Exploring the Similarity between Game Events for Game Level Analysis and Generation

Tian Zhu
University of Southern California
GamePipe Laboratory
Los Angeles, California
tianzhu@usc.edu

Bingtang Wang
University of Southern California
GamePipe Laboratory
Los Angeles, California
bingtanw@usc.edu

Michael Zyda
University of Southern California
GamePipe Laboratory
Los Angeles, California
zyda@usc.edu

ABSTRACT

Player-Game interaction can be viewed as a sequence of game events between game systems and players. In each event, a piece of information (for example, a game challenge) is presented to the player, and the player responds to that information by executing proper in-game actions. Since different games have different systems and mechanics, it is difficult to analyze and compare game events across different games. In this paper, we propose an approach to measure the similarity between game events from different games quantitatively. This similarity information can then be used to transfer probabilistic models built for analyzing and generating game levels for one game to another game.

CCS CONCEPTS

• **Human-centered computing → HCI design and evaluation methods**; • **Applied computing → Computer games**;

KEYWORDS

HCI, player experience, quantitative measure, probabilistic models, domain transfer

ACM Reference Format:
Tian Zhu, Bingtang Wang, and Michael Zyda. 2018. Exploring the Similarity between Game Events for Game Level Analysis and Generation. In *Foundations of Digital Games 2018 (FDG18), August 7–10, 2018, Malmö, Sweden.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3235765.3235804

1 INTRODUCTION

Since the popularity of video games has increased dramatically in the past few decades, there is an increasing need to understand the intrinsic properties that make video games so intriguing, in order to analyze the "quality" of existing games and improve the designs of future games. Exploring the similarity between different games should shed light on understanding the intrinsic properties of video games.

In the Player Experience of Needs Satisfaction model [14], gamers play video games to satisfy their psychological needs. For most gamers, they do not just play one game. Instead, they play different kinds of games. It suggests that those games that one player loves to play can satisfy the player's psychological needs in a similar fashion.

It implies that at a certain level of abstraction, different games can be analyzed and compared together. Games that provide a similar experience are similar at this level. Once this level of abstraction is found, existing knowledge from one game can be transferred to a different game. For example, a game designer can create a game so that the game levels will be generated based on the player's preference. That way, the game can provide playing experiences similar to the levels from the given player's favorite games

In this paper, we argue that a game level can be abstracted as a sequence of game events. Within each game event, the interaction between the player and the game can be characterized as a probability distribution. The similarity of probability distributions from different events measures how similar the experiences provided by those events are. Probabilistic models (n-gram models) for one game are built to classify enjoyable game levels and to generate new enjoyable levels. Those models are then transferred to a different game by mapping each event in the original game to the most similar event in the new game. Results show that a sequence of game events is a valid abstract representation of a game level, and this abstract representation enables cross-game level classification and generation.

2 RELATED WORKS

Cousin's article in [4] argued that the interaction between players and games can be analyzed as a hierarchy of units. The top-level unit is the interaction of a player playing the whole game. This unit can be subdivided into the interaction of a player playing different game levels. Then each level can be subdivided into a series of game events. The smallest units are called "primary elements" within which interaction between the player and the game cannot be further subdivided. It is difficult to compare primary elements across different games because player actions in primary elements can be entirely different. For example, the primary element in a platformer game requires players to press the jump button to jump over obstacles, while the primary element in a shooter game requires players to move the cursor to aim and shoot a target. In this paper, we focus our research on the level of game events. Though game events are also different in different games, we can characterize the challenge presented in each game event by measuring players' performance on that challenge. This approach enables us to compare game events across different games. One can argue that players' performance on primary elements can also be measured to compare

FDG'18, August 7-10, 2018, Malmö, Sweden

ACM ISBN 978-1-4503-6571-0/18/08…$15.00
https://doi.org/10.1145/3235765.3235804

different primary elements. The issue is that players' performance on a single primary element depends on other primary elements in the same game events. For example, when a player is asked to jump over two consecutive obstacles, the chance that the player can successfully jump over the second platform depends on how difficult it is to jump over the first one. It implies that the dependencies between primary elements in the same game event should be examined, which is beyond the scope of this paper. The types of challenges (difficult, medium, or easy) presented to a player are closely related to the enjoyment a player can obtain from a game level [5]. Thus, analyzing a sequence of game events help us classify the enjoyment a player can obtain from the game level.

Sequential analysis is important in understanding the interactions between players and games [19]. In [19], lag sequential analysis (LSA) is used to analyze sequential behavior patterns of players. In [18], the sequential analysis is conducted on players' behavioral code to analyze the quality of video games objectively. Sequences of game level elements and sequences of player behavior are analyzed to find the connection between in-game content and player experiences in [16], which helps improve the quality of automatically generated game content. In our work, probabilistic models are built to analyze sequences of games events that are characterized by players' performance.

Probability models have been widely used in game research. In [12], Bayesian optimization is used to identify game design characteristics that are important for tuning game difficulty and encouraging player engagement. N-gram [3] is a statistical model based on counting n-element subsequences. It has been used in natural language processing for document classification and generation. In [6], one-tile-wide vertical slices of platformer game levels are treated as building blocks for constructing game levels. N-gram models are trained on those building blocks to generate new game levels that have a similar style to the levels in the training corpus. A sentence in natural language processing consists of a sequence of words, while a game level, in our view, consists of a sequence of game events. Because of the analogy between words and game events, we use n-gram models to classify and generate enjoyable game levels.

Also related to our work is that of Snodgrass and Ontanon [17] who find the mappings between game tiles from different platformer games. Once such mappings are found, probabilistic models (multi-dimensional Markov chains) trained for one game can be transferred to generate game levels for another game. The generated levels for the new game would have similar geometric features to the game levels in the original game. Compared to their work, we want to find the mapping between the higher-level building blocks, i.e., game events, across different games. Thus, the levels generated by our models would not have similar geometric features but would provide game experiences that are similar to the levels in the training corpus.

3 PLAYER-GAME INTERACTION

When a player plays a video game, the output devices continuously present information about the current game state to the player, and the player processes this information to form an understanding of the game states and react accordingly through input devices.

In this sense, games can be viewed as systems of information. Here, the information is the "knowledge" - the informational content of the game state.

The player's emotional needs [15] are satisfied by the process of player gathering information through the output device and executing actions through the input device as a result of decision-making about game states. To better analyze how information is presented to a player and how the player reacts to it, we need to understand when and what information is presented.

In this paper, a game event is defined to include the piece of information presented to the player that contributes to his/her game experiences, and the player's response elicited by this piece of information. Thus, the interaction between player and game can be modeled as a sequence of events.

Information presented in an event will contribute to the overall game experience. For example, a passcode on a piece of paper the player will need later to unlock a treasure chest; the enemy attacking animation that notifies the player that he/she needs to take proper actions now, such as blocking or dodging; or the background music that sets tragic mood for the bad ending of the game.

Information can be divided into smaller chunks. For example, the information about an entire level can be broken down into the information about different encounters. The information about each encounter consists of the information about each enemy in that encounter, and the information about each enemy includes its type, its hit points, and its current movement. The hierarchical structure of information implies the interaction between player and game can be analyzed on different levels of granularity. For example, the characteristics of encounter events will determine the game experience of the whole level, and the information for all enemies in an encounter will determine the game experience for that encounter. In this paper, we focus our discussion on the level of encounters, which are called "game events" in our paper.

3.1 Player Experience

In the Player Experience of Need Satisfaction model (PENS) proposed by Rigby in [15], players are "glued" to video games because they have psychological needs for competence, autonomy, and relatedness that can be satisfied by games. Players' experiences are considered "good" if the psychological needs can be satisfied. In this paper, we focus our discussion on the Competence aspect of PENS model.

Competence is described in [15] as the player's need to conquer challenges and feel a sense of mastery. This concept is closely related to the theory of flow proposed by Csikszentmihalyi in [5].

One of the key elements of achieving flow is the balance between challenge and participant's skill level. If the challenge is too hard for the participant, the participant will feel anxious instead of flow; on the other hand, if the challenge is too easy, the participant will be bored. A proper level of challenge is important for creating flow in participants, as well as satisfying players' needs for competence. In this sense, a game level is considered "enjoyable," if proper challenges are provided to players [1].

When an event provides a challenge to a player, the performance of the player for this event can be evaluated by examining how optimal the player's reactions are. For example, in a first-person

shooter (FPS) game, the performance can be evaluated as the total quantity of resources, such as hit points and ammo, consumed to complete the challenge. If a player is an expert in this game, it is likely that he/she will use relatively fewer resources to defeat enemies; on the other hand, if a player is a novice, he/she will likely get hit more often and shoot less accurately, thus consuming more resources.

The performance for an event can be viewed as the result of a player's cognitive ability (reasoning the proper action to take) and physical ability (execute the action reasoned in a timely manner). So, the outcome can be viewed as an indicator of challenging level for the players.

Since the skill levels of different players are not all the same and mental/physical condition for the same player might not remain the same across different game sessions, the performance of different players across different game sessions can be represented as a random variable. So, the level of challenge of an event can be characterized by the probability distribution of player performance in that event. If most people can complete this event optimally, then we consider this event to be relatively easy. However, if most people's performance in this event is poor, then we consider this event to be difficult. Two events in the same game are considered to have the similar levels of challenge if their performance distributions are similar.

We argue that the similarity of challenging level for events from different games can also be measured using player performance distribution. Games are categorized into different genres, and different game genres have different core mechanics, which produce different types of challenges. In a first-person shooter game, the primary challenge is to shoot enemies [20], which requires players first to find the location of the enemy in the game space and coordinate his/her hand to move the cursor to the enemy position and press the fire button. In a platformer game, the core mechanic is to jump over obstacles, which requires that players find the right timing for jump and press the jump button accordingly. Although the challenges in different game genres require different ability, we can still compare the challenging levels of events from different genres if player performance for those events can be measured on a unified scale. Suppose there is an FPS game where each player only has one hit point and one bullet, similar to western duels. Also, suppose there is a platformer game where each player only has one hit point, and the game will end if the player fails to avoid any obstacle once. In both games, the possible outcome of an event is either a success or a failure. This allows us to use the chances of completing the events as a measure for comparing the similarity between events from different genres.

The event sequence for a game level can be characterized by the performance distribution of each event within that sequence. Two event sequences are considered similar if they have the same number of events and similar performance probability distributions for each pair of events at the same temporal location. We argue that game levels whose event sequences are similar would provide similar game experience.

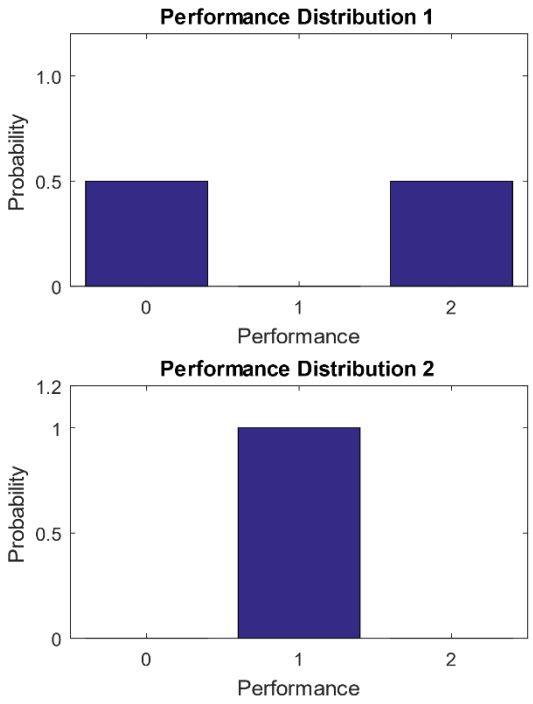

Figure 1: Performance Distributions

4 QUANTIFY SIMILARITY BETWEEN GAME EVENTS

KL-Divergence (Kullback-Leibler divergence) [13] is used to identify the similarity between two performance distributions.

$$D_{KL}(P||Q) = \sum_i P(i) \, log \frac{P(i)}{Q(i)}$$

Here P and Q are the performance distributions we want to compare. P(i) is the probability for outcome i in the performance distribution P.

KL-Divergence is a divergence measure named after Kullback and Leibler. It is a non-negative value, and it is 0 when two performance probability distributions are identical. The more similar two probability distributions are, the closer KL-divergence will approach 0. KL-divergence is not symmetric, so $D_{KL}(P||Q)$ and $D_{KL}(Q||P)$ are not the same. In order to measure the similarity between any pair of events. A modified KL-divergence [13] is used in this study, which is symmetric.

$$D_{KL}(P||Q) + D_{KL}(Q||P)$$

In this paper, player performance is measured as the resources consumed to complete the event. Another possible measure of similarity can be the difference between the mean values (averaged resources consumed to complete the event) of two different performance distributions. In this study, we choose KL-Divergence instead of mean performance difference, because KL-divergence can distinguish between two performance distributions, even if they have similar mean values.

Figure 2: Screenshots for platformer game (left) and FPS game (right) used in experiments.

Consider the performance distributions in Figure 1. The possible value of performance for each event can be 0, 1 or 2. In the first distribution, performance can be 0 or 2 with equal probability. In the second distribution, performance is guaranteed to be 1. The mean performance for both distributions are 1, but the shapes of these performance distributions are entirely different. Suppose the player can only use up to two units of resources (for example, hit points) through the whole level. If an event with the first performance distribution happens at the very beginning of the level, then there is a 50% chance that the player will fail (after consuming all resources). However, if an event with the second distribution occurs at the beginning of the level, it is guaranteed that the player can proceed to the next event. The feeling of progression on those two conditions would vary, thus affecting the overall game experience.

5 TRANSFER EXISTING GAME LEVELS TO ANOTHER GAME

5.1 Games

Two games (Figure 2) of different genres were created for experiments in this paper.

The first game is a classic platformer game. Players need to proceed to end of the level while avoiding obstacles (stationary and moving). The player-controlled character is moved forward by the game system at a constant speed. The only in-game action that players can perform is jumping. In this game, the difficulty of an event can be manipulated by changing the number of obstacles, the height for each obstacle, the distance between obstacles, and the speed of moving obstacles.

The second game is a stationary FPS game. The player-controlled character is placed at a fixed location with a fixed orientation. The player cannot move or rotate the character. Two types of enemies can be spawned in the game. The first type of enemy will self-explode after a certain amount of time. The second type of enemy will continuously fire bullets towards the player. The player can use the mouse to aim the on-screen cursor, then click the mouse button to damage and destroy enemies. The difficulty of an event in the FPS game can be manipulated by changing the number of enemies for each type, the hit points for each enemy, the time before the first type of enemy triggers self-destruction, and the attack frequency and bullet moving speed for the second type of enemy.

For each game, 20 events of various difficulties were pre-designed by authors of this paper. For simple games used in our experiments, we think 20 events are enough to present all core game mechanics at all difficulty levels.

5.2 Performance Distribution

Event performance is measured as the quantity of resource consumed to complete an event.

In the platformer game, the player needs to avoid all obstacles in one event. If a player fails to dodge one obstacle in an event, he/she needs to re-play the current event from the beginning. The resources for each event are the number of attempts available for the player to complete the event.

In the FPS game, resources are hit points that the player has. If a player is hit by an explosion or a bullet, he/she will lose one hit point. For each event, the player needs to defeat all enemies before running out of all hits points.

Performance distributions for 20 pre-designed events in each game are calculated using player logs from fifteen college students (N = 15, fourteen males and one female). All participants are within the age range of 20 to 25, and all reported to have played video games for at least two years. Before any data was collected, a tutorial level in each game was presented to the participants to help them get familiar with game controls and mechanics. Each participant was asked to play all 20 events on a desktop computer inside a quiet room. Event performance was recorded automatically to the hard drive installed on the desktop computer. For each event in both games, three units of resources were provided to the player, thus leaving four possible outcomes for performance distribution, i.e., using zero, one, two, or three units of resources to complete the event. The performance distribution models the probability that a certain number of resources is needed to complete the event.

After all performance distributions were obtained, symmetric KL-divergence was calculated between every platformer event and every FPS event. Each platformer event is mapped to the FPS event with smallest symmetric KL-divergence. We consider the platformer event and its mapped FPS event to be the game events that provide the most similar game experience to players.

5.3 Procedure

Two levels, each of 6 events, were manually generated by the authors for the platformer game using 20 pre-designed events. Level A is composed of relatively easy events. Level B is composed of events with moderate difficulty. The average quantity of resources needed to complete an event in Level A are 0.712 (SD =0.423), while the average quantity of resources needed to complete an event in Level B are 1.326 (SD=0.561).

In order to create FPS levels that provide similar game experience to the two levels we created for the platformer game, events in platformer Level A and Level B were substituted with their most similar FPS events found using KL-divergence. In this way, two transferred levels were created for the FPS game. In FPS Level A, each event needs 0.705 (SD= 0.492) units of resources to complete on average. In FPS Level B, each event needs 1.288 (SD= 0.556) units of resources to complete on average.

Subjects for this experiment were the same 15 college students whose playing logs were used for calculating the performance distributions. Each participant was required to play all four levels

Table 1: Mean enjoyment ratings with standard deviation and results from within genre Friedman's Test

games	Level A	Level B	p-value
Platformer	3.47 ± 0.92	3.80 ± 1.01	0.1317
FPS	3.20 ± 1.21	3.80 ± 1.01	0.0833

(two from each game). The order for which game to play first is randomized, and the order for which level to play within the same game is randomized. A player must finish both levels in one game before playing the other game.

Players were asked to rate how "enjoyable" each level is on a 5-point scale, where one means least enjoyable and five means most enjoyable. Ties are allowed, so two levels from the same game can be assigned the same rating of enjoyment.

5.4 Result

Friedman's Test [8] is conducted to check if all subjects have the same preference for both games. Though the mean enjoyment score for Level B is higher in both games, the difference between Level A and Level B is not statistically significant within each game (Table 1). It implies that different participants might have different preferences for enjoyable levels.

Spearman's correlation coefficient [9] was calculated between the enjoyment rating for platformer game levels and the enjoyment rating for their transferred FPS game levels. The Spearman's correlation is 0.6122 with p=0.015. This result shows that there is a strong correlation between the enjoyment rating for platformer levels and their transferred FPS levels. It implies that if a player prefers a specific platformer level, he/she is likely to prefer its transferred FPS level.

The number of subjects whose preferred level remains consistent across games is counted. 14 out of 15 (93.33%) participants' preferred FPS level is the transferred level of his/her preferred platformer level. This result shows that performance distributions do characterize the types of challenges presented in events from different games, and it can be used to transfer knowledge we have for one game to another.

6 PROBABILISTIC MODELS

In this section, we describe our approach for building the probabilistic models for classifying and generating enjoyable levels for the platformer game. From the last section, the most similar FPS event is found for every platformer event. With this information, models built for the platformer game can be transferred to classify and generate enjoyable levels for the FPS game. The same games and the same 20 pre-designed events from the last section were used in this experiment. A game level is defined as a sequence of game events.

$$l = e_1 e_2 e_3 \dots e_n$$

Here l is a game level of n events. It starts at event e_1 and ends at event e_n.

To classify a game level l as "enjoyable" or "not enjoyable," The Bayesian classifier [11] can be used. Let $P(fun|l)$ and $P(not\ fun|l)$ denote the probability that level l is fun and the probability that level l is not fun, respectively. According to the Bayesian theorem [11], we need to compare the following two probabilities

$$P(fun|l) = \frac{P(l|fun)P(fun)}{P(l)}$$

$$P(not\ fun|l) = \frac{P(l|not\ fun)P(not\ fun)}{P(l)}$$

Probabilities $P(l|fun)$ and $P(l|not\ fun)$ are difficult to calculate directly, but we can approximate them using empirical data. Once we have the approximation for $P(l|fun)$, new levels can be generated directly using this probability distribution.

6.1 N-gram Models

N-gram models [3] have been widely used in natural language processing for document classification. In an n-gram model, a document is represented as a sequence of words, and the probability of a certain word in a document is dependent on previous (n-1) words. We argue that in a game level, the probability of a certain event can also be modeled using n-gram.

6.1.1 Unigram. The unigram model is the simplest form of n-gram (n=1). It assumes that the probability of each event is independent. So, probability $P(l|fun)$ can be approximated as

$$P(l|fun) = \sum_{i=1}^{n} P(e_i|fun)$$

The probability $P(l|not\ fun)$ can be approximated in the same fashion.

If unigram is used, then our probabilistic model is, in fact, the Naive Bayes Classifier. Here, we propose a modified version of unigram, denoted by "unigram-position." In unigram-position model, the probability of each event is dependent on its absolute position in the game levels.

$$P(l|fun) = \sum_{i=1}^{n} P(e_i|fun, i)$$

The intuition behind the unigram-position model is that game levels typically start with rather easy events and become more and more difficult as the player progresses through the level [2], making the absolute position of an event important.

6.1.2 Bigram. Bigram model is another kind of n-gram (n=2) model. It assumes that the probability of a certain event is dependent on the event that proceeds it. In the bigram model, the approximation is

$$P(l|fun) = P(e_1|fun) \sum_{i=2}^{n} P(e_i|fun,\ e_{i-1})$$

Compared to unigram, bigram can model dependence between two consecutive events.

6.2 Event Clustering

Agglomerative Clustering [10] can build a hierarchy of clusters by gradually merging clusters that are similar to each other to form a larger cluster.

Agglomerative Clustering is used in our approximation models for two reasons. First, approximation models need a lot of empirical

data, but collecting the gameplay data is labor-intensive. By clustering similar game events together, we are lowering the number of variables that need to be considered in our approximation models, thus lowering the amount of empirical data that need to be collected. Second, once similar events are clustered, we can generate event patterns that are not present in the collected data. Since events belonging to the same cluster are similar to each other, new patterns can be generated by substituting events in existing patterns with other events in the same cluster.

We used symmetrical KL-divergence as the similarity measure to cluster the 20 pre-designed events for the platformer game. 7 clusters were identified.

6.3 Procedure

For this experiment, 150 random platformer levels and 50 random FPS levels were generated. Each level was generated by randomly selecting five events from 20 pre-designed events in each game.

A college student with two years of game design experience was asked to play all 200 levels and annotate each of them as either "enjoyable" or "not enjoyable." We found in the last section that different players tended to have different preferences for game levels. So, it is difficult to achieve an inter-rater agreement without categorizing players' preferences first. Moreover, the objective of this experiment is to demonstrate that probabilistic models built for one game, for one player can be transferred to a new game for the same player. So, one rater is enough for this experiment.

Among 150 platformer levels, 100 of them were used as the training set to build probabilistic models, and the remaining 50 were used as the testing set. All 50 FPS levels were used as the testing set.

Four probabilistic models were built for the platformer game: unigram, unigram with clustering, unigram-position with clustering, and bigram with clustering. A total of 200 platformer levels were generated from those models, with each model generating roughly 50 levels. The same rater was asked to play those 200 levels and annotate each of them as either "enjoyable" or "not enjoyable."

In order to transfer models for the platformer game to classify FPS levels, each FPS level was first transferred to a platformer level by mapping each FPS event to its most similar platformer event. Then, the transferred level was fed into models for classification.

In order to generate transferred FPS levels, a platformer level was first generated by probabilistic models, and then this level was transferred to an FPS level by mapping each platformer event to its most similar FPS event. A total of 200 FPS levels were generated in this way, with each model generating roughly 50 levels.

6.4 Result

Classification performance for each model is reported in Table 2. Precision is the fraction of levels recognized as fun by our models that are actually fun. Recall is the fraction of fun levels that are successfully identified by our models. F1 is the measure of model performance considering both precision and recall.

$$F1 = 2\frac{precision \cdot recall}{precision + recall}$$

Among all four models, unigram with clustering is the one with the least capability of classifying whether a level is "enjoyable." It is caused by the fact that event clustering averages over all events within the same cluster. Compared to unigram with clustering, unigram-position model has the better performance in terms of precision and F1 score. Cluster 7 is the group of easiest events for the platformer game. When events from cluster 7 are present in "fun" levels, they are more likely to appear at the beginning of the level ($P(c_7|fun, i = 1) = 0.037$, $P(c_7|fun, i = 5) = 0.019$). When those events are present in "not fun" levels, they tend to appear towards the end of the level ($P(c_7|not\ fun, i = 1) = 0.017$, $P(c_7|not\ fun, i = 5) = 0.06$). This shows that absolute position for a certain event or a cluster of events is important for level classification. Bigram with clustering also shows good classification performance, because it can model dependency between two clusters.

Table 2: Classification performance for each platformer model and the percentage of "enjoyable" platformer levels generated by each model

models	precision	recall	F1	enjoy
unigram	0.727	0.762	0.744	60.4%
unigram,clustering	0.609	0.667	0.636	47.8%
unigram-position, clustering	0.824	0.667	0.737	60.4%
bigram, clustering	0.667	0.857	0.750	54.7%
baseline	0.500	0.500	0.500	46.0%

Table 3: Classification and generation performance of transferred FPS models

models	recision	recall	F1	enjoy
unigram	0.600	0.818	0.692	56.3%
unigram,clustering	0.560	0.636	0.596	45.1%
unigram-position, clustering	0.714	0.682	0.697	51.0%
bigram, clustering	0.620	0.818	0.705	53.9%
baseline	0.500	0.500	0.500	44.0%

The percentage of "enjoyable" levels generated by each model is also reported in Table 2 (last column). The baseline is the percentage of "enjoyable" levels when levels are generated completely at random. Unigram with clustering has similar generation quality to baseline because the averaging effect of events clustering will make less favorable events appear more often and more favorable events appear less often. When the bigram model is used, the improvement on generation quality is not as significant as the improvement on classification performance. The lack of position information in the bigram model makes it generate unenjoyable levels starting with difficult events.

Table 3 shows the classification and generation performance of the transferred FPS models. All performance measures are higher than the baseline, which shows the knowledge that probabilistic models learned from the platformer game is transferred to the FPS game. It should be noted that most of the performance measures for the transferred FPS models are lower than the corresponding

performance measures for the original platformer models. One reason for this is that for some platformer events, the KL-divergence between them and their most similar FPS events are not close to zero, thus providing challenges that are not very similar to players. In the future, we plan to analyze how the transferred models improve performance as the divergence between the event from one game and its most similar event from another game decreases.

7 CONCLUSION

In this paper, we propose an approach to measure the similarity between game events from different games quantitatively. Players' performance on each event can be modeled by a probabilistic distribution. The performance distribution can characterize the difficulty of the challenge presented. KL-Divergence is used to calculate the divergence between two performance distributions, thus measuring the similarity between events from different games. The results from our experiments demostrate that knowledge and models from one game can be applied to another game once the similarity between events from those games is explored.

In the next step, events can be characterized by a more complicated model other than the performance distribution. The limitation of the current approach is that we must have a unified measure of players' performance in different games to calculate event similarity. One possible solution is to adopt a cognitive model with which a game event can be characterized by the cognitive process to complete the challenge. Now, the information presented in each event is only viewed as challenges provided to the player to satisfy the need for Competence or create the experience of flow. The needs for Autonomy and Relatedness can also be explored. For example, how the information presented in a event contributes to the storytelling of Role Player Game or elicits certain emotions can be investigated. This would require an investigation of semantic information presented in each event [7].

REFERENCES

[1] Sami Abuhamdeh and Mihaly Csikszentmihalyi. 2012. The Importance of Challenge for the Enjoyment of Intrinsically Motivated, Goal-Directed Activities. *Personality and Social Psychology Bulletin* 38, 3 (2012), 317–330. https://doi.org/10.1177/0146167211427147

[2] Barbaros Bostan and Sertaç Öğüt. 2009. Game challenges and difficulty levels: lessons learned From RPGs. In *International simulation and gaming association conference*.

[3] William B. Cavnar and John M. Trenkle. 1994. N-Gram-Based Text Categorization. In *In Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*. 161–175.

[4] Ben Cousins. 2004. Elementary game design. *Develop magazine* (2004), 51–54.

[5] Mihaly Csikszentmihalyi and Isabella Csikszentmihalyi. 1975. *Beyond boredom and anxiety*. Vol. 721. Jossey-Bass San Francisco.

[6] Steve Dahlskog, Julian Togelius, and Mark J. Nelson. 2014. Linear levels through n-grams. In *Proceedings of the 18th International Academic MindTrek Conference on Media Business, Management, Content & Services - AcademicMindTrek '14*. ACM Press. https://doi.org/10.1145/2676467.2676506

[7] Simon D'Alfonso. 2011. On Quantifying Semantic Information. *Information* 2, 1 (jan 2011), 61–101. https://doi.org/10.3390/info2010061

[8] Milton Friedman. 1939. A Correction: The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance. *J. Amer. Statist. Assoc.* 34, 205 (mar 1939), 109. https://doi.org/10.2307/2279169

[9] Jan Hauke and Tomasz Kossowski. 2011. Comparison of values of Pearson's and Spearman's correlation coefficients on the same sets of data. *Quaestiones geographicae* 30, 2 (2011), 87–93. https://doi.org/10.2478/v10117-011-0021-1

[10] Stephen C Johnson. 1967. Hierarchical clustering schemes. *Psychometrika* 32, 3 (1967), 241–254. https://doi.org/10.1007/bf02289588

[11] Mehmed Kantardzic. 2011. *Data Mining*. John Wiley & Sons, Inc. https://doi.org/10.1002/9781118029145

[12] Mohammad M. Khajah, Brett D. Roads, Robert V. Lindsey, Yun-En Liu, and Michael C. Mozer. 2016. Designing Engaging Games Using Bayesian Optimization. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 5571–5582. https://doi.org/10.1145/2858036.2858253

[13] Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics* 22, 1 (1951), 79–86. https://doi.org/10.1214/aoms/1177729694

[14] Scott Rigby and Richard M Ryan. 2011. *Glued to games: How video games draw us in and hold us spellbound: How video games draw us in and hold us spellbound*. ABC-CLIO. https://doi.org/10.5860/choice.49-0099

[15] Richard M Ryan, C Scott Rigby, and Andrew Przybylski. 2006. The motivational pull of video games: A self-determination theory approach. *Motivation and emotion* 30, 4 (2006), 344–360. https://doi.org/10.1007/s11031-006-9051-8

[16] Noor Shaker, Georgios N. Yannakakis, and Julian Togelius. 2013. Crowdsourcing the aesthetics of platform games. *IEEE Transactions on Computational Intelligence and AI in Games* 5 (2013), 276–290. https://doi.org/10.1109/tciaig.2012.2231413

[17] Sam Snodgrass and Santiago Ontanon. 2016. An approach to domain transfer in procedural content generation of two-dimensional videogame levels. In *Twelfth Artificial Intelligence and Interactive Digital Entertainment Conference*. Twelfth Artificial Intelligence and Interactive Digital Entertainment Conference.

[18] Michael Soppitt and Graham McAllister. 2011. Understanding Player Experience using Sequential Analysis.. In *DiGRA Conference*. DiGRA Conference.

[19] Guenter Wallner. 2015. Sequential Analysis of Player Behavior. In *Proceedings of the 2015 Annual Symposium on Computer-Human Interaction in Play (CHI PLAY '15)*. ACM, New York, NY, USA, 349–358. https://doi.org/10.1145/2793107.2793112

[20] Mark J. P. Wolf. 2001. Genre and the video game. *The medium of the video game* 1 (2001).