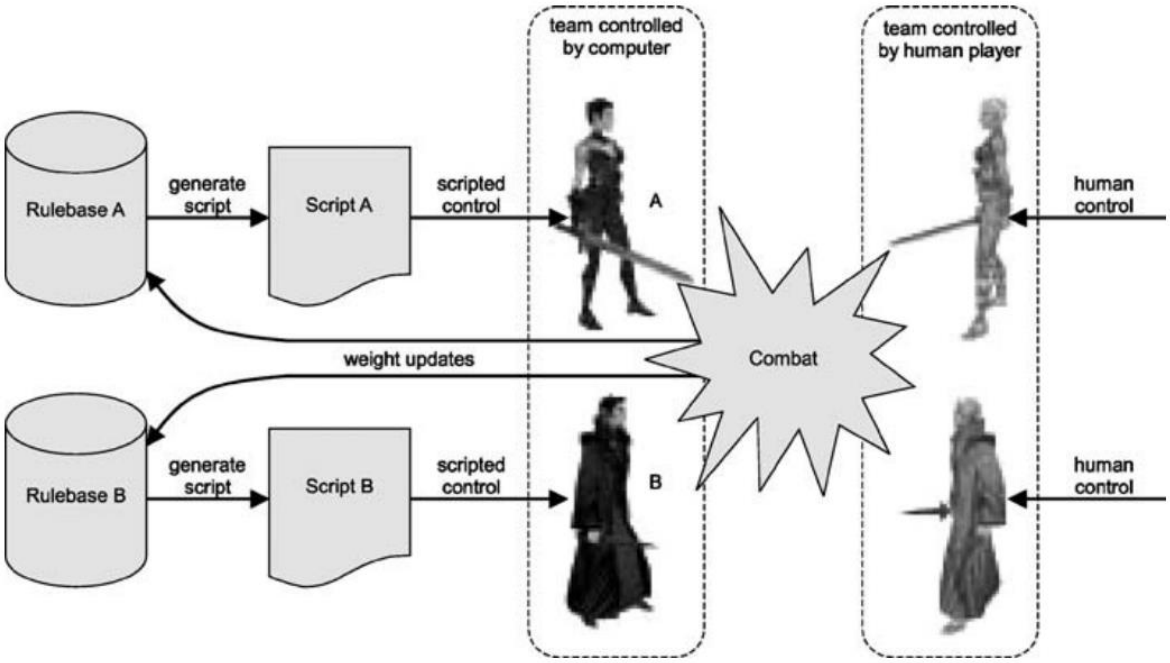
FSMs are models that consist of a finite number of behaviour states, transitions between those states and conditions for state transitions.

The advantage of FSMs is that they are very simple to implement reducing code complexity allowing for more efficient resource management for the game AI developers.

The disadvantages of FSMs are that they can become very large and complex meaning that they are very difficult to maintain. Additionally, if the FSM is too simple it could result in predictable AI behaviour resulting in a diminished game experience.

Scripting

Figure 2: Scripting example in a Video Game (Spronck et al., 2006)



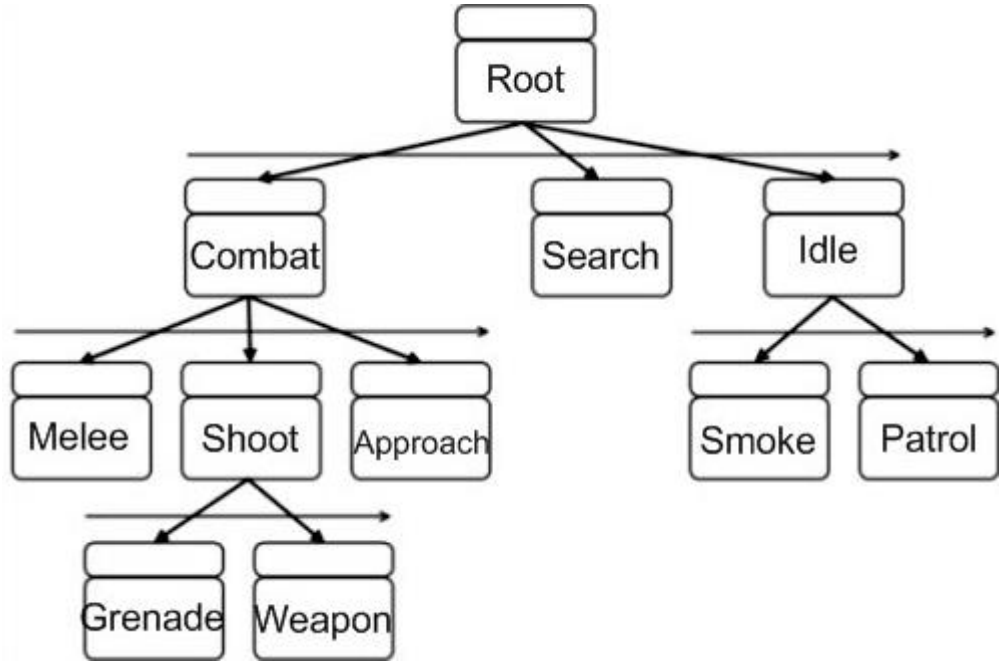
Scripts are programs that produce an expected series of behaviours based on a game's current given context.

The advantage of scripting is that it can be easily written by non-programmers resulting in increased productivity for the game studio.

The disadvantage of scripting is its limited scalability. As game complexity increases the script complexity and length increases resulting in more difficult scripts to maintain and understand due to programmer errors and design flaws. Potentially resulting in a subpar game AI system.

Decision Trees

Figure 3: Decision Tree example for a Video Game (Crytek, n.d.)



Decision Trees are models that can classify observations and make decisions based on those classifications.

The advantages of decision trees are that they are fast and easy to implement resulting in the game AI being less resource intensive on the current platform it is operating on.

The disadvantages of decision trees are that they can result in “artificial stupidity” such as abnormal behaviour in a given context and predictable behaviour. This resulting in a loss of immersion for the user.

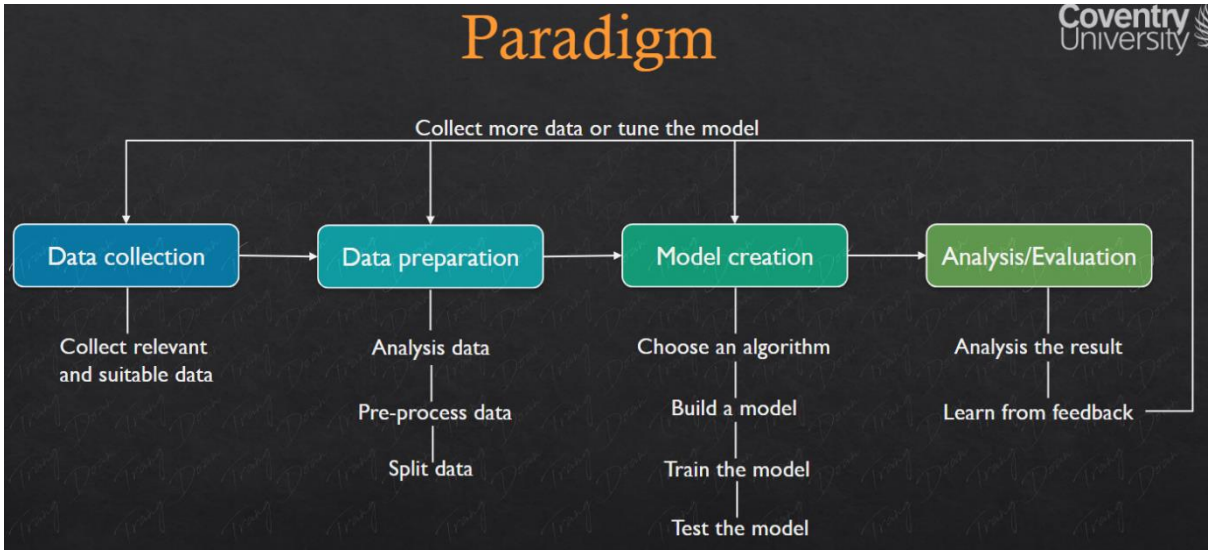
Although these rule-based approaches are the industry standard the major issue with these techniques is that once a vulnerability is discovered in the game AI’s behavioural patterns then human behaviour usually dictates that the player will continuously exploit the discovery for their own gain. The resulting effects of these exploits is that it becomes difficult for game developers to keep the player invested in the game as the stimulated experience is too easy for the player and he/she are not kept in a consistent flow state. Therefore, the overall game experience for the player is dramatically hindered (Guma,2021).

Machine Learning based Systems

Machine learning based systems are systems in which artificial intelligence is accomplished using machine learning. With machine learning according to Selig (2022) being "*an application of AI that enables systems to learn and improve from experience without being explicitly programmed.*".

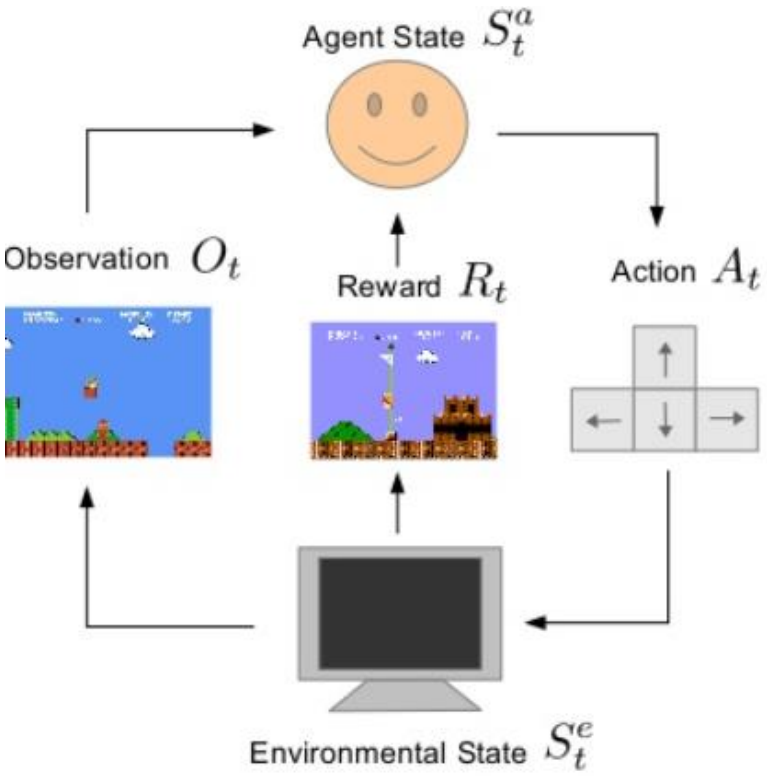
Within game AI development machine learning techniques for artificial agent tend to be seldom implemented due to concerns that it's application may result in uncontrollable and unpredictable behaviour. However recently due to the advancements in the field of game AI technology machine learning techniques such as Reinforcement Learning (RL), Deep Learning (DL) and Deep Reinforcement Learning (DRL) have started to slowly become incorporated in games on a research basis.

Figure 4: Machine Learning Paradigm (Doan,2022)



Reinforcement Learning

Figure 5: Example of Reinforcement Learning in Super Mario (Demense, 2019)



Reinforcement Learning is a machine learning technique in which an artificial agent interactively learns an environment through trial and error using reward/punishment feedback from past actions and experiences to try and optimally achieve a set goal. There 5 crucial components that make up a RL problem.

Environment: The world that the agent is operating in

State: Current state of the agent

Reward: Feedback from the environment

Policy: Algorithm used by agent to decides its actions

Value: Reward an agent receives from taking a certain action in a certain state

Additionally, Reinforcement Learning algorithms can be divided into systems

Model-free RL: In a Model-free RL system the artificial agent does not have access to its environment and will learn to create an optimal model for the unknown environment.

The advantages of Model-free RL systems are that it is easier to implement resulting in less code complexity. Additionally, there is no need-to-know prior state transitions or the entire environment resulting in this system having the potential to be used in a wider array of applications.

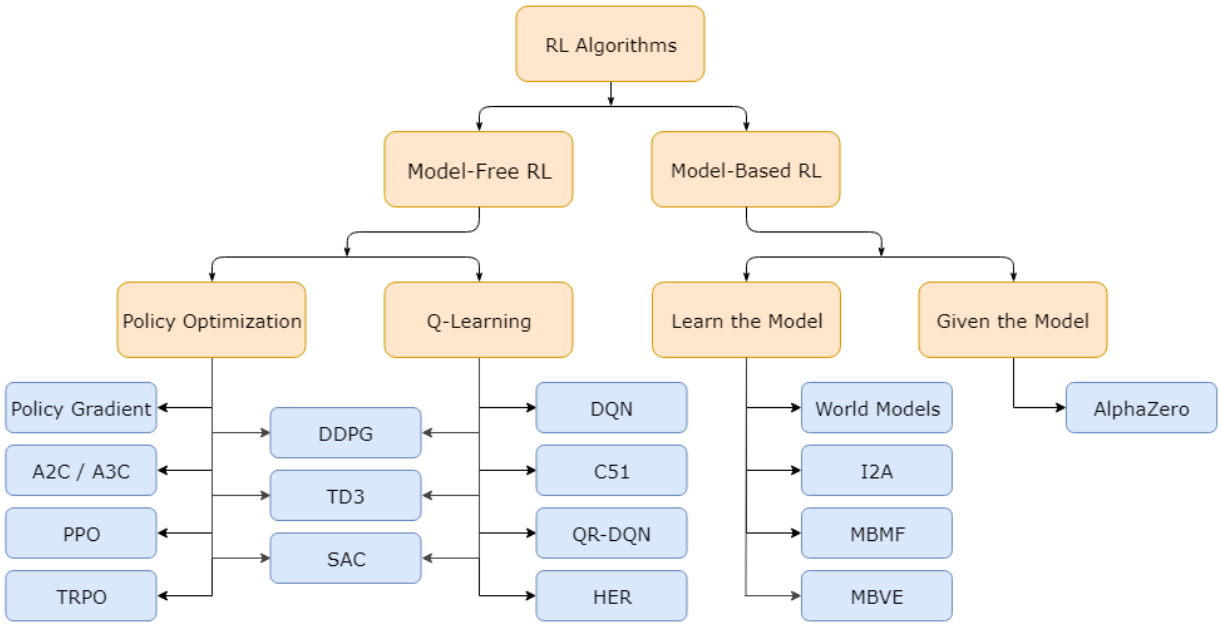
The disadvantage of Model-free RL systems is that agents may be able to perform optimally in a specific learned environment however behave sub-optimally when the environment changes.

Model-Based RL: In a Model-based RL system the artificial agent has access to its environment and will create an optimal model for it based on previous actions and experiences with the environment.

The advantage of Model-Based RL systems is that they tend to produce higher sample efficiencies out of the two systems meaning that it takes less actions and state transitions during training to reach a set performance threshold.

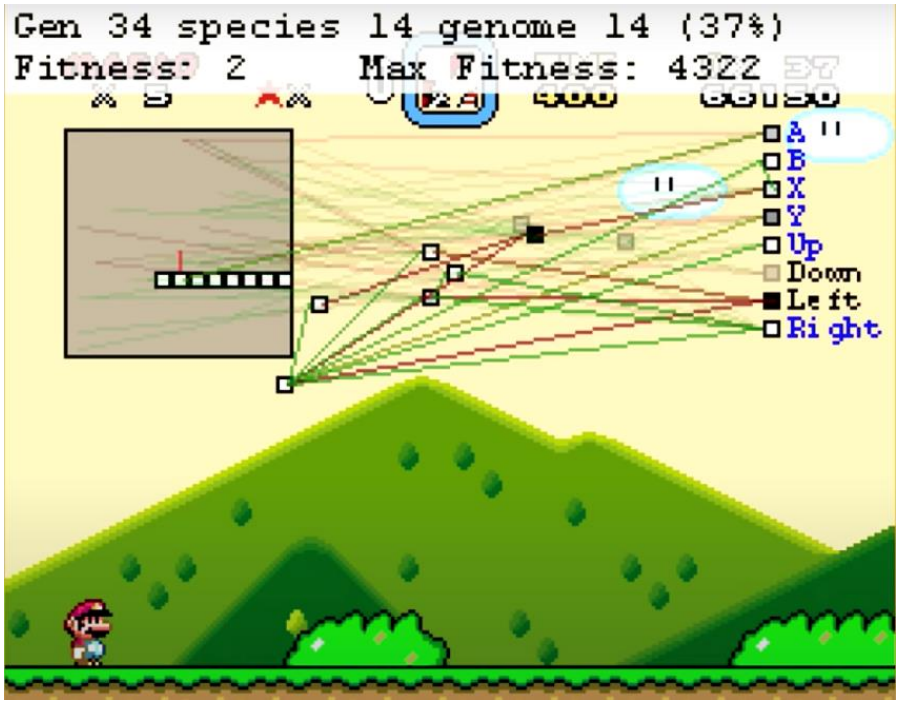
The disadvantages of Model-Based RL systems are that it is heavily dependent on how well the agent can model the environment and if modelled poorly can result in sub-optimal results. Additionally, Model-Based RL systems consider preferences of consequences meaning that the agent always performs a maximum reward action even if the future consequences of that action are detrimental

Figure 6: Reinforcement Learning Algorithms Hierarchal Structure (OpenAI,2018)



Deep Learning

Figure 7: Example of Deep Learning in Super Mario (Tucker, 2021)

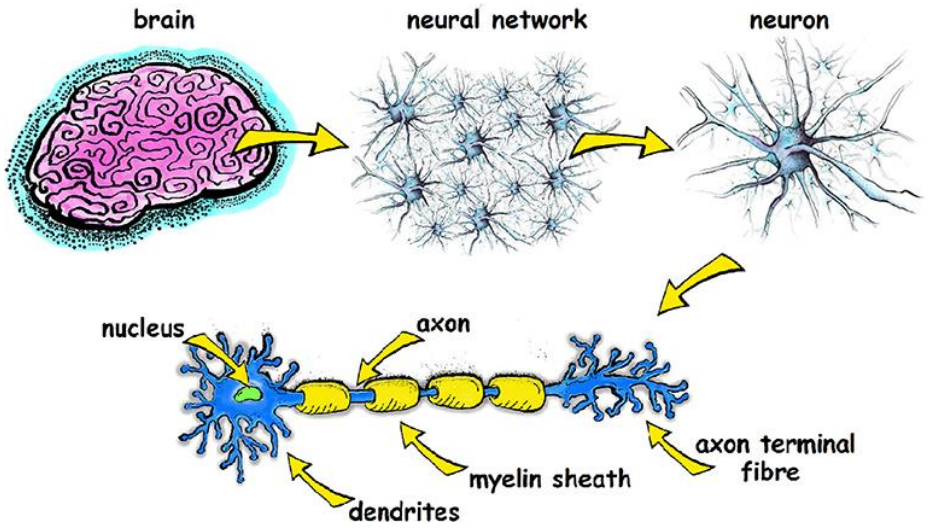


According to Brownlee (2020) Deep Learning is "A subfield of machine learning concerned with algorithms inspired by the structure and function of the brain called artificial neural networks."

To understand how Artificial Neural Networks (ANN) work and how they are structured. It is first critical to understand the basics of the brain's biological neural networks.

According to Vadapalli (2021) "A biological neural network is a network of neurons that are connected by axons and dendrites. The connections between neurons are made by synapses. The axons transport chemicals that cause neurotransmitters to be released onto dendrites, where the neurotransmitters are then able to excite or inhibit an adjacent neuron." This allows the brain to learn and retain information for future decision making.

Figure 8: Biological Brain Neural Network Structure (Frontiers for Young Minds, 2021)



The way an ANN works and is structured is that the neurons are part of 3 layers with each layer representing a specific function of a neural network.

Input layer: This layer contains input neurons that use input data from a particular source. For example, the input could be the game environment.

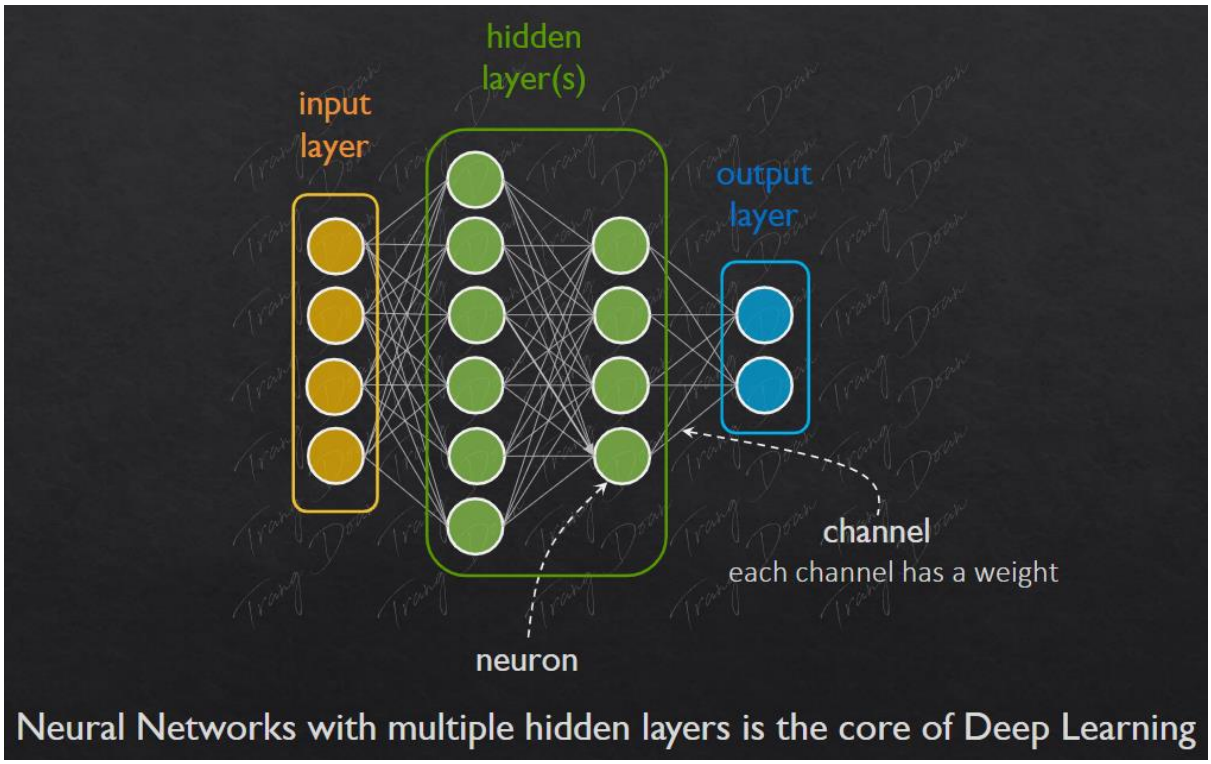
Hidden layer/s: This layer is the learning layer and contains neurons that applies an activation function (function that decides if a neuron is activated and therefore important to the network to solve the problem) to incoming Input neurons and weights or incoming hidden neurons and weights. Or outgoing hidden neurons and weights to output.

Each neuron in this layer has connected associated weight values. With these weights and their updates being critical to help the neural network learn the given problem.

Output layer: This layer contains neurons that produce a given output based on the input layer and hidden layer. For example, in a gaming context the digital pressing of X, Y, B, A which could represent attack, enter, crouch, jump or whatever game specific bind.

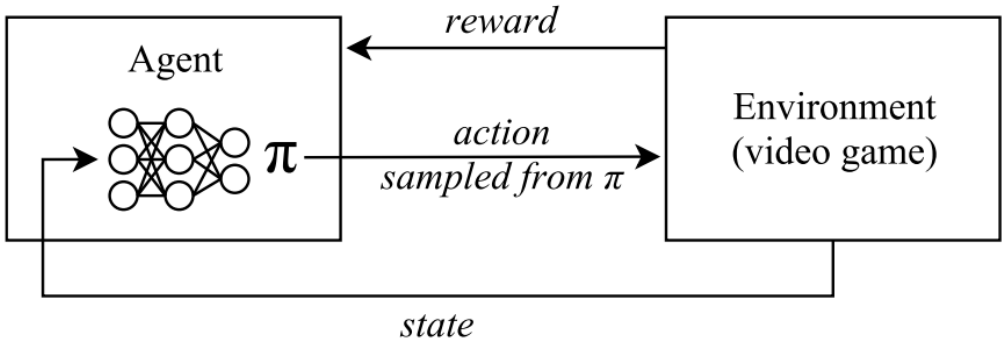
The way in which weights are updated in an ANN is through feed forward and back propagation. With feed forward propagation being the data moving between the layers and producing a predicted output either accurate or inaccurate. Then using back propagation (process of backwards tracing from the output layer through to the input layer to find the error rate based on the accuracy of the predicted output) the weights of the neurons are updated to minimise the error rate and increase accuracy. This process is cyclical and is used to constantly update the weights of the ANN.

Figure 9: Artificial Neural Network Structure (Doan,2022a)



Deep Reinforcement Learning

Figure 10: Deep Reinforcement Learning in a Video Game Context (Justesen et al., 2020)



According to Pathmind (n.d.) "Deep reinforcement learning combines artificial neural networks with a framework of reinforcement learning that helps software agents learn how to reach their goals. That is, it unites function approximation and target optimization, mapping states and actions to the rewards they lead to."

Related Works

With video games becoming increasing popular within modern media, it has highlighted to developers the need to shift from non-adaptive approaches to more advanced machine learnt approaches that can exhibit human like behaviour.

(Lohokare et al., 2020) used You Only Look Once Version 3 (YOLOv3) object detection algorithm to train agents in League of Legends (LOL). YOLOv3 was used identify different champions and Non-Player Characters (NPCs) in LOL. The researchers' also created two agents one agent trained on Proximal Policy Optimization (PPO) + Long-Term Short-Term Memory (LSTM) and the other agent using supervised LSTM trained on key logged information from a top tier LOL player. In a one-on-one custom games, the agents achieved first blood against amateur and average players.

(Creus-Costa et al.,2019) used a Deep Q-Learning Network (DQN) algorithm with the Epsilon Greedy exploration/exploitation algorithm to train a Slither.IO agent to play the online multiplayer game Slither.IO. The objective for the agent was to achieve the highest score possible against human players. When the DQN and Epsilon Greedy Learning Strategy was applied to the agent it observed the game's current frame and determined the best direction to move in.

OpenAI's DotA 2 team, OpenAI Five trained their DotA 2 agents using A3C which is a policy gradient based DRL algorithm. The bots initially played against each other in a 1v1, for a total of 180 in game years every day. The result of this training was that a lane agent was able to outplay professionals in a 1v1 context at The International 7 (TI7), Dota's highest tier of annual competition, with general ease. (Berner et al.,2019).

Gaps in knowledge

In conclusion one of the major reasons for the lack of video game AI advancement is a lack of understanding of advanced AI techniques. An excellent understanding of these techniques is becoming a necessity to create believable game AI systems. This research is going to fill in the gaps in machine learning implementation within game AI development by using the A3C DRL algorithm this is because the related works success condition was an artificial agent that can outperform a human player and show superior game skill whereas the success condition for this project instead focuses on game AI that's shows human-like behaviour imperfections and all. This will therefore lead to a more enjoyable and replayable gameplay experience (Guma,2021).

Method

Project Game

Figure 11: Project Game Visualisation

The game made for this project was an interactive 2D bomber man game which uses the OpenAI (n.d.) implementation of bomber man called Pommerman as the games base framework. OpenAI's (n.d.) Pommerman is their implementation of 1983's Nintendo Entertainment systems bomber man game with custom machine learning libraries and environments.

The basic logic of the projects game is four entities (1 Human controlled + 3 AI controlled) start each game at a set corner on a randomly generated 11x11 grid map.

Project Game AI

AI Action and Environment States

Action and Environment states of the projects game (OpenAI,n.d.) :

Action States

Figure 12: Action States of the project game

Action States	Result of Actions
Stop	This action is a pass.
Up	Move up on the board.
Left	Move left on the board.
Down	Move down on the board.
Right	Move right on the board.
Bomb	Lay a bomb.

Environment States

Figure 13: Environment States of the project game

Environment states	Dimensions	Environment state explanation
Board	121 Ints	The flattened board. All squares outside of the agent's purview will be covered with the fog value (5).
Position	2 Ints, each in [0, 10]	The agent's (x, y) position in the grid.
Ammo	1 Int	The agent's current ammo.
Blast Strength	1 Int	The agent's current blast strength.
Can Kick	1 Int, 0 or 1.	Whether the agent can kick or not.
Teammate	1 Int in [-1, 3].	Which agent is this agent's teammate. For FFA, this will be -1.
Enemies	3 Ints, each in [-1, 3].	Which agents are this agent's enemies. If this is a team competition, the last Int will be -1 to reflect that there are only two enemies.
Bombs	List of Ints.	The bombs in the agent's purview, specified by (X int, Y int, BlastStrength int).

Project Game's AI Systems

At every time step each agent receives an observation of the games environments state and chooses an action based on their given AI system. Within the projects game there are 3 AI systems used to control the behaviour of the agent/s.

A3C (Red Agent) which is an agent that uses the A3C DRL algorithm and has been trained on 25,000 games

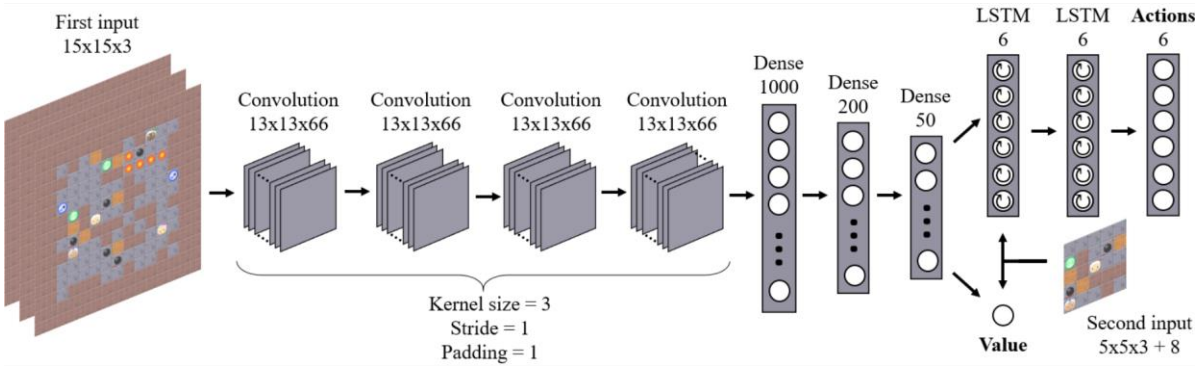
Random Agent (Blue Agent) which randomly takes chooses an action from the games 6 set actions

Simple Agent (Purple Agent) which is an agent that uses rule-based systems and Dijkstra's algorithm

Project Game's A3C Agent

The red artificial agent within the projects 2D bomber man game was controlled by the A3C algorithm which is a policy gradient based DRL algorithm that learns how to play the bomber man game via creating an optimal model for the unknown environment which in this case is the randomly generated 11x11 grid map. Additionally with this implementation of the A3C algorithm the red agent was trained against 2 Simple Agents and 1 Random Agent over 25,000 games. The A3C algorithm implementation used for the project's game A3C agent was heavily influenced by (Zainchkovskyy & Hasanbasic, 2019b).

Figure 14: A3C architecture in project game (Zainchkovskyy & Hasanbasic, 2019a)



Turing Test Experiments

The Turing Test was developed by Alan Turing in 1950 with the purpose to evaluate how well an artificial agent performs at imitating a human. And was the format used for this project to test how effective the A3C agent was at imitating human behaviour. However, the Turing Test format for this project was altered in a way in which all the entities within the game are all non-human.

The way in which the project's Turing Test works is that the human player joins a game session and plays a game of bomber man. Once the player/s has finished the game session they were asked which of the entities they believed were artificial and the reason as to why they thought this. Additionally, the players were asked how the entities affected their gameplay experience and to give the game a star rating. As to avoid influencing player response, players were not given any indication which entities within the game are human or non-human.

To gather these players for this project's experiment there was a recruitment strategy put in place via social platforms such as Discord, Snapchat, Itch.io etc. and these experiments were performed on a game build that was on the Itch.io website as a Zip File. This was done as to provide the most accessibility for all the players. The Itch.io game link = https://gumad.itch.io/pommermanio

The qualitative and quantitative data for this project was collected via a questionnaire which the user could access after finishing each game and clicking the "Answer Questionnaire" menu button. The questionnaire and answers were stored securely via Microsoft Forms to be later analysed and to determine whether machine learning techniques can be effectively applied in game AI development to replicate human behaviour. The Microsoft Forms Questionnaire link = https://forms.office.com/r/CpTww4BfdK

Results and Evaluation

The results and questions obtained for the experiment were both quantitative and qualitative

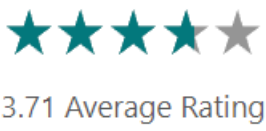
Figure 15: Pie Chart results of when participants were asked "Which of the players do you believe were human controlled?"

As can be seen from the figure above: 50% of the participants believed that the purple player was human controlled; 25% of the participants believed that there were no human controlled players; And 13% of the participants believed that the red and blue player were human controlled. This indicating that most of the participants believed that the most human like agent within the bomber man game was the purple player.

When looking at the participants qualitative answers it is easy to understand as to why the purple player (Simple Agent) was deemed to be intelligent by the participants whereas red player (A3C Agent) was deemed unintelligent. As when the participants were asked “Explain in detail why you think these players were human controlled. (If you think No Players were human controlled explain why you thought so)” The typical response for the purple player and the red player was along the lines of “I think the purple player was human controlled because he moves tactically and he does not die in the first few seconds of the game.” Indicating that the participants believed that the purple player showed strategic thinking which resembles that of complex human thinking whereas the red player died to early showing the inability to fully understand its environment.

The reason as to why the A3C Agent (Red player) seemed to perform poorly in game is because the A3C Agent was trained on 25,000 games which is not enough games for the model to understand the intricacies and complexities of its environment. Ideally for this to happen the A3C model would have been trained on at least 200,000 – millions of games but due to the hardware restrictions of the host computer this extensive training was not feasible. And resulted in a suboptimal A3C model being used for the agent.

Figure 16: Average star rating of the project game given by participants



When the participants were asked “What would you rate this game out of 5 Stars?” the average rating came out to a 3.71. Indicating that the game was good and had good features but had some weaknesses that could be fixed to improve the overall game quality. As can be seen through the participants responses when asked “Explain in detail how the players affected your overall gameplay experience “some participants said positively “The players made the game engaging” and “Pretty difficult” however there were some criticisms such as “The purple player gave the game an interesting challenge however the other players died to early for my liking” and “it is quite simple game so two players who died in the first few second was upset for me”

Project Management

Development Methodology

The development methodology used for this project was Agile. The Agile methodology is a development methodology in which software is developed in incremental, rapid cycles resulting in small incremental releases with each release building on previous functionality with each release/sprint being thoroughly tested to ensure software quality is maintained until a final build is deployed and launched (Try QA, 2020). The reason this methodology was chosen for this project was that the 2D Bomber Man game prototypes could be created quickly based on the potential adaptive requirements without detrimental effects. The two main artefacts associated with the Agile methodology generated for this project were the feature backlog and sprint backlogs. With the feature backlog being a list of all the required features that should be finished before a project is deemed complete and sprint backlogs being a list of features from the feature backlog to be worked on in each sprint. Sprint times usually vary from project to project but for this project 1 week is considered a sprint.

Figure 17:Sprint backlog and Feature backlog of the project with colour coding

Task	Start week	End week	Status
Game Development	**Week 1**	**Week 10**	**Completed**
Build map	Week 1	Week 1	Completed
Create Entities	Week 2	Week 2	Completed
Movement and Collission system	Week 3	Week 3	Completed
Timer System	Week 4	Week 4	Completed
Bomb and Lives system	Week 5	Week 5	Completed
Creating AI which uses Deep Reinforcement Learning	Week 6	Week 8	Completed
Multiplayer Networking	Week 9	Week 10	Removed
Exporting Game build to Itch.IO	Week 10	Week 10	Completed
Experiment and Results	**Week 11**	**Week 11**	**Completed**
Conduct experiments on game build on Itch.IO	Week 11	Week 11	Completed
Gather experiment results	Week 11	Week 11	Completed
Analyse experiment results	Week 11	Week 11	Completed
Dissertation	**Week 12**	**Week 13**	**Completed**
Introduction	Week 12	Week 12	Completed
Literature Review	Week 12	Week 12	Completed
Body of dissertation	Week 12	Week 13	Completed
Proofread	Week 13	Week 13	Completed
Submit	Week 13	Week 13	Completed

Social, Legal, Ethical and Professional Considerations

Social Considerations

Creating advanced machine learning AI systems within virtual video game environments shows proof of concept that such AI can self-teach given a restricted or unrestricted environment. And can be applied to real world applications such as advanced healthcare or autonomous driving.

Advancements in autonomous traffic could result in reduced traffic accidents and reduced CO2 emissions. Resulting in less vehicular incidents and a decrease in global warming. Or advanced healthcare in which surgeries can be performed by robots with pinpoint precision resulting in a decrease of health complications due to human error.

Additionally, with the observation of how of self-teaching systems approach problems and make decisions it provides humans alternative insights on the different techniques and approaches that can be used that weren't initially thought of.

Legal Considerations

General Data Protection Regulations 2018 (GPDR) sets out expectations for both Data Owners (who data is about) and Data Processors (people storing, using, or processing that data). Since this project does not actually utilise users IP addresses there was no need to conform to it.

Computer Misuse Act (CMA) set out the penalties for unauthorised access or modification of systems. To abide by the act the host system used for research and development was protected via both password authentication and anti-malware software.

Copyright Designs and Patent Act (CDPA) sets to protect the investment of time, money and effort by the people who created original pieces of work. To abide by the act CU Harvard referencing was used to give credit for intellectual property or work that was not my own. Additionally, a statement of originality was signed stating that all work was either original or referenced properly.

Ethical Considerations

Since the Coventry University Computer Science course is accredited with the British Computer Society (BCS) the BCS code of conduct was followed during the research and development of this project.

Professional Considerations

Since Pommerman falls under the Apache License 2.0 which is "A permissive license whose main conditions require preservation of copyright and license notices. Contributors provide an express grant of patent rights. Licensed works, modifications, and larger works may be distributed under different terms and without source code." The development and distribution of this projects game followed such requirements to abide by the license.

Risk Assessment

Figure 18:Risk Assessment table

Asset or Operation at Risk	Hazard	Scenario	Prevention / Mitigation	Probability (Low, Medium, High)	Overall Impact (Low, Medium, High)
Game Source code and/or Project Report	Hardware /Software failure	Computer crashes resulting in loss of unsaved code	Regularly back up work locally or via cloud	Medium	High
Health	Repetitive Strain Injury (RSI)	Excessive typing when working on project	Take breaks regularly and maintain good form	Low	Medium
Health	Stress	Increased stress levels due to workload	Effectively schedule to allow for smooth problem decomposition	Medium	Medium
Health	Covid-19	Catching Covid-19	Wear masks in public and properly sanitize	Medium	High

Record of Meetings

Figure 19:Record of meetings table

Date	Subject	Feedback/Actions
23/12/21	Project Proposal	Approved by supervisor
25/06/22	Game Implementation and Research Methodology	Removed the games incomplete multiplayer functionally and changed format of Turing Test based on supervisor recommendation
29/06/22	Results and Analysis	Results and Analysis need to be cleaned up. So were cleaned
1/7/22	Literature Review	Expanded on what video games are on a technical basis based on supervisor wanting a concrete understanding
1/7/22	Literature Review	Added images and diagrams for the AI techniques within video games as supervisor suggested it would make it easier for the reader to understand the tough concepts
2/7/22	Body of Dissertation	Edited the body of the dissertation so that it constantly linked to the project topic based as recommended by the supervisor

Proforma Document Feedback

Figure 20: Proforma Document Feedback table

Feedback	**Action**
Write in the third person (apart from motivation section)	Entirety of project report was written in third person except for the motivation section
Avoid personal bias; no statements like 'I feel', "I think", "I believe" etc…	Statements within the project report are backed up by evidence and any personal bias was removed
Avoid sweeping statements without relevant facts and a suitable reference. For example, "Crime has increased across the globe."	Statements within the project report that can be considered sweeping were removed or backed up with facts and references
Sentences should not be over long – make them concise and easy to read. Limit the use of connectives; 'however', 'therefore' etc…	The grammar and sentence structures used in the project report are concise and straight to the point
Double-check spelling and grammar – ask somebody not connected with project to read it to ensure it is understandable.	The project report was proofread by an external entity and edited accordingly where needed
Make sure it flows – between sections and within sections (particularly important with literature review).	Headings and subheadings of the project report are structured in a cohesive manner to allow for seamless content transitions for the reader
Relate writing back to your project question / topic i.e. content must be relevant	Content within the project report is always related back to the project topic

Project Management Reflection

Finding the initial topic for my final year project was less troublesome than I initially thought that it would be and was more focused on deciding which DRL algorithm that I would personally like to incorporate into a video game.

The research for the project was interesting and easy to understand since the modules I specialise well in were 5000CEM - Introduction to Artificial Intelligence and 6006CEM - Machine Learning and Related Applications. Additionally, the A251DMLL - Games Design Studio module really allowed me to further understand how AI and Machine Learning systems should be incorporated into modern video games.

During development the use of the agile methodology was beneficial as this is the methodology that allows for prototypes to be built in small increments based on changing requirements. An instance of this being useful in the project was when the incomplete multiplayer functionality was removed as to have a chance in completing the project on time.

When learning OpenAI's (n.d.) Pommerman framework I did find it extremely difficult at times to gauge an understanding of how their libraries and environments interlinked together and that did cause a slight delay in the development progress. However, once I figured out their framework, I began to complete the projects games features quicker than I initially anticipated.

Conclusions

Summary Of Results

Simple Agent (Purple Player) was deemed more intelligent than A3C Agent (Red Player) by human participants

A3C Agent (Red Player) and Random Agent (Blue Player) tended to die very early in the project game

Majority of participants enjoyed playing the game with the Simple Agent (Purple Agent) but wish for the A3C Agent (Red Player) and Random Agent (Blue Player) to be improved

Participants highly valued movement in determining human intelligence

Study Contributions

This research has added to the body of knowledge of machine learning AI techniques within video game development. And demonstrated the impact of sub-optimal machine learnt agents within a distributed video game. Additionally, it has given notice to developers that it is critical to fully analyse whether it would be beneficial to incorporate machine learning approaches vs rules-based approaches by using cost benefit on resources such as computation, human etc.

Limitations

The main limitation for this project was in relation to the host computer used to train the A3C model. The system specs for the host computer were an AMD Ryzen 3 1200 Quad-Core Processor for the CPU and a GTX 970 for the GPU which resulted in model training times for 25,000 games being 4-5 hours which when extrapolated to 100,000 games would have been between 16-20 hours and extrapolated to a million games would have been between 160-200 hours. For the scope of this project these training time extrapolations were simply not possible to even consider.

Future Works

Future works for this project would include comparing multiple model free based DRL algorithms against one another to see which produces the most intelligent agents. Additionally, since the multiplayer functionality was removed due to time constraints, this component would be incorporated as to allow for players to face both human and AI controlled player. This giving the research an additional layer of information as to what constitutes intelligent behaviour in a video game player.

Reflection

As a future game developer this research has allowed to me to gain a better understanding of the foundational concepts within AI game development and the new emerging machine learning technologies that will revolutionise game AI development and the entire video games industry.

Bibliography

Bergonse, R. (2017). Fifty Years on, What Exactly is a Videogame? An Essentialistic Definitional Approach. *The Computer Games Journal*, *6*(4), 239–255. https://doi.org/10.1007/s40869-017-0045-4

Berner, C., Brockman, G., Chan, B., Cheung, V., Debiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., Józefowicz, R., Gray, S., Olsson, C., Pachocki, J.W., Petrov, M., Pinto, H.P., Raiman, J., Salimans, T., Schlatter, J., Schneider, J., Sidor, S., Sutskever, I., Tang, J., Wolski, F., & Zhang, S. (2019). Dota 2 with Large Scale Deep Reinforcement Learning. *ArXiv, abs/1912.06680*.

Bevilacqua, F. (2013). *Finite-State Machines: Theory and Implementation*. Game Development Envato Tuts+. https://gamedevelopment.tutsplus.com/tutorials/finite-state-machines-theory-and-implementation--gamedev-11867

Brownlee, J. (2020). *What is Deep Learning?* Machine Learning Mastery. https://machinelearningmastery.com/what-is-deep-learning/

Creus-Costa, J., & Fang, Z. (2019). Learning to play SLITHER.IO with deep reinforcement learning.

Crytek. (n.d.). *Coordinating Agents with Behavior Trees - Technical Documentation - Documentation*. https://docs.cryengine.com/display/SDKDOC4/Coordinating+Agents+with+Behavior+Trees

Demense, T. (2019). *Types of Machine Learning algorithms*. Tzeny's Demesne. https://tzeny.com/2019/02/22/types_of_machine_learning_algorithms/

Doan, T. (2022). *Machine Learning Introduction* [Slides]. PowerPoint. https://coventry.aula.education/#/dashboard/aa801eeb-4b21-4657-a3b0-d3f848208605/journey/materials/be810b0a-f1a7-4bc5-a6b2-f20b9d2e1aa7

Doan, T. (2022a). *Neural Networks* [Slides]. PowerPoint. https://coventry.aula.education/#/dashboard/aa801eeb-4b21-4657-a3b0-d3f848208605/journey/materials/973641f3-f6a5-4a6e-b4de-3838ee970826

Frontiers For Young Minds. (2021). *What Is An Artificial Neural Network And Why Do We Need It?* https://kids.frontiersin.org/articles/10.3389/frym.2021.560631

Good Therapy. (2017). *Intelligence*. https://www.goodtherapy.org/blog/psychpedia/intelligence

Guma, D. 2021. Project Proposal submission made to Coventry University for module 6000CEM – Individual Project Preparation

Justesen, N., Bontrager, P., Togelius, J., & Risi, S. (2020). Deep Learning for Video Game Playing. *IEEE Transactions on Games*, *12*(1), 1–20. https://doi.org/10.1109/tg.2019.2896986

Laird, J.E., & Lent, M.V. (2001). Human-Level AI's Killer Application: Interactive Computer Games. *AI Mag., 22*, 15-26.

Lohokare, A., Shah, A., & Zyda, M. (2020). Deep Learning Bot for League of Legends. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, *16*(1), 322-324. Retrieved from https://ojs.aaai.org/index.php/AIIDE/article/view/7449

OpenAI. (2018). *Part 2: Kinds of RL Algorithms — Spinning Up documentation*. https://spinningup.openai.com/en/latest/spinningup/rl_intro2.html

OpenAI. (n.d.). *Pommerman*. Pommerman. https://www.pommerman.com

Pathmind. (n.d.). *A Beginner's Guide to Deep Reinforcement Learning*. https://wiki.pathmind.com/deep-reinforcement-learning

Selig, J. (2022). *What Is Machine Learning? A Definition.* Expert.Ai. https://www.expert.ai/blog/machine-learning-definition/

Spronck, P. (2005). *Adaptive game AI*. Maastricht University. https://cris.maastrichtuniversity.nl/en/publications/adaptive-game-ai

Spronck, P., Ponsen, M., Sprinkhuizen-Kuyper, I., & Postma, E. (2006). Adaptive game AI with dynamic scripting. *Machine Learning*, *63*(3), 217–248. https://doi.org/10.1007/s10994-006-6205-6

Try QA. (2020). *Try QA*. http://tryqa.com/what-is-agile-model-advantages-disadvantages-and-when-to-use-it/

Tucker, K. K. (2021). *Using Machine Learning To Beat A Video Game*. Infused Innovations. https://www.infusedinnovations.com/blog/artificial-intelligence/using-machine-learning-to-beat-a-video-game

Vadapalli, P. (2021). *Biological Neural Network: Importance, Components & Comparison*. upGrad Blog. https://www.upgrad.com/blog/biological-neural-network/

Zainchkovskyy, E., & Hasanbasic, M. (2019a). *A3C Architecture*. GitHub. https://github.com/eugene/pommerman/blob/master/img/architecture.PNG

Zainchkovskyy, E., & Hasanbasic, M. (2019b). *GitHub - eugene/pommerman: Bomberman deep reinforcement learning challenge in PyTorch*. GitHub. https://github.com/eugene/pommerman