EPiC Series in Computing

Volume 93, 2023, Pages 188–202

Proceedings of Society 5.0 Conference 2023

Game-based learning - Teaching Artificial Intelligence to play Minecraft: a systematic literature review

Reinhardt Smit[1], Hanlie Smuts[1,[0000-0001-7120-7787]]
[1] University of Pretoria, Pretoria, South Africa
U18131451@tuks.ac.za; hanlie.smuts@up.ac.za

Abstract

Artificial Intelligence (AI) are machines designed to think and behave as humans would. Taking AI and placing it into a virtual world they become known as AI agents which uses the knowledge it gained from training to perform tasks in the world. AI agents in the virtual world has only been able to perform a narrow set of tasks with specialised models in environments with limited complexity and diversity. A rich world that requires an agent to continuously learn from and adapt to a wide variety of open-ended tasks and use previously gained knowledge to determine the next course of action will render the agent incapable. In order to investigate the AI teaching methods applied to instruct the agent to perform basic tasks in Minecraft in order to identify which AI teaching methods will yield the best results, a systematic literature review was conducted by extracting 57 papers and identifying themes and sub-themes that suited AI agent training methods and functions. This was to discover wat AI training methods can be implemented to enable an agent to perform tasks in a complex and rich world, contributing to game-based learning. The study found that a well-integrated Reinforcement Learning (RL) method with an effective reward system equipped the agent with the necessary knowledge to be able to perform tasks on a more complex level. A list of unique methods was integrated with RL such as Newtonian Action Advice (NAA), Behavioural Cloning (BC), VideoPreTraining (VPT), human demonstrations, and natural language commands to achieve a certain goal. This meant that AI agents can be taught to perform open ended tasks in a complex environment by setting up a well thought out framework on how to teach the agent in various areas leading to the possibility to incorporate those teachings into the real world through game-based learning.

Keywords: Game-based Learning; Society 5.0 Education; Minecraft Reinforcement Learning; AI agent; Training AI agent

1 Introduction

Artificial Intelligence (AI) refer to machines designed to think and behave as humans would (Kanervisto, 2022). This process entails learning through various trials, identifying patterns, and determining what actions work to equip itself with the skill to deliver efficient outcomes (Kanervisto, 2022). Its performance has reached such heights that its application became attractive in all areas of life, business processes, medical practices, transport, and virtual environments (Thawonmas, Togelius, & Yannakakis, 2019).

In the world of gaming, AI agents are used to learn and perceive in a gaming environment and game-based learning (Garg & Sharma, 2021; Samsudin, Ming, Ahmad, & Abrose, 2022). The AI agents take the role of the player when learning to play the game. Intelligent agents learn and improve as it explores the environment it is placed in (Palanisamy, 2018). Based on its observations and what it perceives in the world, it analyses the way it performs tasks, identifies patterns and finds ways to achieve the outcome more accurately and with efficiency (Kich et al., 2022). This way of learning is in a safe environment where mistakes can be made and learned from (Togelius, 2019). The challenge is to present an environment complex enough to challenge an AI agent, as the depths of the real world would, to grasp more of an idea what AI is and what it is capable of (Aluru, Tellex, Oberlin, & MacGlashan, 2015).

On-line games, and for the purpose of our research study, Minecraft (Edition, 2019), have now gone far beyond its roots as a simple building game (Green, Salge, & Togelius, 2019), and through mods and additional development, Minecraft can also be used as a teaching tool to instruct students on topics such as conservation and resource management (Edition, 2019). This study therefore analyses the AI teaching methods applied to instruct the agent to perform basic tasks in Minecraft in order to identify which AI teaching methods will yield the best results. Hence, this research study considers the research question "*What is the best method to teach AI to execute basic tasks in Minecraft?*". By identifying and understanding the best AI teaching methods related to playing Minecraft, this research may contribute to a better understanding of the effectiveness of Minecraft-game-based learning.

The rest of the paper is structured as follows: Section 2 is the background to the study followed by the research approach in Section 3. Section 4 contains the analysis of the data and Section 5 an overview of the contribution. Section 6 concludes the paper.

2 Background

AI are intelligent machines and computer programs created to use their computational power to learn and replicate intelligent processes to be able to think and act rationally like humans do (Garg & Sharma, 2021). AI can automate this learning process by going through datasets, but still requires human intervention implying that it is not an independent system. As it progressively learns it becomes better at reasoning which algorithm to use, solving problems in the situations they find themselves in and self-correct existing algorithms to make them work more effectively (Yannakakis & Togelius, 2014). AI is also well developed to be able to perceive the things around them and analyze, interpret, and acquire the information presented to them in a human sensory way (Boden, 2016). AI has functions that work together to enable it to learn to perform better and do its tasks in an effective way (Kope, Rose, & Katchabaw, 2013).

2.1 Game-based learning

Society 5.0 refers to "a people-centric society founded on the merging of cyberspace and physical space" (Deguchi et al., 2020:4). Education in Society 5.0 has now entered an era where technology and the internet is an integrated part of human life and learning (Murtiningsih & Ainia, 2022). In this cyberphysical space, digital games present a new teaching paradigm as it encourages computational thinking. Hence, researchers in teaching and learning are interested in the integration of AI and ML with digital gaming (Amla, 2021; Sunarya, 2022). Game-based learning "is structured to integrate gameplay into a subject matter to increase a student's ability to acquire and apply knowledge to the real world" (Hart, 2021:5). Game-based learning increases student attention in addition to the benefits of engagement, competitiveness, and teamwork (Dyulicheva & Glazieva, 2022; Sunarya, 2022). Training a virtual agent to outperform human players can demonstrate how to optimize different processes (Dyulicheva & Glazieva, 2022) while it enables researchers to track how much the algorithm improves over time (Sunarya, 2022). AI needs data to calculate the tasks at hand, and that data can be given in a variety of formats (Frazier & Riedl, 2019). The data is processed through Machine Learning (ML) which is an umbrella term for other functions like Neural Networking and Reinforcement learning (RL) that are used to work through the given data and pick up patterns of behaviour and outcomes to be able to predict what a certain set of actions will produce (Oliveira, Martins, Magalhaes, & Goes, 2019).

Human imitation learning is also a big contributor to teaching AI agents to perform tasks in a virtual world (K. X. Nguyen, Bisk, & Iii, 2022). For the agent to understand the videos it is watching and the content it is reading, the agent uses a language transcript model to go through and learn what it is going through (Dreveck, Salgado, Clua, & Gonçalves, 2021). Some methods entail humans giving the agent advice, a common term for that kind of method is Newtonian Action Advice (NAA) and the Feedback Arbitration (FA) method (Oliveira et al., 2019). We will be evaluating these teaching methods and data formats used for the AI Agent through ML and its sub-functions to see how well the agent performed.

For our study, we chose Minecraft - an open-world sandbox game with elements of exploration, resource gathering, crafting, construction, and combat (Singh, 2020). In this case, we evaluate the actions taken to teach an AI Agent to perform tasks in the virtual world of Minecraft (Justesen, Bontrager, Togelius, & Risi, 2019). Minecraft presents us with a world of open- ended tasks relying hugely on human judgment capabilities to determine the next set of actions (Fan et al., 2022). The challenge that Minecraft presents is the limitless set of possibilities of what can be done and how it can be done (Aluru et al., 2015). This means that incorporating methods for an AI Agent to play the game can be hard to determine (Fedorenko, Kaidan, Velychko, & Soloviev, 2021; Singh, 2020).

2.2 Teaching AI agents

Various methods are used to teach AI Agents to perform tasks in games other than Minecraft. AlphaGO was a project where an agent beat a world champion in the famous Japanese board game called Go (Guss et al., 2021). By making use of RL the AI learned the game by playing simulation games against itself and determined all the possible outcomes of each play, each strategy, and calculated a counteract for it (Wang et al., 2016). A StarCraft 2 agent became a grand master in the game by firstly being trained by a human showing it how to play, then improved by playing against other agents learning what works and what does not work through RL to finally beat a champion StarCraft player (Ontanón et al., 2013). A similar outcome was achieved when an AI system called OpenAI learned to play Dota 2 for 10 months using RL techniques and used what it had learned to play against a champion Dota 2 team. The OpenAI defeated the team in all the matches they played (Berner et al., 2019).

Agents that performed in these game worlds, were confronted with only a few complexities. This is what makes Minecraft so significant. Minecraft is an open-world sandbox game presented in a 3D virtual environment (Duncan, 2011). Players placed in this open-ended world can execute all kinds of tasks like building, crafting, mining, fighting, eating, and ultimately surviving (Nebel, Schneider, & Rey, 2016). It also allows players to set up complex structures and implement scientific systems all to fit the players' creativity (Nebel et al., 2016). Working in a virtual environment such as Minecraft created the potential methods to get AI into a learning and task-performing state. It puts the agents in an environment that presents real-world tasks and challenges. This opens new grounds for discovery to learn more about AI and discover what we understand about it and what its potential might be (Duncan, 2011).

The discoveries that the previous games showed were the enormous computational power AI has and how it can easily outperform human champions in games they have mastered (Milani et al., 2020). Minecraft expands the horizon for AI learning capabilities which shows that the achievements that can be made are something to look forward to (Singh, 2020). In previous studies, RL has been implemented into Minecraft to perform various tasks through a competition setting where the agents were evaluated on how well they performed certain tasks (Milani et al., 2020). For example, in a competition it was required that the agent find diamonds (Kanervisto et al., 2022). What was discovered was the complexity of the reward system that only needed to be implemented with crafting a pick-axe which showed that it was hard for the RL Agent to learn due to the rarity of finding certain items in the Minecraft world (Milani et al., 2020). Another factor that played a role was the restrictive measures that were set in place. There were too many restrictions during the teaching and learning phase to expect high performance from the agent (Shah et al., 2022). Once one of the teams implemented their method outside of the restrictions the diamond was found multiple times and certain complexities were slowly overcome (Guss et al., 2021).

The significance of these discoveries is related to how it guides us to what still needs to be achieved. Working with AI in a safe virtual environment shows us that there is still significant progress to be made (Garg & Sharma, 2021). The agents playing the game still rely on human input to guide them (Zhadan, 2018). Making progress to have it learn by itself and become a master of the game like the agents from StarCraft 2, Dota 2 and the board game Go, we could see innovative ways that AI complete some of the in-game tasks that could be game-changing for real-world scenarios.

3 Research Approach

The aim of this study is to analyse the AI teaching methods applied to instruct agents to perform basic tasks in Minecraft in order to identify which AI teaching methods will yield the best results. We executed a systematic literature review (SLR) in order to extract and analyse existing academic, peer-reviewed papers related to the research question (Oates, Griffiths, & McLean, 2022). We executed the SLR process by following 4 steps. Firstly, we applied the search terms ("Artificial Intelligence" and "Minecraft" and "teaching methods") in Google Scholar and obtained a results dataset of 319 papers. Secondly, we removed 32 duplicates leaving us with 287 papers. Thirdly we screened the abstracts of the papers for relevance to our research questions, leaving us with 79 papers. Finally, we studied the full text of the 79 papers, ending up with a final dataset of 57 papers. We excluded papers not written in English and papers with the full text not available. Quality assessment was applied in support of the primary research question i.e. an AI agent was indeed taught how to complete a task in the world of Minecraft and the method and outcome of teaching the AI agent, were clearly described.

The data extracted for this study focused on AI executing tasks in Minecraft or references to AI in relation to its application in teaching it to play. The purpose of the extract was to establish what methods were used to train AI agents to be able to perform tasks in the world of Minecraft. In order to analyse the data extracted, we created a data extraction dataset and conducted thematic analysis. Thematic analysis entails doing the sort of analysis that requires texts to discuss a specific theme to discover common features, patterns, and ideas in the format of qualitative data (Braun & Clarke, 2012). It is all about discovering the narrative the data is trying to carry and unpacking that to identify if there is any relevance to the goals of the researcher's question (Braun & Clarke, 2012).

The way we implemented our data analysis in this specific study was by identifying AI training methods as the main themes. Each method was studied thoroughly and was analysed to discover the unique sub-themes each method used. We then focused on the sub-themes to be able to clearly explain the important functions each method used to train an AI agent how to perform tasks in the world of Minecraft. Table 1 depicts the data extraction table after thematic analysis was completed.

Table 1: AI Agent training methods themes and sub-themes

Main Theme	Sub Theme	References
AI agent training Method 1: agent receives human feedback based on the reward calculated from the performance of a certain action	Elimination Process	(Justesen et al., 2019; Loubos, 2018; Oliveira et al., 2019)
	Delta Debugging	(Loubos, 2018; K. X. Nguyen et al., 2022; Woof & Chen, 2018)
	World State	(Canaan, 2018; Salge, Green, Canaan, & Togelius, 2018; Wichlacz, Torralba, & Hoffmann, 2019)
AI agent training Method 2: RL AI agent that must be able to interact with the environment	Newtonian Action Advice	(Krening & Feigh, 2018a, 2018b, 2019)
	Feedback Arbitration	(Frazier & Riedl, 2019; Kanervisto, 2022; K. X. Nguyen et al., 2022)
	Synthetic Oracle	(Pyarelal, Banerjee, & Barnard, 2023)
AI agent training Method 3: hybrid solution that makes use of ML, a rule-based approach, and human feedback	Sate Classifier	(Beyret et al., 2019; Goecks, Waytowich, Watkins, & Prakash, 2021)
	State Machine	(Illanes, Yan, Icarte, & McIlraith, 2019; Lin et al., 2021)
	Navigation Policy	(Kanervisto, 2022; Shah et al., 2022; Sun, Wu, & Lim, 2020; Xihan, Mendez, & Hadfield, 2022)
	Simulator	(Duan, Yu, Tan, Zhu, & Tan, 2022; Shah et al., 2022)
	Demonstration Labels	(Jain et al., 2019; K. X. Nguyen et al., 2022)
	Estimated Odometry	(Shah et al., 2022)
AI agent training Method 4: trained the AI agent before placing it in the world of Minecraft, pre-trained with a learning algorithm that went through large internet-scale knowledge	Observation Space	(Canaan, 2018; Krening & Feigh, 2018a)
	Action Space	(Fan et al., 2022; Krening & Feigh, 2018b)
	MineClip	(Allison, Luger, & Hofmann, 2018; Fan et al., 2022; Luketina et al., 2019; Shu & Tian, 2018; Zaidi, n.d.)
	MineDojo Simulator	(Allison et al., 2018; Jansen, 2021)
AI agent training Method 5: openAI used semi-supervised imitation learning method	Inverse Dynamics Model	(Baker et al., 2022)
	Behavioural Cloning	(Kanervisto, 2022; Pearce & Zhu, 2022)
	Video PreTraining Foundational Model	(Broll, Hausknecht, Bignell, & Swaminathan, 2019; Guss et al., 2021)

The main theme in Table 1 categorises different training methods, while the sub-theme indicates the specific training method applied. The final column in Table 1 reflects the references. The results of the thematic analysis shown in Table 1 are discussed in detail in Section 4.

4 Results

4.1 AI agent training Method 1

Method 1 focusses on a method where the agent receives feedback based on the reward calculated from the performance of a certain action, a common method used with RL (Justesen et al., 2019). It follows a high-level concept called Mission Spec with an integrated Minecraft API called Project Malmo (Justesen et al., 2019). They test and use their method in a small 50x50 world that allows the agent to run around as a normal player would. To determine its movement the agent is presented with a grid and determines if he can move forward by making use of a 2D array that has the names of blocks in their positions to ultimately calculate the world state (Wichlacz et al., 2019). The agent uses the information from this array to calculate if it must jump and in what direction it needs to go. The algorithm also saves the agent when it gets stuck by making it turn and recalculate to a new route. At the start of each attempt, the agent is placed in the centre with pre-calculated routes (Salge et al., 2018). The goal of the agent is to pick up all the items in the world and craft with the given recipes at the start of each round. After each round, a new world state is presented to give the agent new obstacles and environments to work with and be able to learn how to handle them (Canaan, 2018).

For the agent to learn how to pick up the correct items for a crafting recipe it goes through an elimination process (Loubos, 2018). The elimination process is to get rid of items that are of no value to the goal of the task (Oliveira et al., 2019). In each round, the agent is given a recipe for an item to craft. The agent only attempts to gather all the items, and then uses a delta debugging method to find the solution to a recipe (K. X. Nguyen et al., 2022)). This is similar to the RL reward system - the agent will use the first item it picked up that has not been used for crafting before and drop that item (Woof & Chen, 2018). After the item is dropped, it will attempt to craft the item it was given to craft at the start. If the craft was successful, the dropped item is of no value and is given a float value of 0 meaning the item will not be picked up again which also works as the reward for dropping the correct item (Loubos, 2018). If the craft is unsuccessful the agent determines that it either has not gathered all the items or the item that was dropped was of value. The item that was dropped then gets a value of 1 which makes an item that the agent will pick up in the next round with another new world state (Salge et al., 2018). This way the agent learns which items are of great value for the crafting recipe and which ones are not and will know how to navigate and find them. This method ultimately makes use of the RL for their AI agent.

4.2 AI agent training Method 2

In this method, they have a RL AI agent that must be able to interact with the environment and find a non-player character on the given map while navigating through a maze. The method used to train the agent is two action-advice augmented RL algorithms with one being the FA algorithm which works the same as the RL training loop that consistently evaluates the feedback it receives and compares the confidence and quality in its learned policy (Frazier & Riedl, 2019). In this method, they also removed the human trainer element from the FA algorithm and replaced it with a synthetic oracle for better accuracy (C. Nguyen, Reifsnyder, Gopalakrishnan, & Munoz-Avila, 2017). The synthetic oracle knows a lot about maze environments and knows how much action advice to give the agent and how often to give the agent the incorrect advice (Pyarelal et al., 2023).The other algorithm they use is the NAA algorithm which uses human advice in any given period especially when no other advice is implemented (Krening & Feigh, 2018b). In this case, the algorithm will be used to guide the AI agent's path. To get the agent to perform actions they use the FA algorithm to check the queued action advice the agent receives (Kanervisto, 2022).

When the agent wants to perform an action it either performs a random action or uses his Q-network to pick the best action based on previous knowledge or looks at the queued action advice (Kanervisto, 2022). The random action activates through the greedy strategy (Siljebråt, 2015). The Q-network is used by checking its confidence level, a low confidence level means it will go with the action advice (Frazier & Riedl, 2019). The NAA comes into play across certain time steps where the action advice is then recited by the agent back to itself (Krening & Feigh, 2018b). Because the agent is placed within a maze and has limited movements, it enables the NAA to work perfectly. The NAA will give the agent a direction to go which will either be north, east, south, or west. When the agent receives the instruction, it directs itself in that direction and takes steps forward (Krening & Feigh, 2019). As this is mainly an RL technique, the agent needs a reward signal to be sure it is doing the correct things. The reward signal produced gives -0.5 points every time an action is performed and +15000 when a non-player character is found. Every time the agent passes through a hedge corridor it receives +1500 points. In this way, the AI agent uses 2 algorithms that come from a bigger Learn from Advice (LFA) algorithm embodied in RL (Frazier & Riedl, 2019).

4.3 AI agent training Method 3

Team KAIROS took part in a MineRL competition where they were assigned to perform specific tasks using an AI agent. The method they used to train their agent for the competition was they proposed a hybrid solution that makes use of ML, a rule-based approach, and human feedback (Goecks et al., 2021). What they had was the Minecraft simulator provided to them and 12 Human demonstrations (Duan et al., 2022). To train the agent to perform the assigned tasks they used different modules with different training methods (Shah et al., 2022). The first thing that they had was the state machine which selects a task and breaks it down into sub-tasks using a hierarchical approach (Illanes et al., 2019). What makes the selection of a task, is the state classifier linked with the human labelling of different states (K. X. Nguyen et al., 2022). The state classifier highlights relevant states in the environment by making use of Red, Green, and Blue (RGB) information (Beyret et al., 2019). The image presented to the agent uses a classifier that labels the entire image and not just parts of the image (Jain et al., 2019). There is the possibility of multiple labels on the same image, because there may be multiple objects on the image. These labels are hard coded by humans that know all the relevant states. The label system was trained with 81 888 different images used by a graphical user interface (GUI) (Goecks et al., 2021). The state machine looks at the output of the state classifier and determines which subtask should be performed (Lin et al., 2021).

How the sub-tasks are implemented is where the hybrid solution comes in. Some of the sub-tasks were also hard-coded by humans, but they were mainly trained by human demonstrations through a navigation policy (Shah et al., 2022). They used the human demonstration to train the navigation module and use it to learn how to navigate the environment, and they achieved this by first matching the action space and then throwing away all actions not related to the movement (Xihan et al., 2022). They then end up training an entire behaviour cloning model that will then know how to navigate the environment (Kanervisto, 2022). To teach the agent how to pick up and use items they hand- engineered it by having a human play the game using the same actions the agent had access to and designed a sequence of actions to be taken to perform the sub-task (Sun et al., 2020). While that happens, they also have an estimated odometry that keeps a record of the agent’s location and certain spots (Shah et al., 2022). It can then work with the state machine to know about key features in the environment.

4.4 AI agent training Method 4

The method that MineDojo implemented was that they trained the AI agent before placing it in the world of Minecraft. The agent was pre-trained with a learning algorithm that goes through large internet-scale knowledge (Fan et al., 2022). The agent watches a large set of YouTube videos read through Wiki pages and went through Reddit posts. For the agent to be able to watch and learn from these videos, MineDojo made language a key factor and made it take the form of YouTube transcripts, the long descriptions on the Wiki pages, and the comments in the Reddit discussions (Luketina et al., 2019). The language component is then used as an open vocabulary to allow the agent to structure and understand the text and videos it works through (Allison et al., 2018). The volume of YouTube videos was 33 years' worth of content, over 6000 Wiki pages, and 340 000 Reddit posts. Measures were taken to filter out any toxic or low-quality content from their database. After the agent has completed watching and reading all the provided content, they use a multi-task RL setting where the agent performs its tasks in the MineDojo world simulation through text commands (Jansen, 2021). For the agent to understand these commands it went through multi-task RL (Allison et al., 2018). They made use of “a dense, language-conditioned reward function from in-the-wild YouTube videos and their transcripts" known as MineClip (Fan et al., 2022). MineClip is the video-language model used to translate the video snippets and language descriptions to the agent. The MineClip model is an algorithm trained on open vocabulary and various English transcripts (Zaidi, n.d.). During the RL training, the MineClip model was there to provide the necessary reward signal without making use of any adaptation techniques (Shu & Tian, 2018). MineClip also serves as an automatic evaluation metric on how well a task is performed. This made MineClip a practical module for open-ended agent learning in Minecraft using RL. How MineClip calculates its reward signal is it gathers the agents' latest 16 RGB frames and forms a video snippet out of it in the observation space (Krening & Feigh, 2018a). This way it takes all task prompts and zero-shots it which is a machine learning model that can identify things without any labels or examples (Canaan, 2018). MineClip stays consistent with this method. When the agent must perform a task, it reads the command and uses the raw pixels on the screen as input, and then takes a sequence of actions to complete the task pulling it from the action space (Krening & Feigh, 2018b). In the meantime, MineClip evaluates the action and provides the necessary signal based on the outcome (Fan et al., 2022).

4.5 AI agent training Method 5

OpenAI made use of a semi-supervised imitation learning method called VideoPreTraining (VPT). To gather data, they assigned human contractors to play Minecraft and record all their actions and decisions in the game, they also recorded the keys they pressed to play the game which is known as Behavioural Cloning (BC) for the agents (Pearce & Zhu, 2022). With this recording data, they trained an inverse dynamics model (IDM) that uses future and past information to predict actions at each step of the video (Baker et al., 2022). Once the IDM was fully trained, they used its model to learn from 70 000 hours' worth of labelled online videos posted by other players. The videos were filtered out to only be about playing the game in "survival mode" and videos that are without any artifacts such as non-game visuals, subscribe buttons, and advertisements. Learning from this data was to get the AI to perform high-level tasks in the world of Minecraft like chopping down trees, gathering wood, and picking up items. To fine-tune the capabilities of the AI and to allow it to perform more specific tasks it goes through BC (Kanervisto, 2022).

Contractors are used to play for 10 minutes in a new environment, tasked to build a house with basic materials. They recorded how the contractors had to plan and build the house from scratch to note everything that needed to be considered when building a house. For OpenAI to take a step further and teach the AI agent to be able to craft a diamond pickaxe, they realized there were a sequence of steps that needed to be followed to be able to perform this task (Guss et al., 2021). To achieve this, they implemented the use of RL from the VPT model by rewarding the agent each time it picked up the right item in the correct step of the sequence (Broll et al., 2019). The rewards were low for the start of the sequence to get bulk items and high for final step items that are frequently rare items. To aid the RL model, the VPT foundations model was trained by watching examples of players performing these tasks and they placed an auxiliary Kullback-Leiber (KL) in between the RL model and the frozen pre-trained policy to prevent the RL model to forget the skills it has learned (Trott, Zheng, Xiong, & Socher, 2019). The AI agent used this teaching sequence of training the IDM model to get the necessary data to be able to predict the next set of steps in any given scenario and to be able to watch unlabelled videos and learn from them, then be able to go through BC training to perform more specific tasks like building a house, then finally implement RL to use its reward system to complete the sequence of steps to craft a diamond pickaxe (Baker et al., 2022).

5 Discussion

In *Method 1* they were more interested in the crafting mechanics of the game and to effectively focus on this component they also had to change the environment from the original Minecraft world to something more suitable to get the results efficiently. This approach made use of a publicly available API that is specifically designed for Minecraft, and that can work with any C++, C#, and python programming. Another aspect to consider is the scale of this experiment, as it is beginner friendly for anyone to try and see what results they can get. The algorithm it uses is also easy to understand, it presents a simple way of using the elimination process to better train the agent to gather the correct items for a recipe to craft something (Justesen et al., 2019). The limitation of this approach is that it can only pick up items and craft with them and not perform the necessary task to gather the items like chopping down a tree to get wood for example. A discovery that was made was that the fewer items there were in the environment, the more time it took the agent to successfully craft something (Loubos, 2018). This presents a problem that if the agent were placed within the real world of Minecraft, it would not be able to craft because of the actions it needs to take before it can collect an item. The good thing about this method is that it gives you a basic understanding of how training an AI agent in Minecraft works and gives the freedom to the user to try out different things and the test periods are not very time-consuming, and it also does not demand a large amount of high-quality data.

In *Method 2* they focused on having the agent equipped with the correct training tools to perform any action no matter where the agent finds itself. They integrated many back-ups to ensure that, if the agent is stuck or exhibits low confidence level without advice actions queued, it has the NAA as a last resort to get it moving again to reset the algorithm of what next possible actions can be taken (Krening & Feigh, 2018b). Using this approach, they proved that any advice is better than having none and even more so than using RL methods only. It was interesting to see them implement a LFA algorithm and replace a human trainer with a synthetic oracle (Frazier & Riedl, 2019). In their case, it worked well because their environment only was a maze, if it were something more open like how the real world of Minecraft is, it would prove to be ineffective. In an open world, a human trainer would be more effective because humans are good navigators in a sandbox world like Minecraft. However, if their concern was that it puts too much demand on a human to provide the necessary action advice, it exploits a weakness in this method that it is very limited to only a maze environment and will become very reliant on the NAA from humans. This presents an interesting solution that when the agent navigates around the maze and finds itself stuck, the NAA kicks in to get it turning around and try a different route (Krening & Feigh, 2019). This also resets the confidence level the agent is working with, and so determines its next action (K. X. Nguyen et al., 2022). This way the agent keeps moving and keeps on exploring until it finds its goal. This method proves that having a backup action such as the NAA allows the agent to stay active for long which results in good data gathering until it reaches its goal. Another aspect to consider, is that it can be hard to balance the right amount of advice and agent performance. So, to implement a method like this one will need some form of experience with an LFA algorithm.

Method 3 was used in a competition with various rules and limited time and data to train the agent. They were also told what tasks to perform, and each task brings forth new challenges and factors to consider to be able to train the agent to perform these tasks (Shah et al., 2022). As we indicated in our background having an AI agent perform in a restricted environment that a competition presents, we found that will not allow the agent to perform at its best. What was interesting with this group, was firstly that they won the competition and made an agent that did present itself as a human playing the game when it had to perform its tasks. There still were instances where the training of the agent was incomplete and required further development (Goecks et al., 2021). A aspect that the team struggled with was working with the low-resolution images and human feedback, because they worked with labels to help the state classifier identify the current state the agent finds itself in and because the competition gave low-resolution images, the team occasionally found dots on the images that they could not identify and did not know what to label it with. This resulted in gathering inaccurate data when the agent used it to perform tasks. As there were many restrictions in the way the teams could train their agent, it created bigger potential once those restrictions were removed. It forced the team to use the resources as effectively as possible. Another factor to consider was the fact that, because they had such limited time, it made sense to use a tedious approach like having a human record demonstrations for each action whereas if it was meant for a larger volume of actions, this could end up taking too much time and become too big for the project to handle (K. X. Nguyen et al., 2022). Also, with the navigation policy, it could become too large for the system due to all the possible guidelines the agent might have to follow outside of the tasks they were assigned to in the competition (Xihan et al., 2022). It already proved to be a big challenge with the 12 tasks they had to perform. Expanding this training method to tasks outside of the competition could become a problem, because, with the end results of the 12 tasks given, there was still room for improvement (Shah et al., 2022), implying that this method is slow to train an agent (Goecks et al., 2021). As they used the method of imitation learning in their training, they required large amounts of high-quality data (Beyret et al., 2019). The scope of tasks that needed to be performed took up a large amount of space to get all the required data (Goecks et al., 2021). For this project improve and develop an agent with more adaptable skills, more high-quality data needed to be collected, ending up being too much to manage. One of the actions heavily relied upon long-term dependencies in a hierarchical task structured approach, making it hard to get the agent to accurately perform the tasks after doing all the necessary calculations of the environment around it (Illanes et al., 2019). This method proved that using a hybrid method for teaching an agent to perform hierarchical tasks, is a better way to train rather than using an end-to-end machine learning method which is about humans understanding the approach to sub-tasks, but provides limited feedback (Goecks et al., 2021). This method was used on a publicly available API and encourages people to start with simple training methods to perform simple tasks, building up to higher level methods.

Method 4 suggested a way of training an agent on a much different scale compared to the other methods. Their agent is very adaptable, has extreme accuracy when it must perform tasks, and has even greater potential. This is also the first method that boldly challenged the open-ended world that Minecraft presents. They wanted to achieve adaptability and make the agent capable of performing the tasks as a real player would. In the tests, the agent was capable of being a “human” in the way it acted. It covered such a large volume of training by using all the online content that the agent is able to understand and perform vague tasks such as "explore", be cautious around dangerous environments and calculate the set of actions they need to take to reach their goal. The agent can even use craft mechanics and craft items to build structures. The agent achieved an accurate human judgment metric of 97% (Fan et al., 2022). It also proved how one can use different methods together to generate an agent capable of performing these different tasks. This method makes use of a lot of different components, and the connection required between each phase was bridged with the necessary skill which is to train an AI agent to watch and read labelled videos and texts (Fan et al., 2022). This one skill opened many possibilities for the agent. The training method presents something more practical and something much more related to humans. Instead of gathering knowledge through trial and error, it learned how humans would learn how to play the game (Krening & Feigh, 2018a) and the agent can remember everything it watches (Trott et al., 2019). In this way, we can relate to the agent and be more in sync with what it is trying to achieve, making it easier to fix some of the bugs that it may present. The creative tasks that can be performed in this method is 3 times larger in magnitude than with Method 3. This method is also much more open for exploration to test what it is capable to do, which also makes it easier to correct as once you command the agent to perform a certain action and the agent cannot perform it, you can easily then just look for the YouTube video that will demonstrate and explain how to do it. This information may then easily be added to the agent's body of knowledge. This is a much more interactive way of improving the agent's performance and capabilities.

Method 5 took the challenge that Method 4 presented and made improvements. Where Method 4 required only labelled videos to learn from and text commands to perform an action, Method 5 proved that with very little labelled video training you can use that trained knowledge to be trained by unlabeled videos as well (Baker et al., 2022). In this way, they developed Method 4 further and opened the agents' learning capabilities to a bigger field of knowledge. Also, where other methods were either trying to do specific tasks or train adaptable agents, this method tried to stick to the core gameplay of Minecraft. They wanted to train an agent that could play the game from beginning to end the same way a real player would, also taking the survival mode of the game seriously. For this reason they implemented the craft mechanics features, trained an agent that performed actions by itself, adapted to the real world of Minecraft, and presented itself as the most real-life agent of all the other methods. This method introduced a hierarchical system of functions working together to complement the next level of functions to expand the agents' capabilities as it learns more (Baker et al., 2022; Broll et al., 2019). The application of the hierarchical system enabled the agent to successfully perform one of the most difficult tasks in Minecraft i.e. to craft a diamond pickaxe. For the agent to achieve this, it followed mining patterns to discover the diamond ore and return to the crafting table and equip the necessary items to perform a crafting action (Baker et al., 2022). There is planning, deciding, and then taking action involved, making this method the best method so far to make a real-life agent playing the game of Minecraft as a human would.

Overall, *Method 3* proved to be the best method to be used in the specifically given constraints of a competition. It may however become a tedious method to use once the agent is placed in a larger environment. In areas to learn about training an AI Agent with low complexities and small testing grounds, Method 1 is the best approach. It also uses Project Malmo which is a user-friendly API specifically designed for Minecraft and training Minecraft agents. Method 2 only proves that having a backup NAA helps an agent get out of situations where it was stuck, but it is not useful for learning like Method 1. Method 2 is also too small to test how far you can take AI in Minecraft. This makes Method 2 the worst method to train an agent. The best method that proved to have trained the best AI agent, that can perform the most tasks, adapt the best to its environment and navigate effectively, is the levelled-up Method 4, namely Method 5. Method 5 is also one that can continue its learning to become an even smarter agent and maybe one day outperform humans.

6 Conclusion

The aim of this paper was to categorize and identify the best method(s) to teach AI to execute basic tasks in Minecraft. We executed a SLR and thematic analysis to categorize 5 methods of training an AI agent in the world of Minecraft. We established that RL was the primary method used to teach AI agents to perform tasks in Minecraft and most methods introduced some form of human interaction to guide the agent in the learning process. With RL as the primary teaching method, there were various functions and unique ways to teach the agent and we established that training the agents by displaying a high volume of video or reading through content equipped them with the best knowledge to perform more tasks. The methods that performed well took on more complicated challenges, used larger environments, incorporated a better structured AI framework for the agents and effectively used human training elements when using RL. The findings presented in this paper can be useful to all AI and robotics engineers and programmers working with AI, as well as being applied in game-based learning.

For future research, as Methods 4 and 5 are the most developed methods, it provides possibilities for further development and future research. These 2 methods created 2 paths that can both be explored to establish what new knowledge can be extracted. With Method 4 the AI agents was prompted with natural language to perform tasks in the world and the agent completed a set of complex tasks with high accuracy. The software for this method is available to the public and this enables researchers for future research to experiment with the software and try to implement new tasks and activities to develop it further. A step further for this method would be to attach speech recognition AI to enable a natural language processing (NLP) interface. In this instance it may create a virtual friend that can perform tasks for you while playing the game, and this opens a new door for AI interaction. Method 5 has shown significant promise and set a high standard for its performance. This is an AI that can learn new things from unstructured data that was never achieved before but, the AI agent is mainly focused on just to be able to perform tasks. Future research may increase the complexity of the tasks including teaching the agent management function e.g. how to contextually manage its items collected in Minecraft. Gaining an understanding of this will not only help understand AI in the virtual world, but also in the real world and game-based learning.

References

Allison, F., Luger, E., & Hofmann, K. (2018). How players speak to an intelligent game character using natural language messages. *Transactions of the Digital Games Research Association, 4*(2).

Aluru, K. C., Tellex, S., Oberlin, J., & MacGlashan, J. (2015). *Minecraft as an experimental world for AI in robotics.* Paper presented at the 2015 aaai fall symposium series.

Amla, M. (2021). Digital Education and Society 5.0. In *Transforming Higher Education Through Digitalization* (pp. 113-130): CRC Press.

Baker, B., Akkaya, I., Zhokhov, P., Huizinga, J., Tang, J., Ecoffet, A., . . . Clune, J. (2022). Video pretraining (vpt): Learning to act by watching unlabeled online videos. *arXiv preprint arXiv:2206.11795*.

Berner, C., Brockman, G., Chan, B., Cheung, V., Dębiak, P., Dennison, C., . . . Hesse, C. (2019). Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*.

Beyret, B., Hernández-Orallo, J., Cheke, L., Halina, M., Shanahan, M., & Crosby, M. (2019). The animal-ai environment: Training and testing animal-like artificial cognition. *arXiv preprint arXiv:1909.07483*.

Boden, M. A. (2016). *AI: Its nature and future*: Oxford University Press.

Braun, V., & Clarke, V. (2012). *Thematic analysis*: American Psychological Association.

Broll, B., Hausknecht, M., Bignell, D., & Swaminathan, A. (2019). *Customizing scripted bots: Sample efficient imitation learning for human-like behavior in minecraft.* Paper presented at the AAMAS Workshop on Adaptive and Learning Agents.

Canaan, R. (2018). *Games as co-creative cooperative systems.* Paper presented at the Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment.

Deguchi, A., Hirai, C., Matsuoka, H., Nakano, T., Oshima, K., Tai, M., & Tani, S. (2020). What is society 5.0. *Society, 5*, 1-23.

Dreveck, E. A., Salgado, A. V., Clua, E. W. G., & Gonçalves, L. M. G. (2021). *Easy Learning of Reinforcement Learning with a Gamified Tool.* Paper presented at the 2021 Latin American Robotics Symposium (LARS), 2021 Brazilian Symposium on Robotics (SBR), and 2021 Workshop on Robotics in Education (WRE).

Duan, J., Yu, S., Tan, H. L., Zhu, H., & Tan, C. (2022). A survey of embodied ai: From simulators to research tasks. *IEEE Transactions on Emerging Topics in Computational Intelligence, 6*(2), 230-244.

Duncan, S. C. (2011). Minecraft, beyond construction and survival.

Dyulicheva, Y. Y., & Glazieva, A. O. (2022). *Game based learning with artificial intelligence and immersive technologies: an overview.* Paper presented at the Ceur Workshop Proceedings.

Edition, M. E. (2019). What is Minecraft: Education Edition. *Minecraft Education Edition*.

Fan, L., Wang, G., Jiang, Y., Mandlekar, A., Yang, Y., Zhu, H., . . . Anandkumar, A. (2022). Minedojo: Building open-ended embodied agents with internet-scale knowledge. *arXiv preprint arXiv:2206.08853*.

Fedorenko, E. G., Kaidan, N. V., Velychko, V. Y., & Soloviev, V. N. (2021). *Gamification when studying logical operators on the Minecraft EDU platform*.

Frazier, S., & Riedl, M. (2019). *Improving deep reinforcement learning in minecraft with action advice.* Paper presented at the Proceedings of the AAAI conference on artificial intelligence and interactive digital entertainment.

Garg, P. K., & Sharma, L. (2021). Artificial Intelligence: Challenges and Future Applications. In *Artificial Intelligence* (pp. 229-245): Chapman and Hall/CRC.

Goecks, V. G., Waytowich, N., Watkins, D., & Prakash, B. (2021). Combining learning from human feedback and knowledge engineering to solve hierarchical tasks in minecraft. *arXiv preprint arXiv:2112.03482*.

Green, M. C., Salge, C., & Togelius, J. (2019). *Organic building generation in minecraft.* Paper presented at the Proceedings of the 14th International Conference on the Foundations of Digital Games.

Guss, W. H., Castro, M. Y., Devlin, S., Houghton, B., Kuno, N. S., Loomis, C., . . . Salakhutdinov, R. (2021). The minerl 2020 competition on sample efficient reinforcement learning using human priors. *arXiv preprint arXiv:2101.11071*.

Hart, A. B. (2021). *Learning through Minecraft: A Phenomenological Study of Game-Based Instruction in Higher Education.* The Chicago School of Professional Psychology,

Illanes, L., Yan, X., Icarte, R. T., & McIlraith, S. A. (2019). *Leveraging symbolic planning models in hierarchical reinforcement learning.* Paper presented at the Knowledge Representation & Reasoning Meets Machine Learning Workshop (KR2ML@ NeurIPS).

Jain, U., Weihs, L., Kolve, E., Rastegari, M., Lazebnik, S., Farhadi, A., . . . Kembhavi, A. (2019). *Two body problem: Collaborative visual task completion.* Paper presented at the Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.

Jansen, P. A. (2021). A systematic survey of text worlds as embodied natural language environments. *arXiv preprint arXiv:2107.04132*.

Justesen, N., Bontrager, P., Togelius, J., & Risi, S. (2019). Deep learning for video game playing. *IEEE Transactions on Games, 12*(1), 1-20.

Kanervisto, A. (2022). *Advances in deep learning for playing video games.* Itä-Suomen yliopisto,

Kanervisto, A., Milani, S., Ramanauskas, K., Topin, N., Lin, Z., Li, J., . . . Yang, W. (2022). Minerl diamond 2021 competition: Overview, results, and lessons learned. *NeurIPS 2021 Competitions and Demonstrations Track*, 13-28.

Kich, V. A., de Jesus, J. C., Grando, R. B., Kolling, A. H., Heisler, G. V., & Guerra, R. d. S. (2022). Deep Reinforcement Learning Using a Low-Dimensional Observation Filter for Visual Complex Video Game Playing. *arXiv preprint arXiv:2204.11370*.

Kope, A., Rose, C., & Katchabaw, M. (2013). *Modeling autobiographical memory for believable agents.* Paper presented at the Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment.

Krening, S., & Feigh, K. M. (2018a). *Characteristics that influence perceived intelligence in AI design.* Paper presented at the Proceedings of the human factors and ergonomics society annual meeting.

Krening, S., & Feigh, K. M. (2018b). Interaction algorithm effect on human experience with reinforcement learning. *ACM Transactions on Human-Robot Interaction (THRI), 7*(2), 1-22.

Krening, S., & Feigh, K. M. (2019). *Effect of interaction design on the human experience with interactive reinforcement learning.* Paper presented at the Proceedings of the 2019 on Designing Interactive Systems Conference.

Lin, Z., Li, J., Shi, J., Ye, D., Fu, Q., & Yang, W. (2021). Juewu-mc: Playing minecraft with sample-efficient hierarchical reinforcement learning. *arXiv preprint arXiv:2112.04907*.

Loubos, D. (2018). *Automated testing in virtual worlds.*

Luketina, J., Nardelli, N., Farquhar, G., Foerster, J., Andreas, J., Grefenstette, E., . . . Rocktäschel, T. (2019). A survey of reinforcement learning informed by natural language. *arXiv preprint arXiv:1906.03926*.

Milani, S., Topin, N., Houghton, B., Guss, W. H., Mohanty, S. P., Nakata, K., . . . Kuno, N. S. (2020). *Retrospective analysis of the 2019 MineRL competition on sample efficient reinforcement learning.* Paper presented at the NeurIPS 2019 Competition and Demonstration Track.

Murtiningsih, R. S., & Ainia, D. K. (2022). *The Realization of Traditional Children's Game-Based Education in Facing Educational Challenges in Era 5.0.* Paper presented at the ICONSEIR 2021: Proceedings of the 3rd International Conference of Science Education in Industrial Revolution 4.0, ICONSEIR 2021, December 21st, 2021, Medan, North Sumatra, Indonesia.

Nebel, S., Schneider, S., & Rey, G. D. (2016). Mining learning and crafting scientific experiments: a literature review on the use of minecraft in education and research. *Journal of Educational Technology & Society, 19*(2), 355-366.

Nguyen, C., Reifsnyder, N., Gopalakrishnan, S., & Munoz-Avila, H. (2017). *Automated learning of hierarchical task networks for controlling minecraft agents.* Paper presented at the 2017 IEEE Conference on Computational Intelligence and Games (CIG).

Nguyen, K. X., Bisk, Y., & Iii, H. D. (2022). *A framework for learning to request rich and contextually useful information from humans.* Paper presented at the International Conference on Machine Learning.

Oates, B. J., Griffiths, M., & McLean, R. (2022). *Researching information systems and computing*: Sage.

Oliveira, B. A., Martins, C. A. d. S., Magalhaes, F., & Goes, L. F. W. (2019). Difference based metrics for deep reinforcement learning algorithms. *IEEE Access*, *7*, 159141-159149.

Ontanón, S., Synnaeve, G., Uriarte, A., Richoux, F., Churchill, D., & Preuss, M. (2013). A survey of real-time strategy game AI research and competition in StarCraft. *IEEE Transactions on Computational Intelligence and AI in Games*, *5*(4), 293-311.

Palanisamy, P. (2018). *Hands-On Intelligent Agents with OpenAI Gym: Your guide to developing AI agents using deep reinforcement learning*: Packt Publishing Ltd.

Pearce, T., & Zhu, J. (2022). *Counter-strike deathmatch with large-scale behavioural cloning.* Paper presented at the 2022 IEEE Conference on Games (CoG).

Pyarelal, A., Banerjee, A., & Barnard, K. (2023). *Modular Procedural Generation for Voxel Maps.* Paper presented at the Computational Theory of Mind for Human-Machine Teams: First International Symposium, ToM for Teams 2021, Virtual Event, November 4–6, 2021, Revised Selected Papers.

Salge, C., Green, M. C., Canaan, R., & Togelius, J. (2018). *Generative design in minecraft (gdmc) settlement generation competition.* Paper presented at the Proceedings of the 13th International Conference on the Foundations of Digital Games.

Samsudin, M. A., Ming, G. K., Ahmad, N. J., & Abrose, Y. (2022). Levelling Up Primary School Students' 21st Century Skills Through Minecraft-Game-Based Learning. In *Handbook of Research on Acquiring 21st Century Literacy Skills Through Game-Based Learning* (pp. 750-770): IGI Global.

Shah, R., Wang, S. H., Wild, C., Milani, S., Kanervisto, A., Goecks, V. G., . . . Mills, E. (2022). Retrospective on the 2021 BASALT Competition on Learning from Human Feedback. *arXiv preprint arXiv:2204.07123*.

Shu, T., & Tian, Y. (2018). M3RL: Mind-aware Multi-agent Management Reinforcement Learning. *arXiv preprint arXiv:1810.00147*.

Siljebråt, H. (2015). MAIA: The role of innate behaviors when picking flowers in Minecraft with Q-learning.

Singh, S. (2020). *Minecraft as a Platform for Project-Based Learning in AI.* Paper presented at the Proceedings of the AAAI Conference on Artificial Intelligence.

Sun, S.-H., Wu, T.-L., & Lim, J. J. (2020). *Program guided agent.* Paper presented at the International Conference on Learning Representations.

Sunarya, P. A. (2022). Machine Learning and Artificial Intelligence as Educational Games. *International Transactions on Artificial Intelligence, 1*(1), 129-138.

Thawonmas, R., Togelius, J., & Yannakakis, G. N. (2019). *Artificial general intelligence in games: Where play meets design and user experience.* Paper presented at the NII Shonan Meeting.

Togelius, J. (2019). *Playing smart: On games, intelligence, and artificial intelligence*: MIT Press.

Trott, A., Zheng, S., Xiong, C., & Socher, R. (2019). Keeping your distance: Solving sparse reward tasks using self-balancing shaped rewards. *Advances in Neural Information Processing Systems, 32*.

Wang, F.-Y., Zhang, J. J., Zheng, X., Wang, X., Yuan, Y., Dai, X., . . . Yang, L. (2016). Where does AlphaGo go: From church-turing thesis to AlphaGo thesis and beyond. *IEEE/CAA Journal of Automatica Sinica, 3*(2), 113-120.

Wichlacz, J., Torralba, A., & Hoffmann, J. (2019). *Construction-planning models in minecraft.* Paper presented at the Proceedings of the ICAPS Workshop on Hierarchical Planning.

Woof, W., & Chen, K. (2018). *Learning to play general video-games via an object embedding network.* Paper presented at the 2018 IEEE Conference on Computational Intelligence and Games (CIG).

Xihan, B., Mendez, O., & Hadfield, S. (2022). *SKILL-IL: Disentangling Skill and Knowledge in Multitask Imitation Learning.* Paper presented at the 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).

Yannakakis, G. N., & Togelius, J. (2014). A panorama of artificial and computational intelligence in games. *IEEE Transactions on Computational Intelligence and AI in Games, 7*(4), 317-335.

Zaidi, A. (n.d.). CS229 Final Project: Language Grounding in Minecraft with Gated-Attention Networks. *Network, 1*, 0.

Zhadan, A. (2018). Artificial Intelligence Adaptation in Video Games. In.