РАЗРАБОТКА СТРАТЕГИИ ИСКУССТВЕННОГО ИГРОКА В ИГРЕ FLAPPY BIRD С ИСПОЛЬЗОВАНИЕМ НЕЙРОЭВОЛЮЦИИ РАСШИРЕНИЯ ТОПОЛОГИИ (NEAT)

Шевкет Умут Чакир[1)], Васиф Набиев[2)]

1) Университет Памуккале, г. Денизли, Турция, sucakir@pau.edu.tr
2) Черноморский Технический Университет, г. Трабзон, Турция, vasif@ktu.edu.tr

Аннотация: Видеоигры представляют собой упрощенные модели сложных реальных сред и реализованы в виде совокупности упрощенных задач, имитирующих реальную проблему. Видеоигры также можно рассматривать как хорошие тестовые стенды для алгоритмов искусственного интеллекта. Решение задач в видеоиграх может привести к решению реальных задач с использованием проверенных алгоритмов искусственного интеллекта. В этой работе мы разработали искусственный игровой агент с использованием нейроэволюции расширения топологии игры Flappy Bird. Игровые состояния представлены с использованием восьми характеристик, включая расстояние птицы от земли и до труб, скорость птицы в верхнем и нижнем уровнях трубы. Эксперименты показывают, что наши агенты могут изучать игровую среду без предварительного знания игрового домена.

Ключевые слова: игровой искусственный интеллект, нейроэволюция, летящая птица, видеоигры.

CREATING AN ARTIFICIAL FLAPPY BIRD GAME PLAYER USING NEURO EVOLUTION OF AUGMENTING TOPOLOGIES (NEAT)

Şevket Umut Çakır[1)], Vasif Nabiyev[2)]

1) Pamukkale University, Denizli, Turkey, sucakir@pau.edu.tr
2) Karadeniz Technical University, Trabzon, Turkey, vasif@ktu.edu.tr

Abstract: Video games are simplified representations of complex real-world environments and consist of several simplified tasks of real-world problems. Video games are also good testbeds for artificial intelligence algorithms. Solving the tasks in video games might lead to solve the real-world problems using tested artificial intelligence algorithms. In this work we developed an artificial game playing agent using Neuro Evolution of

Augmenting Topologies for Flappy Bird game. Game states are represented using eight features including distance of the bird from ground and to pipes, velocity of the bird and top and bottom locations of the pipes. Experiments show that our agents are able to learn the game environment without prior knowledge of the game domain.

Key words: game artificial intelligence, neuroevolution, flappy bird, video games.

1. Introduction

Video games are simplified versions of complex real-world environments for entertainment. Problems in video games might be difficult to solve for computers, and even for humans. Video games might help researchers to develop new artificial intelligence algorithms or test of existing ones better in artificial environments.

Video games are different from classic board games like chess, checkers and go. Classical board games are turn-based games which player has plenty amount of time to decide the action. In a video game running at 25 frames per second player has 40 ms time to decide the action in a certain state. Though an artificial player also has limited time to act. The complexity of video games is also big. Search space of a typical video game is many orders of magnitude larger than a classical board game[1]. Video games can also be used as testbeds for artificial intelligence algorithms.

Both artificial intelligence and computational intelligence methods are widely used in video games. Computational intelligence refers to a set of nature-inspired algorithms methodologies and approaches to solve complex problems. For convenience, we are going to use the term artificial intelligence throughout the paper.

In section 2 there is a short review of artificial game players in literature and Flappy Bird related research. Section 3 describes the game environment used and learning algorithm. Experimental setup and training results are explained in section 4 and finally, section 5 is the conclusion of the paper.

2. Related Work:

Artificial intelligence is widely used in board and video games. In diverse selection of games artificial players can beat human players. IBM’s artificial chess player Deep Blue has won against world chess champion Garry Kasparov with the score of 3½ - 2½ in 1997. Recently AlphaGo, an artificial player for Go game, has won 5-0 against European Go champion Fan Hui in 2015 and 4-1 against Lee Sedol, winner of the 18 world title and widely considered to be the greatest Go player of the past decade, in 2016 [2]. Yannakakis and Togelius have

made a review of how many subcategories exists about how artificial intelligence methods are used in video games [3]. There are ten different categories including non-player character behavior learning, player modeling, procedural content generation, general game artificial intelligence etc. Artificial game players are computer agents that try to solve a video game like humans do. Evolutionary computation, reinforcement learning, planning, tree search methods, artificial neural networks and deep learning have been widely applied to create artificial game players so far [3]–[5].

Flappy Bird game has been used for player modeling [6]–[8] and optimization problems [9] so far which is different from our approach, creating an artificial player.

3. Game Environment and NEAT:

Flappy Bird, originally created by Dong Nguyen and published at Apple App Store, is a side-scrolling game which player tries to advance by controlling a bird without hitting pipes and with applied gravity to the bird[10]. The game had become very popular amongst mobile players in spite of its irritating gameplay experience. After developer of Flappy Bird took down the game from mobile stores, many replicas have been developed for various platforms. One of these platforms is PyGame Learning Environment which is created for reinforcement learning tasks[11].

OpenAI Gym is an open source library for developing and comparing reinforcement learning algorithms[12]. It consists of many different learning environments for various reinforcement learning tasks including Atari games, classical control problems, algorithmic tasks etc. We used PyGame Learning environment ported to OpenAI Gym.

Neuroevolution of Augmenting Topologies(NEAT) is a machine learning approach which tries to determine the weights and topology of a neural network using genetic algorithms[13]. NEAT has been developed by Stanley and Miikkulainen to overcome direct encoding problems like permutation and noisy fitness. NEAT is both suitable for supervised, unsupervised and reinforcement learning tasks like other neuroevolution algorithms[4]. In NEAT each individual represents an artificial neural network. A sample neural network represented by an individual can be seen in Figure 1. Cells in upper part represent connection genes, which forms a connection between two nodes. Connections might be enabled or disabled. NEAT keeps track of the structural changes with a number called innovation. Whenever a new connection is formed, they get a new incremented innovation number. For example all connection genes between nodes 3 and 5 will have same innovation number.

In:1 Out:3 Weight: 0.6 Enabled: Yes Innovation: 1	In:1 Out:4 Weight: -0.8 Enabled: Yes Innovation: 2	In: 2 Out: 3 Weight: 0.1 Enabled: No Innovation: 3	In: 2 Out: 4 Weight: 0.4 Enabled: Yes Innovation: 4	In: 3 Out: 5 Weight: 0.9 Enabled: Yes Innovation: 5	In: 4 Out: 5 Weight: -0.5 Enabled: Yes Innovation: 6

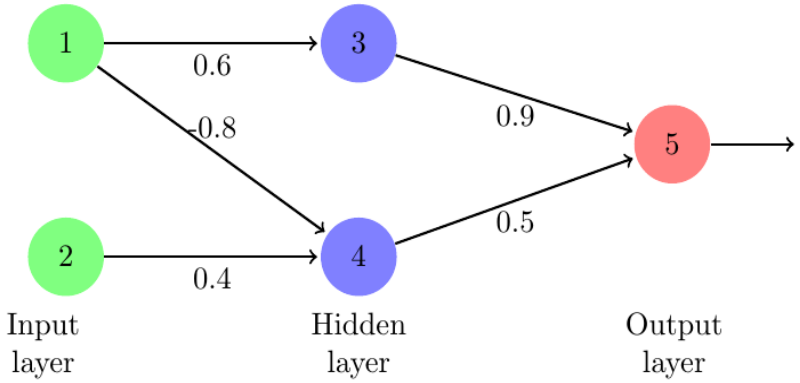

Figure 1: Sample representation of a typical NEAT genome and corresponding neural network

Stanley et al. have applied a real-time version of NEAT to a game called NeuroEvolving Robotic Operatives(NERO) to create artificial players[14]. Since then NEAT has been applied to various games and even used to create new type of games[4]. We used an open source library called neat-python to train agents for the game[15].

4. Experiments:

The population consists of individuals and each individual in a population represents an artificial neural network. Each individual is evaluated distinctly by playing a game session. In a certain game state observations are neural network inputs and actions are outputs. Seven computed distance shown in Figure 2 and velocity of the bird is used for neural network inputs. There is only one output of neural networks to determine if the bird will flap or not.

We started with a topology of all input nodes connected to the output node and no hidden nodes. Default activation function for neural networks was rectified linear units (RELU) and new nodes created by mutation used RELU and clamped function described in equation 1.

$$clamped(x) = \begin{cases} x < -1 & -1 \\ -1 \le x \le 1 & x \\ x > 1 & 1 \end{cases} \quad (1)$$

Initial population, as in all evolutionary algorithms, of 300 individuals created at the beginning. Every individual in the population is evaluated by how

many pipes the bird has survived. After doing crossover and mutation operations, new individuals are evaluated and survival selection is done.

We trained our agents for 250 generations. Best and average fitness of generations can be seen in Figure 3. Best individual of the training is able to pass 142 pipes without hitting them or falling on the ground. Training is done on an Intel Xeon E5-2620 CPU with 24 cores. Using parallel processing evaluator of neat-python, training process took around 4 hours with 24 worker processes. To obtain reproducibility we used same ephemeral random constant seed throughout all training to create the game environment.

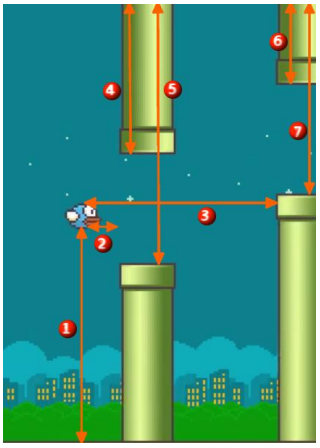

Figure 2: Neural network inputs

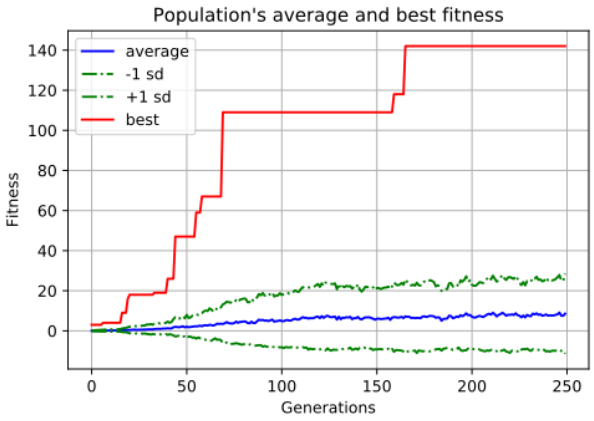

Figure 3: Training fitness's

Agents might get stuck at a certain score during training. Between 65th and 160th generations agents got stuck around the score of 110 which can be seen in Figure 3. After 160th generation, as better skilled agents evolve, 110 score barrier has been passed over but got stuck at the score of 142.

NEAT uses speciation to treat individuals as species. During training stagnated species are removed from population and new individuals join. If new individuals are in a certain distance with one of the existing species, then they join those species. Otherwise, they form new species. Figure 4 shows the size of the species. At the beginning there are only 3 species and 300 individuals are in one of these 3 species. During training process some of the species become extinct due to stagnation and new species are formed. For example one of the starting species, colored blue, becomes extinct around generation 90.

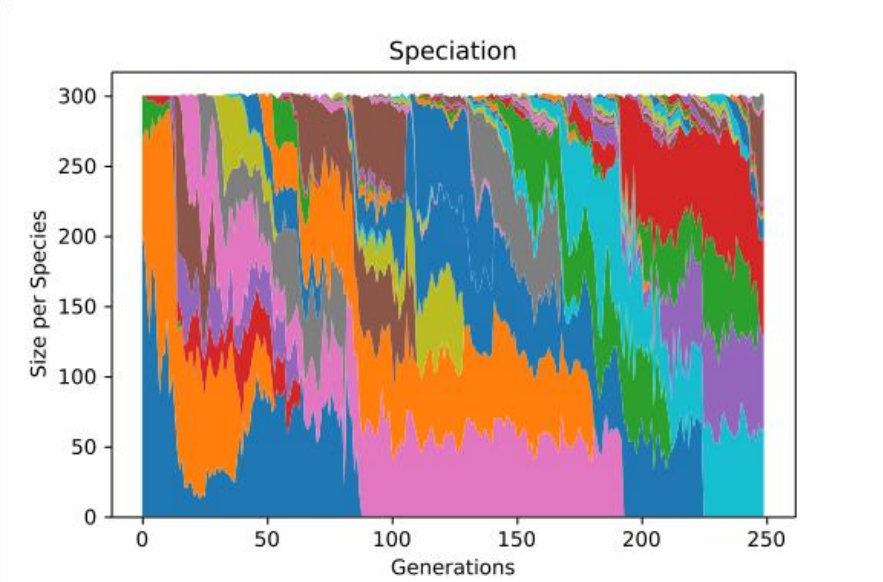

Figure 4: Speciation during training

5. Conclusion:

We tested the NEAT algorithm on Flappy Bird game. Agents trained by NEAT are able to learn the Flappy Bird environment without prior knowledge of the problem domain. Agents might get stuck at some point due to unfair placement of pipes. With enough training, it has been observed that new agents in the population are able to pass over the score barrier. Average fitness of the population is another property which is increasing during training.

Acknowledgements:

We would like to thank OpenAI Company, creator of OpenAI Gym platform, and the creators of PyGame Learning Environment, neat-python and Flappy Bird who made this research possible.

References:

[1] S. Ontañón, G. Synnaeve, A. Uriarte, F. Richoux, D. Churchill, and M. Preuss, "A Survey of Real-Time Strategy Game AI Research and Competition in StarCraft," *IEEE Trans. Comput. Intell. AI GAMES*, vol. 5, no. 4, pp. 293–311, 2013.

[2] V. Nabiyev, *Artificial Intelligence*, Seçkin Yayıncılık, Ankara, 2016.

[3] G. N. Yannakakis and J. Togelius, "A Panorama of Artificial and Computational Intelligence in Games," *IEEE Trans. Comput. Intell. AI Games*, vol. 7, no. c, pp. 1–1, Dec. 2014.

[4] S. Risi and J. Togelius, "Neuroevolution in Games: State of the Art and Open Challenges," *CoRR*, vol. abs/1410.7, no. c, pp. 1–19, 2014.

[5] N. Justesen, P. Bontrager, J. Togelius, and S. Risi, “Deep Learning for Video Game Playing,” Aug. 2017.

[6] A. Isaksen and A. Nealen, “Comparing Player Skill, Game Variants, and Learning Rates Using Survival Analysis,” *Elev. Artif. Intell. Interact. Digit. Entertain. Conf.*, Sep. 2015.

[7] A. Isaksen, D. Gopstein, J. Togelius, and A. Nealen, “Discovering Unique Game Variants.”

[8] A. Isaksen, D. Gopstein, and A. Nealen, “Exploring Game Space Using Survival Analysis.”

[9] C. D. Schuman, A. Disney, S. P. Singh, G. Bruer, J. P. Mitchell, A. Klibisz, and J. S. Plank, “Parallel Evolutionary Optimization for Neuromorphic Network Training,” in *2016 2nd Workshop on Machine Learning in HPC Environments (MLHPC)*, 2016, pp. 36–46.

[10] D. Nguyen, “Flappy bird,” *Apple App Store*, 2013.

[11] N. Tasfi, “PyGame Learning Environment,” *GitHub repository*. GitHub, 2016.

[12] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, “OpenAI Gym,” *CoRR*, vol. abs/1606.0, 2016.

[13] K. O. Stanley and R. Miikkulainen, “Evolving Neural Networks through Augmenting Topologies,” *Evol. Comput.*, vol. 10, no. 2, pp. 99–127, 2003.

[14] K. O. Stanley, B. D. Bryant, and R. Miikkulainen, “Real-time neuroevolution in the NERO video game,” *IEEE Trans. Evol. Comput.*, vol. 9, no. 6, pp. 653–668, 2005.

[15] CodeReclaimers, “neat-python,” *GitHub repository*. GitHub, 2015.