

A NEAT Approach to Wave Generation in Tower Defense Games

Daniel Hind
DMTLab

School of Computing and Digital Technology
Birmingham City University, B4 7XG
Email: daniel.hind@mail.bcu.ac.uk

Carlo Harvey
DMTLab

School of Computing and Digital Technology
Birmingham City University, B4 7XG
Email: carlo.harvey@bcu.ac.uk

Abstract—Neural networks have shown promise when applied to video games and have proven effective at performing tasks such as dynamic difficulty adjustment (DDA). This paper explores how an evolving neural network can be applied to a tower defense game in order to generate dynamic content with the intent of increasing player engagement through the principals of flow. A NeuroEvolution of Augmenting Topologies (NEAT) neural network (NN) was trained as a wave manager to observe the current game state and generate an opposing enemy wave which best challenges the players' current tower defenses. The resulting network was compared against manually designed human waves in a blind A/B test using the Games Experience Questionnaire (GEQ) to evaluate the waves across a range of criteria. The results show that an approach like this could be viable if used for content generation purposes as no discernible difference existed in reported player experience between AI and human designed waves. However, the findings regarding subsequent increases to player engagement were inconclusive. More research is required in this field to conclusively determine if machine learning generated content can exceed the quality of content created by human designers, but the findings of this paper indicate that this approach may prove valuable to game developers in the near future by allowing them to save time and money by having AI generate content instead of requiring costly human game designer time.

I. INTRODUCTION

This paper aims to explore Artificial Intelligence (AI) techniques when applied within video games. We investigate whether it is possible to generate dynamic content that is similar to content developed by a game designer and separately, investigate the effect on player engagement across these conditions. A machine learning driven AI wave manager was built for a tower defense game using NEAT and has been compared against human designed waves in order to determine its viability as a content generation system.

Video game content can be expensive and time consuming to develop, hand crafting each encounter the player may experience and manually tuning difficulty for a range of differing skill levels may take up a large part of any game designer's work time, and yet players may still reach the end of this finite content and find themselves craving for more. This project aims to investigate whether the possibilities of Artificial Intelligence (AI) can be applied to ease this strain on development and dynamically generate engaging content for players to experience. This work can lead to reducing the

amount of human work hours required to create content. The AI generated content has the potential to be more dynamic and more engaging than finite manually designed content, therefore also causing the end-product to be endlessly re-playable and far more valuable to the player base, and reciprocally, game studios.

This project adapts the output into a Tower defense (TD) paradigm, many other game genres use a similar AI enemy waves vs player structure, such as a range of base building and action games. Another reason why a TD game was chosen as the basis of this project is due to the research undertaken by Olesen *et al.* when investigating the use of Dynamic Difficulty Adjustment (DDA) using rtNEAT within a real-time strategy game [1]. Their findings suggested that games that have easily observable game states and simple goals work best for this approach, and a TD game fits that criteria perfectly so are an ideal candidate for a similar ML DDA technique.

The aim of this project is to explore if an evolving neural network can be effectively trained and applied to a tower defense game to generate enemy waves which challenge the player's strategy, with the goal of increasing player engagement. The specific contributions are as follows:

- An implementation of a wave director using a NEAT NN which can observe the current state of the game and make informed decisions about which enemies to spawn in the next wave;
- An evaluation of whether player engagement was increased by the addition of the wave manager by using a contrasted A/B testing method;
- An analysis of whether the AI generated content is better than or comparable to human designed content to conclude whether dynamic AI content generation can be used to minimise human designer time;
- Suggestions for practical uses cases via a cogent discussion.

II. RELATED WORK

Artificial Neural Networks (ANNs) come in a wide range of varieties and are often specialised for their desired purpose, such as: Convolutional Neural Networks (Krizhevsky *et al.* [2]), which are designed for image classification; Dynamic Recursive Neural Networks (Guo *et al.* [3]), a modern design

that excels in computer vision tasks; and Long/Short Term Memory Networks (Hochreiter and Schmidhuber [4]), which are ideal for speech recognition tasks.

NEAT is an established and effective method for neuro-evolution which is unique as it allows ANNs to not only evolve their weights and biases, but also their topologies [5]. This allows for NNs to be evolved rapidly. NEAT has since been iterated on, and has even proven to be viable to be used in real-time video games to achieve various goals such as automated content generation using cgNEAT [6], and dynamic difficulty adjustment using rtNEAT [1], in a Real-Time Strategy Game. ANNs have also been used in video games to perform a variety of tasks such as DDA, demonstrated by Li *et al.* [7] and Ebrahimi and Akbarzadeh-T [8]. Genetic techniques have also been used in wave design for games [9]. For a recent SOTA report on this field, please see the work by Risi and Togelius [10].

Another key theme for this work is Dynamic Difficulty Adjustment (DDA), in which a video game can dynamically change its difficulty in real-time to match the player's skill level. This can allow for greater player engagement by ensuring that players are constantly met with a fair, consistent level of challenge. DDA has already proven to be successful and has been implemented into a range of well-known games. The idea of using a brain-like topology to perform complex computation was first conceived by McCulloch and Pitts [11] and has since gone on to be a fundamental form of machine learning. By learning from given examples, ANNs can be trained to solve a wide range of problems such as decision making, prediction, and classification.

Automated Game Design (AGD) is a broad field, but it generally involves the concept of using AI to generate content for video games, or even entire games from scratch. AGD has been used in the past to combine existing game systems in order to create new games using machine learning [12], and to generate two-dimensional level layouts inspired by classic games [13]. AGD is important as it allows developers to reduce hands-on designer time required on a given project. Automated systems can also be used to create endless procedural playable content that can increase the value of a game and allow for content that would not otherwise be possible through manual creation. Recently, Wave Function Collapse techniques have been implemented in the area of Procedural Content Generation to successfully produce 3-dimensional meshes as game content [14]. This technique has been bolstered in its utility with design-level constraints embracing knowledge of tile-based connective structures of the used imagery [?].

Csikszentmihalyi proposed the concept of "flow", meaning that people enjoy activities most when they strike a balance between their own skill and the challenge at hand; if a task is too difficult, players may feel frustrated or anxious, but if it is too easy then players could feel bored and disengaged [15]. As Zohaib explains [16], this concept can be applied to difficulty in video games to ensure that players are kept engaged throughout the course of a game. However, video games traditionally rely on pre-set difficulties such as Easy,

Medium, and Hard, which the player must select at the start of a game, but these static difficulties are often restrictive and are not nuanced enough to account for a wide range of skill levels. The tower defense game genre in particular could benefit from the use of DDA. In a typical tower defense game, the player is tasked with placing defenses in order to fight back waves of pre-planned enemies. Tower defense games are generally pure strategy games where levels are predictable and can be solved once a player discovers a viable strategy. This can lead to players feeling disengaged once they find a strategy that works as there is no incentive to deviate.

Sutoyo *et al.* attempted to tackle this problem by creating a traditional DDA system for a wave-based tower defense game with the goal of ensuring that players of all skill levels are always presented with a suitable challenge [17]. During gameplay, their DDA system looks at key in-game metrics to determine the player's skill level and adjusts the difficulty accordingly between waves. Difficulty is adjusted by changing three difficulty scalars: the status point effects the power and health of enemies, the gold point effects how much currency the player generates at the end of a wave, and the spawn point effects how many enemies are spawned. By using hand-picked scalars for difficulty adjustment, the system is still reliant on a designer having set these multipliers to appropriate values, otherwise the system suffers from difficulty swings, or the system may not adjust significantly enough to cater for a wide range of player skill.

The proposed system considers three in-game metrics when deciding how to adjust difficulty. The generic metrics of player health, enemy health, and skill points may not give the system the full picture of how competent a player actually is. For example, the metrics of player and enemy health are simply side effects of how the player has structured their defense, so it would likely be more valuable to evaluate the tower placement itself. A machine learning method can be used instead in order to evaluate tower placement directly and observe tower synergies, which could then in turn produce a more accurate estimate for actual player strategy and allow for more fine-tuned DDA via wave generation and management.

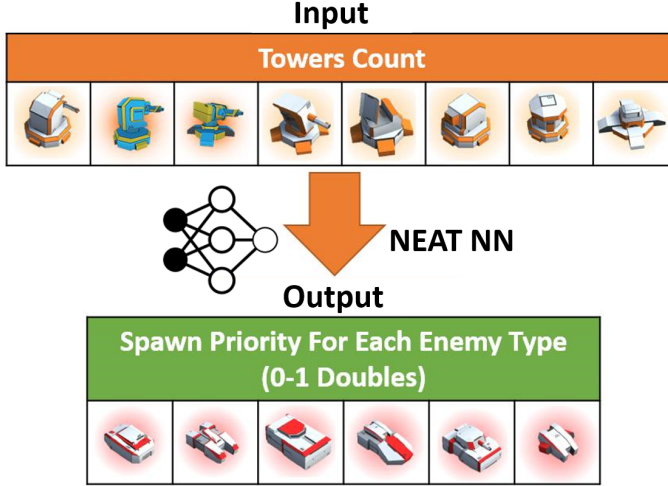
III. METHODOLOGY

The goal for this project is to create a wave manager driven by a NEAT network which is trained to analyse the player's defenses in a tower defense game in order to decide which enemies have the best chance of success. The wave manager will have a certain amount of points to spend on spawning enemies, and the amount of allocated points will be driven by traditional difficulty increments and the current wave count. These points are used to ensure the wave manager is still fair and doesn't generate any unbeatable waves. The overview of this can be visualised in Figure 1.

A. Game Development and Project Setup

The Unity game engine (2020.2.1f1) was used to build the game, this project took advantage of game templates available to download for the engine. SongGameDev's Tower

Fig. 1. Illustration of the inputs and outputs for the neural network for the tower defense game



Defense Toolkit 4 (TDTK-4) was used as a basis for the game as it contains a comprehensive set of tools and assets that can be used to quickly build a generic tower defense game. UnitySharpNeat was used to implement the NEAT ANN, this contains most of the necessary features required to begin training an ANN immediately with some additional project specific set-up required.

1) *Units and Towers*: Throughout development and testing it became clear that certain strategies were favoured by the network in generating waves due to imbalance in the features of individual units, and indeed towers in the game design.

The wave unit spawn costs and stats were adjusted to make them feel fair and less challenging as some units were more powerful than expected when used against the player tower strategies. These are presented in Tables I and II with ratios of units chosen presented in Figure 4.

2) *NEAT Management and Fitness*: An AIWaveManager class was created in order to feed the NEAT agents their required inputs and to handle their outputs, as well as feeding back data regarding the outcome of the wave which was then used to determine network fitness.

The fitness function of the ANN should be determined by the following factors and is presented in Algorithm 1:

- The number of units that make it to the end of the level thus dealing damage to the players health pool. The ANN will be rewarded for each point of damage dealt.
- Average distance travelled by units. This will encourage the ANN to learn that having units survive for longer will increase the chance of damage being dealt.
- Severe punishment if the number of units requested to spawn exceeds the number of available spawn points.

ruled

The existing SpawnManager class within TDTK-4 initially required all waves to be defined at the start of a game so this was modified to allow for new waves to be inserted at

Input: P, U // Path of defence and List of units

Output: R // Average Path Completion

Function GetAveragePathCompletion(P, U):

```

    fTotComp ← 0;
    foreach  $U \in Units$  do
        fTotComp ←
            fTotComp +  $P.U.PathCompletion$ ;
    end
    iNumSpawned ←  $U.Len()$ ;
    if  $iNumSpawned = 0 || fTotComp = 0$  then
         $R \leftarrow 0$ ;
    end
     $R \leftarrow fTotComp / iNumSpawned$ ;
     $R \leftarrow Clamp(R, 0, 1)$ ;
    return  $R$ ;

```

End Function

Input: W // Wave applied to Path

Output: F // Fitness Function

Function GetFitness(W):

```

    fMaxFitness ← 100;
    fAvgComp ←
        GetAveragePathCompletion( $P, U$ );
     $F \leftarrow fMaxFitness * fAvgComp$ ;
     $F \leftarrow Clamp(F, 0, fMaxFitness)$ ;
    return  $F$ ;

```

End Function

Algorithm 1: GetFitness implementation details

runtime from the AIWaveManager class. This was set up so that whenever a new wave was created the AIWaveManager would then let the NEAT supervisor know that a new wave has begun, clear any fitness trackers from the previous wave, then request each NEAT agent in the scene to generate a wave for their associated TD tower path. This project used ten NEAT agents as can be seen in Figure 2.

The NEAT agents took the form of a NeatGameWaveGeneratorUnit class which handled all of the inputs, outputs and fitness for a given agent. The specifics of this classes function is to return a double array [0-1] representing the priority for which enemy units to spawn at the start of a given wave. When the AIWaveManager requests a wave to be generated it will supply each NEAT agent with an associated path, the agent will then register the inputs relevant to that path by querying the game state, then the agent activates itself and evaluates the neural network to generate an output array representative of the wave to spawn.

B. Training

The AI wave manager ultimately settled on a network where the inputs are the number of each tower type in the level and the outputs are the spawn priority for each enemy type, the







Icon	Name	Spawn Cost	Health	Shield	Speed	Armour	Special
	Transport	1	17	0	1.5		
	Speeder	3	15	10	2.5	Physical	
	Support	5	50	30	1.5		50% resistance to nearby units
	Carrier	7	40	40	1	Electric	Spawns a drone when killed
	Tank	25	650	0	0.5		
	Drone	3	1	10	2		Flying: can't be slowed

TABLE I
SHOWING THE WAVE UNIT STATS THAT WERE USED IN GAME DEVELOPMENT AND FOR TRAINING THE AI.







Icon	Name	Cost	Damage	Cooldown	Range	Crit	Damage Type	Special
	Machine Gun	10	1-2	0.75	2	25% chance 1.25x multiplier	Physical	
	Zapper	15	3	1.25	2.5	10% chance 2x multiplier	Electric	10% to stun on hit
	Laser	20	2-3	1	3	10% chance 2x multiplier	Electric	Damage over time 6 damage / 3s
	Cannon	25	8-10	2	3.5	0% chance	Physical	Area of effect
	Slow	25			2.25			Slows enemies in range 40% slow speed
	Support	30			3			Fire rate increase 25% attack speed

TABLE II
ILLUSTRATING FINAL TOWER STATS THAT WERE USED TO TRAIN THE NEAT NETWORK.

fitness was entirely determined by the average path completion percentage of all enemies spawned that wave.

During training, each of ten lanes was given a unique tower layout and used ten trials so that each NEAT agent now performed against each tower layout before having fitness evaluated. This meant that networks were evaluated based on their mean fitness at the end of the trials, so it ensures that the overall best network always survives. The tower layout of each lane was designed to allow the network to learn the strengths and weaknesses of each tower type and to also test it against different combinations of towers. Each of the damage dealing towers has its own lane where only that tower is used so that the network can learn which units are best to defeat those towers, some lanes have only a single damage type meaning that the network should adapt to spawning the

associated damage type resistant enemy against those lanes. The rest of the lanes are made to represent the sort of real defense layouts that players will be using in the final game, these test if the network is able to use a mix of units to best tackle those complex defense layouts. The finalised tower layouts used can be seen in Figure 2 where the training process is also shown.

As shown in Figure 1, the input for the network is the integer values representing how many of each of the tower types are currently in the level, upgraded towers were counted twice as they are effectively twice as impactful as a single tower. The output is once again a [0-1] double for each of the 6 enemy types, this represents the spawn priority of the given enemy type. To determine how many of each enemy type to spawn, the available spawn points for the wave are distributed across

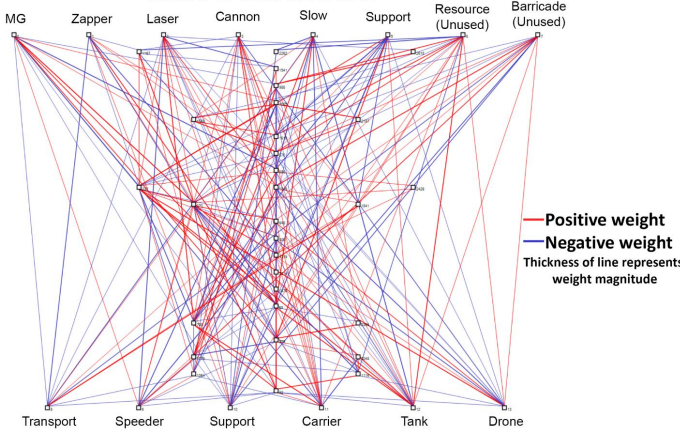
Fig. 2. Illustration of five NEAT agents, concurrent training within Unity of the NEAT network used for wave generation.



the enemy types proportionally to their given spawn priority, any unspent points are then redistributed to the next highest priority enemy type until all points are spent.

After approximately 5000 generations the network began to consistently defeat the training lanes so learning plateaued with fitness reported between 95-100, at this point the tower layouts were made significantly harder to challenge the network and the network was trained for another 500 generations where fitness stayed relatively consistent. The finally selected champion network had a fitness of 56.3 against the harder tower layout and when internally playtested against was sufficiently challenging and dynamic, the final topology for the network selected can be seen in Figure 3.

Fig. 3. Graph showing the topology for the trained NEAT network used for wave management. Inputs at the top, outputs at the bottom.



C. Evaluation

The two key metrics to measure to evaluate the success of this project are:

- Content Generation: is the system able to generate content that is considered similar to human generated waves?
- Player Engagement: are players differently engaged when playing against the AI wave manager than against human designed waves?

A/B testing was used to effectively evaluate changes in player engagement. Winkler [18] demonstrated that a minimum of 10 participants are needed in each test group. Half of the participants played the manually designed waves while the other half played the AI waves.

Test participants were given a set of levels in the game to play and were instructed to play for 25 minutes or until they completed all three levels. Due to the ongoing (at the time) COVID-19 pandemic, this testing was conducted remotely, so participants were given a link to download the game to their local machines, a participant information leaflet and web consent form and played from the comfort of their own home while entirely unsupervised. Immediately after the play session participants were surveyed using the Game Experience Questionnaire (GEQ) developed by Ijsselstein *et al.* [19]. The GEQ Core Module was used in order to measure how they felt about the game and how engaged they were throughout. The most important areas of the survey to observe will be questions relating to Flow, Tension/Annoyance and Challenge as these are the areas that should be most affected by the AI wave manager.

IV. RESULTS AND DISCUSSION

A total of 22 people participated in the survey, meaning that each trial (A or B) had 11 participants each. As an inclusivity criteria, every participant had experience playing at least one tower defense game previously meaning they had at least some knowledge of the genre.

GEQ Total Scores							
	Co	SaII	FL	T/A	Ch	NA	PA
AI	149	152	107	18	72	23	179
Human	120	115	86	33	86	32	167
p-value	0.19	0.06	0.21	0.19	0.41	0.34	0.38

TABLE III

TABLE DEMONSTRATING THE TOTAL SCORES FROM THE GEQ A/B TESTING SURVEY WITH 11 PARTICIPANTS FOR EACH GROUP (AI AND HUMAN WAVES). THIS IS COMPLEMENTED WITH A T-TEST TO DETERMINE SIGNIFICANCE. CO = COMPETENCE, SaII = SENSORY AND IMAGINATIVE IMMERSION, FL = FLOW, T/A = TENSION/ANNOYANCE, CH = CHALLENGE, NA = NEGATIVE AFFECT, PA = POSITIVE AFFECT.

Table III shows the final results from the survey and demonstrates the total scores achieved for both the AI and Human across all seven aspects of the GEQ. Overall, the results are favourable for the AI with it achieving higher scores in positive aspects such as competence, sensory and imaginative immersion, flow and positive affect. The AI also scored lower in the negative aspects of tension/annoyance and negative affect. Though unexpectedly the AI scored slightly lower than the human waves for the challenge aspect meaning that the AI wave manager might not be challenging players as much as it was expected to.

One finding from this survey is that the AI waves scored 107 in the flow category whereas the human waves only

scored 86, meaning that the AI waves were generally more engaging than the human ones. Another key category was tension/annoyance where the AI scored a total of 18 compared to the 33 of the human waves. This could mean that players were generally less frustrated when playing against the AI which likely contributes to the higher flow score too. There was a serious risk that the AI would score higher here due to the fact that it would be less predictable and intentionally counter player strategy over the human designed wave.

Another result is that the AI scored slightly lower for the challenge aspect with a score of 72 compared to the 86 scored by the human waves. While a lower challenge score isn't necessarily a bad thing, it certainly wasn't expected. The entire concept behind the AI wave manager's implementation was that it could be trained to adapt to player strategies and therefore always produce a challenging wave, thus theoretically increasing levels of engagement through the principles of flow. However, in this case challenge decreased but flow still increased. These lower than expected challenge scores may also go some way to explaining why the related tension/annoyance score are also lower than anticipated.

At a glance the results seem favourable for the AI wave manager, however a deeper look at data in Table III shows that the results are not statistically significant when comparing the two groups. This leads us to accept a null hypothesis that means are equal under testing and that there is no difference between these groups. On the other hand, whilst no group can be determined better or worse than one another with this sample, this does demonstrate that in terms of the GEQ and player experience, the AI generated waves were considered not statistically different to the human generated waves against the dependent variable of Game Experience Factors. This illustrates that the NEAT guided wave management system can perform the function of a human wave designer, delivering similar game experience to a player audience.

A. Engagement

One of the key goals of this project was to determine if an evolving neural network could be used to increase player engagement within a video game. The results from the blind A/B test GEQ survey failed to prove any statistically significant improvement in player flow. However, the results are potentially still promising as the AI waves did score higher than the human designs for flow and the lack of statistically significant findings at least proves that the AI is comparable to the human designs.

The plan for increasing engagement was to use the ANN to ensure that the player is constantly being challenged thus increasing engagement due to the theory of flow. However, the survey results show that players generally found the AI waves to be less challenging than the human waves, meaning that the AI didn't really achieve its goal. There are a few factors that may have contributed to the AI being less challenging than the human waves: the ANN might not have been trained for long enough, the method of training may not have been

representative of actual gameplay, or the fitness function used may not have been sufficient thus training bad habits.

Another potential reason may be due to the way the AI structures its waves. The AI is able to spawn any enemy type on any wave (assuming it has the spawn points to afford it), which means that it's possible that the AI wave manager does a better job at naturally introducing each enemy type to the player earlier in the game compared to the human waves which don't introduce some of the harder units until later waves. This may lead to players playing against the AI waves to develop better strategies earlier in the game as any weaknesses in their defences are exposed earlier in the game where the punishment for letting your defences get overwhelmed is far less significant and less likely to lead to a game over compared to later in the game when the human designed waves introduce certain enemies such as the challenging carrier unit.

Even though the AI waves were perceived as being less challenging than the human waves, the AI still scored higher in flow. This may mean that the human waves were too challenging as they also had a higher score for tension/annoyance, so therefore lowering the challenge could have increased flow if the high level of challenge was the limiting factor at the time. While the AI and human waves conform to the same spawn points limitation in an attempt to keep them balanced, there are other factors that may affect the difficulty of the human waves. Unlike the human waves, the AI waves are structured with a constant 1 second gap between each enemy being spawned and enemies are spawned from highest cost to lowest cost, meaning that the structure and timing is predictable and consistent across all AI waves. The human waves however are intentionally designed to take advantage of the natural synergies between certain enemy types, such as combining support units with closely grouped collections of transport units, combinations like this are possible to generate with the current AI implementation, but are not curated by design intervention, so it is possible that these specifically designed combinations and unique wave timings are contributing to the increased challenge of the human waves.

To conclude whether an AI approach like this can be used to improve engagement, the results are promising but inconclusive, it may still be viable with further research. There are many factors at play when it comes to increasing engagement and the results can be unpredictable, so each game to use this approach would likely need an entirely bespoke and fully tested system. This may prove to be too expensive and risky for some game developers, though if a game were to be designed with this method in mind from the start it is possible that it could benefit greatly from this approach.

B. Content Generation

The second goal of this project was to determine if an ANN could be used to generate valuable game content, and this project has succeeded in achieving that goal. Though as stated previously the survey results were inconclusive, this also proves that the AI generated waves were at the very least statistically indifferent from the human waves meaning that the

AI generated content was at least as enjoyable as the human content and is therefore valuable for automated game design.

The main advantage of an AI content generation system such as the one developed for this project, is that it allows for endless content generation and endless replayability. As the AI generates waves in real time while the player is playing, there is no limit to the amount of content that can be generated so the potential playtime is theoretically limitless which is often an important metric for perspective players, therefore this approach could increase revenue streams via replayability. Another important advantage is that time and money can be saved during development by having the AI handle all of the wave generation while real human designers focus on other areas of the project, designing unique waves for each level and difficulty can prove to be an extremely time consuming process. The ANN developed for this research benefits from not requiring to be trained specifically for each level, meaning that it works for rapid iteration and would allow for level designers to playtest their levels early on without needing designers to curate waves, this is another potential development time and cost saving. Often a downside of automated content generation is that designers lack the ability to fine tune content, though the developed system can still allow for designers to tweak parameters such as the allowed spawn points to spend and the individual cost for each unit. It would also be easily possible to add additional tweakable parameters such as blocking certain units type from spawning until specified waves or changing the wave timings.

However, there are some potential downsides to this approach. The initial training of the network took a significant amount of time and could prove to be a bottleneck in the often parallelised game development process as the rest of the game mechanics need to be in place before the network can begin learning. More complex games with more features and mechanics would likely require more training time and may required more bespoke automated training configurations, particularly if the player can impact the wave while it's in progress. Updating games with balance changes or new content post-release is a common practice throughout the industry, but making any changes to the game will also require the network to be retrained, for small balance changes this shouldn't be too costly but larger changes particularly with new towers or enemies would require the entire network to be retrained rather than fine-tuned.

A potential improvement for this approach could be to rather than allowing the AI to spawn any enemies it wants by generating a wave from scratch, to instead have it pick from pre-set human designed subwaves. For example, a subwave of support and transport units with their wave timings set effectively or a subwave of alternating physical and electrical resistant enemies. This could allow for the AI generated content to still have some of the benefits of the human designed content (customised wave timings, use of designed enemy synergies, and more unit variation) while still allowing for potentially endless content generation. This would come with the downside of more human design time being required,

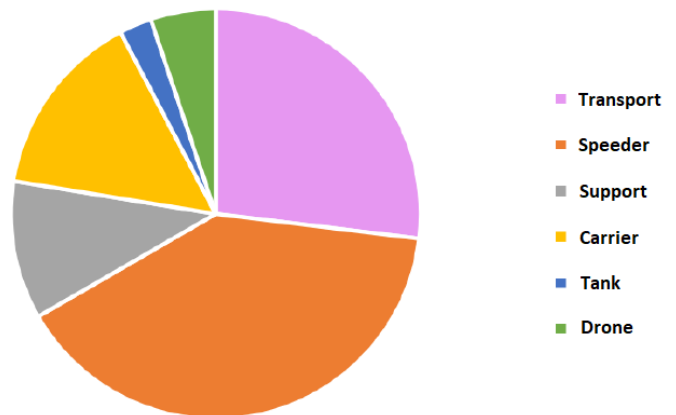
but higher quality content may make this worthwhile. Another potential improvement could be to allow designers to set which enemy types are allowed to spawn on which waves, this would allow for a more natural designed difficulty curve as new units could be introduced once at a time as is typical for the genre.

In summary, while the specific system developed for this research may not be suitable for use in a real product, a similar purpose-built system could definitely be viable for use within video games to generate endless content. This research has proven that AI generated content can be indistinguishable from human designed content, meaning that it can provide value to developers that wish to save designer's time.

C. Future Use Cases

Beyond the uses discussed already, there is potential for this methodology to be applied to other areas within games. One potential avenue could be to use a machine learning approach like this as a tool to aid game designers in balancing a game. Throughout the training process of the tower defence example explored for this project it was observed that the AI had a preference for certain enemy types and rarely chose others, this helped fine-tune the enemy stats to ensure that the units were more balanced. Figure 4 shows the proportions of each enemy type spawned in the final network and data like this is very useful to game designers looking to create a balanced experience. In theory, a game designer could use a tool like this to make balance changes to a game, then run the training process until the fitness plateaus, then run the resulting champion network and harvest data from it to observe how the balance changes effected the game when compared to the results prior to the change. By running these simulated waves constantly this could also be seen as a form of automated testing and additional analytics could be recorded in order to provide more insight into the health of the software.

Fig. 4. Proportion of each enemy spawned by the final champion network when being run through the 10 test lanes.



Beyond tower defence games this approach could easily be mapped to many other prominent game genres where the player competes against an AI opponent such as real time

strategy games, base building games, or horde style games. Other games such as Left 4 Dead 2 (2009) already use a similar system of observing the current game state when procedurally generating enemy waves, but those encounters are predetermined and limited in their nature, whereas a machine learning approach could allow for this content to be generated infinitely with the resulting enemy waves being far less predictable and repetitive.

AI generated content like this doesn't necessarily have to be limited to the wave generation demonstrated in this project. Given the somewhat promising results of this experiment it is possible that a similar machine learning approach of allowing a network to observe the current game state and make informed decisions could be applied to other areas of game development. Perhaps an ANN could be trained to generate rooms and scenarios in a dungeon-crawler style game, or maybe it could generate dynamic quests in a role-playing game, the potential is near limitless. This approach could also be a great way to incorporate DDA within games as the AI can observe the game state to see how well the player is doing and adjust difficulty accordingly.

V. CONCLUSION

This paper explores whether an ANN can be used to generate engaging video game content via wave management for a TD game. The result means showed that while this technique did increase overall player engagement, the game's perceived difficulty was actually reduced by this addition. Meaning that the link between challenge and engagement derived through the principals of flow did not apply as expected, so more research is required to further evaluate this approach. Whilst means were different across approach, there was no statistically significant difference between the human and AI generated waves. Whilst no significant engagement increase or decrease was observed, this result proves that this approach would be viable as a form of dynamic content generation as the AI waves are effectively indistinguishable from human designed ones. This research also shined a light on other potential uses for this technology, such as being used as a tool for game balance and automated testing. In summary, there is a lot of potential for this technology to drive innovation within the games industry and this research helps highlight potential approaches to this end.

REFERENCES

- [1] J. K. Olesen, G. N. Yannakakis, and J. Hallam, "Real-time challenge balance in an rts game using rtneat," in *2008 IEEE Symposium On Computational Intelligence and Games*, 2008, pp. 87–94.
- [2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds., vol. 25. Curran Associates, Inc., 2012. [Online]. Available: <https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf>
- [3] Q. Guo, Z. Yu, Y. Wu, D. Liang, H. Qin, and J. Yan, "Dynamic recursive neural network," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 5142–5151.
- [4] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 11 1997. [Online]. Available: <https://doi.org/10.1162/neco.1997.9.8.1735>
- [5] K. O. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," *Evol. Comput.*, vol. 10, no. 2, p. 99–127, Jun. 2002. [Online]. Available: <https://doi.org/10.1162/106365602320169811>
- [6] E. J. Hastings, R. K. Guha, and K. O. Stanley, "Automatic content generation in the galactic arms race video game," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 1, no. 4, pp. 245–263, 2009.
- [7] X. Li, S. He, Y. Dong, Q. Liu, X. Liu, Y. Fu, Z. Shi, and W. Huang, "To create dda by the approach of ann from uct-created data," in *2010 International Conference on Computer Application and System Modeling (ICCA SM 2010)*, vol. 8, 2010, pp. V8–475–V8–478.
- [8] A. Ebrahimi and M.-R. Akbarzadeh-T, "Dynamic difficulty adjustment in games by using an interactive self-organizing architecture," in *2014 Iranian Conference on Intelligent Systems (ICIS)*, 2014, pp. 1–6.
- [9] S.-H. Cho and S.-J. Kang, "An automated wave generation technique in tower defense games based on a genetic algorithm," *Journal of Korea Game Society, Korea Academic Society of Games*, April 2011.
- [10] S. Risi and J. Togelius, "Neuroevolution in games: State of the art and open challenges," *CoRR*, vol. abs/1410.7326, 2014. [Online]. Available: <http://arxiv.org/abs/1410.7326>
- [11] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, Dec 1943. [Online]. Available: <https://doi.org/10.1007/BF02478259>
- [12] M. Guzdial and M. Riedl, "Automated game design via conceptual expansion," *CoRR*, vol. abs/1809.02232, 2018. [Online]. Available: <http://arxiv.org/abs/1809.02232>
- [13] A. Khalifa, P. Bontrager, S. Earle, and J. Togelius, "PCGRL: procedural content generation via reinforcement learning," *CoRR*, vol. abs/2001.09212, 2020. [Online]. Available: <https://arxiv.org/abs/2001.09212>
- [14] H. Kim, S. Lee, H. Lee, T. Hahn, and S. Kang, "Automatic generation of game content using a graph-based wave function collapse algorithm," in *2019 IEEE Conference on Games (CoG)*, 2019, pp. 1–4.
- [15] L. A. Beck, "Csikszentmihalyi, mihaly. (1990). flow: The psychology of optimal experience," *Journal of Leisure Research*, vol. 24, no. 1, pp. 93–94, 1992. [Online]. Available: <https://doi.org/10.1080/00222216.1992.11969876>
- [16] M. Zohaib, "Dynamic difficulty adjustment (dda) in computer games: A review," *Advances in Human-Computer Interaction*, vol. 2018, p. 5681652, Nov 2018. [Online]. Available: <https://doi.org/10.1155/2018/5681652>
- [17] R. Sutoyo, D. Winata, K. Oliviani, and D. M. Supriyadi, "Dynamic difficulty adjustment in tower defence," *Procedia Computer Science*, vol. 59, pp. 435–444, 2015, international Conference on Computer Science and Computational Intelligence (ICCCSCI 2015). [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S187705091502092X>
- [18] S. Winkler, "On the properties of subjective ratings in video quality experiments," in *2009 International Workshop on Quality of Multimedia Experience*, 2009, pp. 139–144.
- [19] W. IJsselstein, Y. de Kort, and K. Poels, *The Game Experience Questionnaire*. Technische Universiteit Eindhoven, 2013.