Deep Convolutional Player Modeling on Log and Level Data

Nicholas Liao
Georgia Institute of Technology
nliao7@gatech.edu

Matthew Guzdial
Georgia Institute of Technology
mzguzdial@gatech.edu

Mark Riedl
Georgia Institute of Technology
riedl@cc.gatech.edu

ABSTRACT

We present a novel approach to player modeling based on a convolutional neural net trained on game event logs. We test our approach and a hybrid extension over two distinct games, a clone of *Super Mario Bros.* and *Gwario*, a human computation version of *Super Mario Bros.: The Lost Levels*. We demonstrate high accuracy in predicting a variety of measures of player experience across these two games. Further we present evidence that our technique derives quality design knowledge and demonstrate the ability to build a more general model.

CCS CONCEPTS

•**Human-centered computing** → *HCI theory, concepts and models;* •**Applied computing** → *Computer Games;*

KEYWORDS

player modeling, deep neural nets, convolutional neural nets, super mario bros.

ACM Reference format:
Nicholas Liao, Matthew Guzdial, and Mark Riedl. 2017. Deep Convolutional Player Modeling on Log and Level Data. In *Proceedings of Foundations of Digital Games Conference, Cape Cod, Massachusetts USA, August 2017 (FDG'17),* 4 pages.
DOI: 10.475/123_4

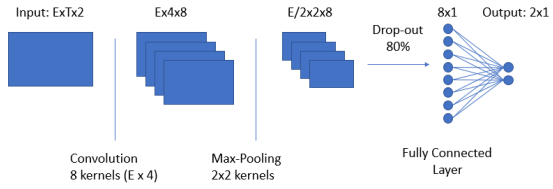
1 INTRODUCTION

Player modeling is the field associated with the problem of learning to predict player experience. A common machine learning approach involves a designer picking out a set of super-features to summarize the player's performance (e.g. total enemies killed, total number of deaths, etc), writing code to pull these values from game logs (timestamped records of button presses and in-game events occurrences. from a particular playthrough), and mapping these features to player experience measures (e.g. fun, challenge, etc). This mapping is then used to predict novel player experiences. Despite successful experimental applications, player modeling goes unused in most modern games, with game companies preferring to model players in aggregate with player analytics [2, 3]. One reason that designers choose not to pursue player modeling might be the difficulty in designing appropriate super-features to summarize player experience. In addition, with the recent diversification of the video game industry, techniques must account for larger variability in a player preferences[9].

FDG'17, Cape Cod, Massachusetts USA

DOI: 10.475/123_4

Figure 1: A visualization of our presented convolutional neural network architecture. A single CNN layer empowers this model that scans 3-step sequences of events from each pair of game logs and makes predictions on which presented game log would be ranked higher according to self reports.



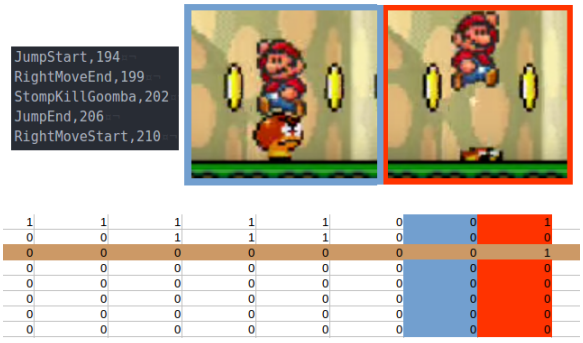
In this paper we present techniques to automatically rank player experiences from game event logs and level structure information based on self-reported rankings. We examine the applications of a convolutional neural net (CNN), for its ability to learn a "set of features" to track automatically, cutting back on designer authoring burden. We evaluate this technique in two games, a *Super Mario Bros.* clone and a related platformer. We demonstrate that this technique, along with a complementary prior technique [8], can accurately predict player experience. The primary contribution presented in this paper is an approach to model players automatically from game event logs based on pairwise rankings.

2 RELATED WORK

Yannakakis et al. [16] describe player modeling as "the study of computational means for the modeling of player cognitive, behavioral, and affective states which are based on data (or theories)". Most commonly player modeling is applied to the problem of player customization, adjusting elements such as difficulty to tailor an individual user experience [4][3][13][1]. We identify a set of prior player modeling work relevant to this paper.

Drachen et al. [5] created a player modeling system in the game *Tomb Raider: Underworld* trained on a set of hand-defined variables (times help command used, level completion time, and number/cause of death) extracted from game event logs, which are files that represent the sequence of actions taken and events that occurred during play. They make use of self organizing maps [14], a type of artificial neural network as the basis of their model. Shaker et al. [11] present a general player modeling system applied to *Super Mario Bros.* and a first person shooter game called *Sauerbraten*. They hand define a wide set of features summarizing game log events (e.g. total enemies killed), but make use of an unsupervised approach to pick from this set. They make use of a model based on an neural network architecture constructed via an evolutionary process, and predict player experience based on self reports.

Figure 2: Example of the game log matrix and their corresponding ticks. The third row records when the player stomps on a Goomba.



There are prior uses of CNNs to games outside of player modeling. Guzdial et al. [8] proposes a technique for level modeling using a computer agent that plays through the levels. They have shown that CNNs can independently determine what parts of a level indicate player enjoyment, effectively selecting its own features. We utilize the Guzdial et al. system in a "hybrid" approach in conjunction with our novel log-based method. Mnih et al. [10] have further shown that CNNs can capture player strategies and behaviors.

3 SYSTEM OVERVIEW

3.1 Log Network

We describe our "log" network, the convolutional neural net (CNN) approach based entirely on game logs as input. Given that we make use of a neural network architecture all game logs must be of the same shape. We choose to format all information as a matrix.

We present an illustration of our game log matrix in Figure 2. The rows of the matrix correspond to different events in the logs, and columns correspond to the number of time steps (called "ticks" in games) needed to complete the level. Since CNNs require a fixed input dimension and completion time varies among players, we normalize the total ticks to fit within the predefined size. We then train our CNN architecture by passing in two game logs (representing two different levels a single player played) and set as it's target the ranking the player reported across a particular feature (e.g. "level 1 was more fun than level 2"). In this fashion we train a unique CNN for each feature present in the self-reported rankings.

After transforming each player's game logs into a matrix we have 2-dimensional matrices with a consistent E value (given that the same events occur across all levels), but differing values of T (as players take varying amounts of time to finish the level). A deep neural network requires that all input be of same size. We found the most success with normalizing all matrices to the same T value of 1000 time steps (a value slightly lower than the lowest actual completion time in the dataset). We anticipate that this was successful as most events occurred across several frames, and therefore this allowed the CNN to capture more interactions between ticks.

We visualize our CNN architecture in Figure 1. The core component of the CNN, the convolution layer, scans subregions of the input to find patterns. A filter is a fix-sized "picture frame", that moves across the input, creating subregions. These techniques were originally used to analyze images. We find CNNs to be appropriate for the task of learning a model of player experience from events due to their relational awareness in other contexts. Since the CNN can capture relational information, we make use of a filter that captures four columns at a time, representing a sequence of four actions. In a level, we would expect blocks close to each other in space to be represent a structure. In a player log, a sequence of logs may represent some high level action like stomping on an enemy.

Our first technique takes only log matrices as its input, feeds it through the CNN architecture as discussed above, and predicts solely on the events that occur to the player. Two players may play through the same levels and rank them opposite of one another. While the level technique would feed the same information in and have contradicting datapoints, the log matrix would have unique inputs, specific to each player's playthrough.

3.2 Hybrid Network

Our second technique is a hybrid of our system and the Guzdial et al. system, with both neural network architectures combining into a final fully connected layer. This "hybrid" therefore represents a combination of log and level information, with the ability to make decisions based on both individual systems. We see this as an extension of previous techniques by integrating our log information. For more detail about the log system, see [8].

For levels, we make use of a similar matrix representation. Each level can be broken up into its underlying grid system, where a grid space can only be occupied by one object at a time. Each unique foreground object, whether it be blocks, collectibles, or enemies, is mapped to a unique identifying number. The location in the matrix corresponds to its position in the level.

The log and level matrices go through their own convolution, max-pooling, and dropout layers. They are then connected together by a fully-connected layer to be used for prediction.

4 EVALUATION OVERVIEW

We ran a total of three evaluations of our system. We applied our system on two games, a *Super Mario Bros.* clone called Infinite Mario [15] and a Mario-derivative focused on performing human computation *Gwario* [12]. We note that in *Gwario*, the player must collect specific sets of items as opposed to only finding the end of the level (i.e. the player's objectives are different). We make use of two separate games in our evaluation in order to demonstrate the generalizability of our model, and focus on Mario-like games due to the popularity of Mario as a baseline. We ran a final third evaluation to further address the question of generalizability.

We include results from a baseline. Guzdial et al. [8] made use of a CNN-based approach to predict an aggregate player score of a Mario level based on the level architecture, and a small set of hand-defined variables (e.g. number of deaths to enemies, number of deaths to gaps, number of enemies killed, and time to complete a level divided by its width). We include it as our baseline as it is a component part of our "hybrid" approach, if it beats out that extension of our system that would demonstrate a failure of our log-based CNN approach for player modeling.

Table 1: Mean and median accuracies across 10 folds. Mario

	Challenge		Creativity		Design		Frustration		Fun	
	mean	median	mean	median	mean	median	mean	median	mean	median
Level	74.44%	74.44%	53.33%	54.44%	77.56%	76.67%	76.22%	77.78%	63.33%	62.22%
Log	65.78%	64.44%	64.67%	64.44%	75.11%	73.33%	69.33%	68.89%	64.89%	65.56%
Hybrid	**83.11%**	**82.22%**	**71.33%**	**70.00%**	**81.11%**	**80.00%**	**81.55%**	**83.33%**	**81.56%**	**81.11%**

5 SUPER MARIO BROS. EVALUATION

For our first evaluation we applied our system to a clone of *Super Mario Bros.* called "Infinite Mario". We drew on a dataset from a study previously conducted in the game engine, which we describe briefly below but for more detail see [6]. Ultimately we ran a ten cross-fold analysis on the engine between the three experimental systems. In the following subsections we discuss the evaluation setup (including a description of the game), discuss the results of our ten cross-fold analysis, and give examples of the learned features of our CNN.

We adapted the dataset used by Guzdial et al. in [7]. Seventy-five players were asked to play Level 1-1 from the original Super Mario Bros, and then two other levels from a pool of 15 artificially generated levels. After, players were asked to rank the three levels based on fun, frustration, challenge, level design, and creativity of the levels. For each player, we took permutations of two level logs, and labeled the pair with a classification of "Level 1 was more X" or "Level 2 was more X" where X was fun, frustrating, challenging, well designed, or creative. This resulted in 6 data points per person, or 450 data points. By adding the reverse of the reported ranking into the dataset, we have guaranteed that half the dataset is of class 1, and the other half class 2. Therefore, we would anticipate a pure random system to perform at around 50% accuracy.

5.1 SMB Results

We report results over the five categories in tables 1. We use "Log" to indicate our system, "Level" to indicate the prior system largely reliant on level structure [8], and "Hybrid" to indicate the combined architecture of the two prior systems. We use the Wilcox test to evaluate statistical significance in output between pairs of systems.

In all cases but design, the hybrid system performs significantly better than either of its two constituent systems. This suggests that the two constituent systems (log and level) offer complimentary information towards making predictions of player experience, rather than one being strictly better than the other. In addition, these results demonstrate that some types of information are more predictive to certain measures of player experience. For example, the creativity labels were predicted more accurately given access to logs of events, suggesting players reflected on the comparative experiences when deciding on this ranking.

5.2 SMB Learned Actions

One of the strengths of CNNs is their ability to learn useful features from training data. To evaluate the CNN's ability to extract useful super-features from raw game logs, we visualize the pairs of comparative event sequences that maximally activate the learned filters our "log" CNN trained to predict. We visualize four of the eight trained filters for our CNN trained to predict challenge (Figure 3).

Figure 3: Maximally activated visualizations of four of the eight filters for challenge.

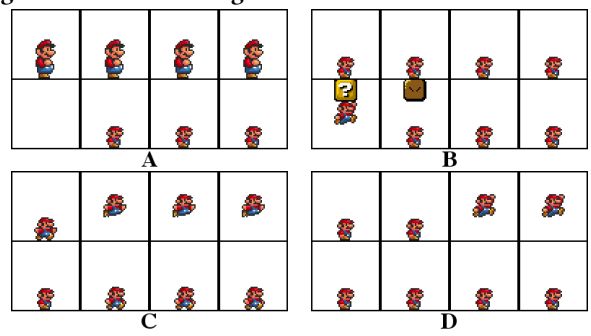


Figure 3 presents visualizations of four of the eight trained filters for the challenge labels. Figure 3(a) demonstrates two sequence pairs where one game log includes the player restarting the level where the other game log section has the same player standing still as "large mario". The other visualized filters largely involve comparisons of progress. For example, Figure 3(d) compares a sequence where the player jumps forwards versus a sequence where the player is standing still.

6 GWARIO EVALUATION

For our second evaluation we applied out system to *Gwario*, a game with a purpose (or GWAP), adaption of the Japanese sequel to *Super Mario Bros.*, *Super Mario Bros.: The Lost Levels*. GWAPs are games that outsource work in the form of a game. In this instance, the player may take the same actions, but in addition to finding the end of a level, the player attempts to collect items that answer a human computation question.

We draw on the dataset from the study conducted in Siu et al in [12]. Players were asked to play two levels from a pool of four adapted from *Super Mario Bros.: The Lost Levels*, and rank them based on challenge, fun, and frustration. We note that 58 players took part in the study, resulting in 116 data points. As in the previous evaluation we split the dataset into 10 cross-folds.

6.1 Gwario Results

We report results over the three categories in tables 2. We use the same notation as our SMB Results.

We find that these results suggest the player logs were much more predictive than the level information for the professionally designed *Gwario* levels, perhaps due to greater variation and size.

Table 2: Mean and median accuracy across 10 folds for the Gwario dataset.

	Challenge		Frustration		Fun	
	mean	mdn	mean	mdn	mean	mdn
Level	52.7%	59.1%	43.6%	45.5%	49.1%	50.0%
Log	75.5%	77.3%	78.2%	77.3%	66.4%	68.2%
Hybrid	**78.2%**	**81.8%**	**80.9%**	**81.8%**	**76.4%**	**77.3%**

Figure 4: Maximally activated visualizations of four of the eight filters for challenge.

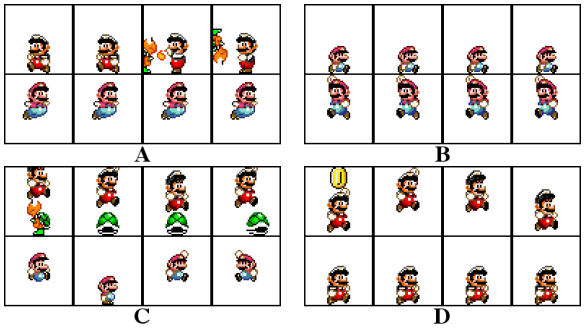


Table 3: Mean and median accuracies across 10 folds with a dataset made of equal halves of the two datasets.

	Challenge		Frustration		Fun	
	mean	mdn	mean	mdn	mean	mdn
Level	63.6%	63.6%	61.4%	59.1%	55.0%	54.6%
Log	62.3%	59.1%	63.6%	65.9%	**61.8%**	**59.1%**
Hybrid	**71.4%**	**70.5%**	**71.8%**	**72.7%**	58.6%	56.8%

Given the success of the hybrid approach, this indicates that both the log and level systems contributed to the predictive power, suggesting they again represented complementary approaches.

6.2 Gwario Learned Actions

In this section, we visualize the patches of comparative event sequences that maximally activate the learned filters our "log" CNN trained to predict. We visualize four of the eight trained filters for our CNN trained to predict challenge (Figure 4).

Figure 4 represents the four sequence pairs that maximally activated four of the eight trained filters of our CNN. Overall the sequence pairs match our intuition, with two focusing on defeating enemies as opposed to making progress (Figures 4a and 4c), and one focused on making progress versus not (Figure 4b). We note in particular Figure 4d, which highlights the player collecting a coin versus making forward progress. This is important for the system to learn given the added role coins play in *Gwario*.

7 GENERALIZATION EVALUATION

We have thus far demonstrated that our systems can perform reasonably accurately across two distinct, but similar games. However, each of these games required completely retraining our system. Ideally we would create a more general game player modeling system, at least across games of the same genre, in order to cut back on training time and present player experience predictions for games without training data. We present the results of an evaluation to address the issue of generalizability. We note that the only major alteration of our system to this evaluation was the inclusion of the full set of events from both games for our game matrix.

For our generalizability evaluation we composed a dataset made of half *Gwario* and half *Super Mario Bros.* data, selected randomly from each dataset. We then split this new "mixed" dataset into ten folds, with each fold containing data half from each game. We then ran a similar evaluation as above, and report the average and median accuracies in Table 3. We note that while there is a drop of about 10% from the average accuracies we saw when each system was trained on each individual game, the accuracies are above the random baseline of 50%. This suggests that it is possible to apply these techniques more generally across different, if similar, games. If we could determine a universal set of event log types, it may be result in higher accuracies.

REFERENCES

[1] Mohamed Abou-Zleikha and Noor Shaker. 2015. Evolving random forest for preference learning. In *European Conference on the Applications of Evolutionary Computation*. Springer, 318–330.
[2] Mike Ambinder. 2011. Biofeedback in gameplay: How valve measures physiology to enhance gaming experience. In *game developers conference*, Vol. 2011.
[3] Sander Bakkes, Shimon Whiteson, Guangliang Li, George Viorel Vişniuc, Efstathios Charitos, Norbert Heijne, and Arjen Swellengrebel. 2014. Challenge balancing for personalised game spaces. In *Games Media Entertainment (GEM), 2014 IEEE*. IEEE, 1–8.
[4] Glen Berseth, M Brandon Haworth, Mubbasir Kapadia, and Petros Faloutsos. 2014. Characterizing and optimizing game level difficulty. In *Proceedings of the Seventh International Conference on Motion in Games*. ACM, 153–160.
[5] Anders Drachen, Alessandro Canossa, and Georgios N Yannakakis. 2009. Player modeling using self-organization in Tomb Raider: Underworld. In *Computational Intelligence and Games, 2009. CIG 2009. IEEE Symposium on*. IEEE, 1–8.
[6] Matthew Guzdial and Mark Riedl. 2016. Game Level Generation from Gameplay Videos. In *Twelfth Artificial Intelligence and Interactive Digital Entertainment Conference*.
[7] Matthew Guzdial and Mark Riedl. 2016. Learning to Blend Computer Game Levels. In *Seventh International Conference on Computational Creativity*.
[8] Matthew Guzdial, Nathan Sturtevant, and Boyang Li. 2016. Deep static and dynamic level analysis: A study on Infinite Mario. In *Experimental AI in Games Workshop*, Vol. 3.
[9] Jesper Juul. 2010. *A casual revolution: Reinventing video games and their players*. MIT press.
[10] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, and others. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529–533.
[11] Noor Shaker, Mohammad Shaker, and Mohamed Abou-Zleikha. 2015. Towards generic models of player experience. In *Proceedings, the Eleventh Aaai Conference on Artificial Intelligence and Interactive Digital Entertainment (aiide-15)*. AAAI Press.
[12] Kristin Siu, Matthew Guzdial, and Mark Riedl. 2017. Procedural Content Generation via Machine Learning (PCGML). *arXiv preprint arXiv:1703.00818* (2017).
[13] Adam Summerville, Matthew Guzdial, Michael Mateas, and Mark O Riedl. 2016. Learning player tailored content from observation: Platformer level generation from video traces using LSTMs. In *Twelfth Artificial Intelligence and Interactive Digital Entertainment Conference*.
[14] Christian Thurau, Christian Bauckhage, and Gerhard Sagerer. 2003. Combining Self Organizing Maps and Multilayer Perceptrons to Learn Bot-Behaviour for a Commercial Game.. In *GAME-ON*. Citeseer, 119.
[15] Julian Togelius, Sergey Karakovskiy, and Robin Baumgarten. 2010. The 2009 mario ai competition. In *IEEE Congress on Evolutionary Computation*. IEEE, 1–8.
[16] Georgios N Yannakakis, Pieter Spronck, Daniele Loiacono, and Elisabeth André. 2013. Player modeling. In *Dagstuhl Follow-Ups*, Vol. 6. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.