Extending Neuro-evolutionary Preference Learning through Player Modeling

Héctor P. Martínez, Kenneth Hullett, and Georgios N. Yannakakis, *Member, IEEE*

***Abstract*—In this paper we propose a methodology for improving the accuracy of models that predict self-reported player pairwise preferences. Our approach extends neuro-evolutionary preference learning by embedding a player modeling module for the prediction of player preferences. Player types are identified using self-organization and feed the preference learner. Our experiments on a dataset derived from a game survey of subjects playing a 3D prey/predator game demonstrate that the *player model-driven preference learning* approach proposed improves the performance of preference learning significantly and shows promise for the construction of more accurate cognitive and affective models.**

I. Introduction

Academic and industry interest in adaptive gaming has grown significantly over the last few years. Several research studies have demonstrated that the inclusion of adaptive processes in game dynamics can enhance player experience [1], [2], [3], [4]. Commercial games embed similar principles by adapting certain game characteristics based on player performance. For instance, difficulty is tailored to an estimator of the player's skill by adjusting the number of enemies in *Max Payne*, the likelihood of an enemy dropping a lifepack in *SiN Episodes: emergence*, the likelihood of getting good power ups in *Mario Kart*, and the spawn point positions in *Left 4 Dead*.

The key in designing games that tailor themselves to the player experience lies in the implementation of adaptive mechanisms which decide *whether*, *how* and by *how much* the game (i.e. specific controllable features of it) needs to be adjusted. The very first step towards designing efficient player experience-centered adaptive games is to assess the player's emotional state and cognitive focus and develop predictors of player experience. For this purpose, it is advisable that multimodal user input (e.g. user physiology or user gameplay actions) is collected and fused to derive indicators for certain aspects of the playing experience or the player, such as player skill level [5], level of frustration [6], or level of satisfaction [4]. In addition, players may report their experience via a survey consisting of questions given in a scaling or preference format — this paper considers *self-reported preferences* that may be combined with relevant statistical features (derived from user input) to construct computational predictors of the user's reported preferences [7], [8]. Predicting the player's preference can be a complex task, which is further complicated when the target function is one of the players' affective states (e.g. frustration). Learning to predict preferences of users using neuro-evolutionary preference learning has provided highly accurate computational models of player experience in several dissimilar studies (e.g. see [1]). This paper proposes an extension to this methodology to enhance the accuracy obtained.

A player model may classify the interaction between the player and the game into a number of different player behavior types. Our hypothesis is that embedding a player model generated via self-organization to the neuro-evolutionary player preference learner may result in more accurate estimators of player preferences. To test our hypothesis we collected gameplay data and pairwise emotional preferences from thirty six subjects playing a 3D prey/predator game. The results presented in this initial study show that the hypothesis holds for five out of six affective states investigated and provide promise for highly accurate predictions of user preferences in games and beyond.

The paper is organized as follows: Section II reviews previous work on preference learning and player modeling; Section III presents the methodology and the individual components of our approach while Section IV provides details on the dataset that our approach is tested on. Experiments and conclusions derived are presented in Section V and Section VI respectively.

II. Related Work

This paper examines improving the accuracy of affective state predictors trained on users' preferences by incorporating a player model in the preference model creation process. This section reviews related work on both player modeling and preference learning.

A. Player Modeling

Different approaches of user modeling in games, namely *player modeling*, have been utilized to provide playing experiences tailored to the type of player identified. Charles et al. [9] motivate for the inclusion of player modeling in the design of adaptive games and propose a framework in which player modeling becomes a vital component of game design. Additionally, they conclude that players learn and evolve as they play; therefore, the existence of dynamic player models that adjust during play are necessary for successfully capturing complex gameplay dynamics. In [3], a player model is used to dynamically adapt the story in a role-playing game (RPG). Five types of players are defined related to five different styles of play. The class of the player is updated during gameplay based on specific player decisions

HPM and GNY are with the Center for Computer Games Research at the IT University of Copenhagen, Denmark and KH is with the Computer Science department at UC Santa Cruz, USA (hpma@itu.dk, yannakakis@itu.dk, khullett@soe.ucsc.edu).

978-1-4244-6297-1/10/$26.00

and actions such as, for instance, when a player asks for a reward in exchange of assistance.

Predefined fixed classification of players is designer dependent since the designer has to decide the types of players and hand-craft the events that determine the type that the player belongs to. The complexity of this task increases when the game does not contain clear events or defined types of players (e.g casual games). Using in-game indicators such us indexes of player performance or physiological signal characteristics may generate player models that can be used to adjust game parameters during play, even for changing types of playing behavior. On that basis Yannakakis and Maragoudakis [4], designed Bayesian networks relying on player performance features, such us player score and time played in a Pacman game, to cluster players, which in turn modified certain game parameters to maximize the *interestingness* of the play. Using users' trail information, Thawonmas et al. [10] investigate groups of players with shared interests clustered via a self-organizing map (SOM). A variant of SOMs, Emergent SOMs (ESOM), is also utilized for identifying dissimilar player types in the popular game *Tomb Raider: Underground* (TRU) [11]. In that study, the player model is built on statistical playing features (e.g. completion time, number of deaths, etc.) of thousands of TRU players.

B. Preference Learning

As a first approach to predict preferences, the game designer might use a player model and hand-craft a set of player preferences for each type of player; e.g. in [3] it is implied that a player classified as a *fighter* will tend to prefer to encounter challenging monsters. This approach relies on the assumption that the defining characteristics of a player cluster implicitly hold information about the preferences of the members integrated in the model.

Alternatively, for the approach followed in this study (*preference learning*) [7], [8], players are requested to explicitly report their preferences on variants of the game via questionnaires, and computational models are built on the preference data. Based on this data-driven preference modeling approach, Sharma et al. [12] monitor player traces during episodes of a videogame and collect player opinions (including enjoyment and lack of interest) after each game session is completed. During the game, the current trace of the player is compared to the information stored from earlier players and the story most likely to be preferred by the specific player is generated. On a similar basis Yannakakis [13] presents a methodology for learning self-reported preferences of users. Experiment participants play a pair of games and report which one was preferred under a set of affective states. Reported preference and multimodal input data are used to train a computational model for predicting the emotional preferences of the subjects. This methodology has been used successfully in a number of dissimilar studies including the prediction of affective states (i.e. challenge, fun, and frustration among others) of Super Mario Bros players [1] using playing characteristics (e.g. number of coins gathered) and the prediction of reported challenge in a 3D prey/predator game relying on gameplay information (e.g. number of enemies visible on screen) [14].

This paper extends upon previous work, combining the notions of player modeling and preference learning within a single mechanism, named *player model-driven preference learning*, that assists in predicting player preferences more accurately.

III. Method

The scheme on Fig. 1 depicts the main components of the proposed methodology for learning preferences from players. The upper box illustrates the player modeling extension while the preference learning process is depicted in the lower box. In our approach, a set of statistical features is extracted from recorded game data of players of the game, namely *gameplay features*. These are processed by a feature selection algorithm which chooses the most appropriate subset that yields the highest performing preference model. The preference model is trained to learn the mapping between this feature subset and the preference data reported by the players of the game. In our extended version of preference learning, another subset of gameplay features is selected to create a player model that feeds the preference model with a player classification. Creating this additional component provides the preference predictor with an explicit indicator of the player behavior or style.

More specifically, in this study we utilize an ESOM for modeling players and neuro-evolution for learning player preferences. The final preference model, after the two-phase approach is completed, is a non-linear Single Layer Perceptron (SLP) that predicts preferences trained on selected statistical gameplay features and the type of player given by the ESOM — which also uses another selected subset of features to classify the players. Appropriate statistical features for both the ESOM and the SLP are chosen by a Sequential Forward Selection (SFS) feature selection method. The three main components of the *player model-driven preference learning* are described in the remainder of this section.

A. Feature Selection: Sequential Forward Selection

SFS is a bottom-up search procedure where one feature is added at a time to the current feature set. The feature to be added is selected from the subset of the remaining features such that the new feature set generates the maximum value of the performance function over all candidate features for addition. The performance function considered for SFS varies for the two mechanisms examined: the quantization and topographic errors of the ESOM are used to assess the quality of considered feature sets (see Section III-B) whereas the average cross validation performance on unseen folds of data is the corresponding performance measure for the SLP (see Section III-C).

The feature selection stage is essential for finding the minimal (and most appropriate) subset of features that maximize the predictability of the preference model since we

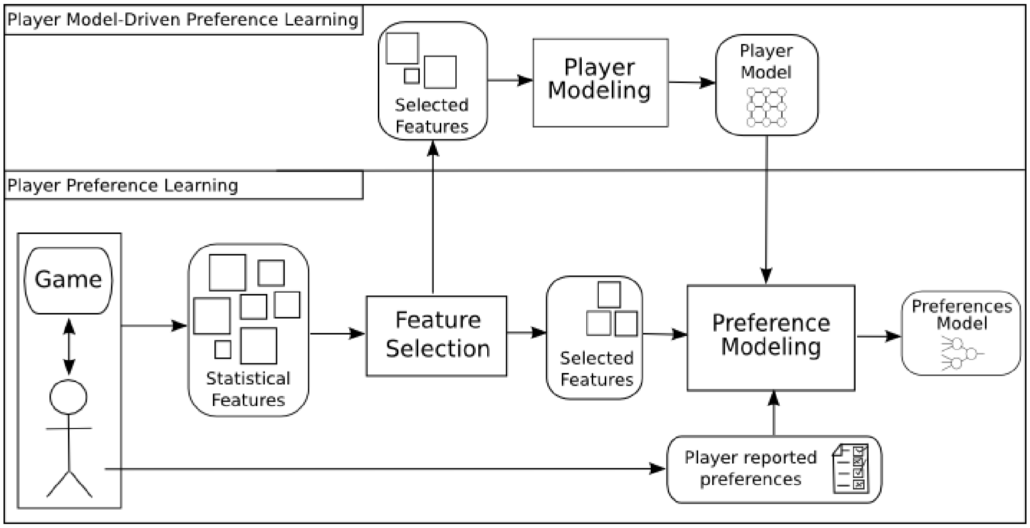


Fig. 1. Player model-driven preference learning.

desire to keep the size of the model small and thereby real-time efficient. We would like our model to be dependent on as few features as possible, both to make it easier to analyze, and to make it more useful for incorporation into future implementations of real-time adaptive mechanisms that tailor the game to the gameplay experience needs of the player. Moreover, there is evidence that omitting unnecessary inputs improves the learning quality of neuro-evolution ([15] among others). Therefore, feature selection is utilized to find the feature subset that yields the most accurate user model and save computational effort of exhaustive search on all possible feature combinations. The quality of the predictive model constructed by the preference learning outlined above depends critically on the set of input data features chosen.

B. Player Modeling: Self-Organizing Maps

The self-organizing map [16] iteratively adjusts a low dimensional projection of the input space via vector quantization [17]. A SOM consists of neurons organized in a low (2 or 3) dimensional grid. Each neuron in the grid (map) is connected to the input vector through a d-dimensional connection weight vector $\mathbf{m} = \{m_1, \ldots, m_d\}$ where d is the size of the input vector, $\mathbf{x}$. In addition to the input vector, the neurons are connected to neighbor neurons of the map through neighborhood interconnections which generate the structure of the map.

The training of the map is based on the principles of competitive learning which allows the SOM to find associations between the inputs presented and project them in a topographic map which is a function of the trained weight vector. The training outcome of a SOM is the projection of the input data to the two dimensional space (map) in which neighboring neurons have similar weight vectors. What differentiates SOM from a standard vector quantization algorithm (e.g. k-means) is the update of the topological neighbors of the best-matching neuron: i.e. the whole neuron neighborhood is stretched towards the presented input vector. For a more detailed description of SOMs, the reader is referred to [16].

The topology preservation of the SOM projection is of little use when small SOMs are employed and the advantage of neighbor-neuron relation is neglected making a small SOM almost identical to k-means. Using large SOMs — called Emergent Self-Organizing Maps [18] to emphasize the distinction — and reliable visualization techniques help in identifying clusters in the low-dimensional projection of the data. We use the batch algorithm for training the ESOM[1]. In all experiments presented in this paper the toroid topology is used and neurons are interconnected within the map in a rectangular grid (i.e. each neuron has four immediate neighbors).

While there are numerous clustering performance measures proposed in the literature providing dissimilar indications for the properties of the generated clusters, no measure can guarantee approximation of the performance with high accuracy. In this study we choose the average quantization error and the topographic error [16] as measures of ESOM training performance. More specifically, the quantization error (QE) equals $\frac{1}{N}||\mathbf{x} - \mathbf{m_c}||$ across all N data samples, where $\mathbf{m_c}$ is the weight vector of the best-matching neuron. Topographic error (TE) measures topology preservation of the map and is calculated as the proportion of all input data vectors for which the first and second best-matching neurons are not adjacent [16].

Data clusters are drawn on the two-dimensional map and input data is placed on the cluster that contains the input's best-matching neuron. In this study, clusters are selected by observing the distance between best-matching neurons and the density of best-matching neurons depicted by the *U-matrix* and *P-matrix*, respectively. The U-matrix represents

[1]The *databionic ESOM software tool* [19] is used for training and visualizing the ESOM

the sum of the distances (Euclidean in this study) between the weight vector of each map neuron of the SOM and the weight vectors of its immediate neighbors. Areas on the SOM with high U-matrix values suggest that there is a boundary between clusters whereas areas with low U-matrix values suggest the existence of a cluster. The P-matrix representation depicts a measure of the input data local density on the 2D map given by the Pareto Density Estimation [20]. Areas with high density correspond to regions with a high number of best-matches suggesting that those neurons belong to the same cluster.

C. Preference Modeling: Neuro-Evolution

Single-layered perceptrons were trained to approximate an assumed unknown non-linear function that maps selected statistical gameplay features and clusters of playing behavior to dissimilar self-reported affective states given in preferences. The SLP is the simplest form of a fully-connected feed-forward neural network; it consists only of an output layer. In this study the output layer consists of a single neuron which employs the sigmoid (logistic) activation function. Connection weights take values from -5 to 5 to match the normalized input values that lie in the [0, 1] interval.

The training data samples consist of a set of gameplay features and player types as input and corresponding reported preferences as output. Given that reported preferences do not provide prescribed target outputs but only specify which game is preferred in each pair of variants ANN training algorithms such as back-propagation are inapplicable. Learning is achieved through artificial evolution by following the preference learning approach presented in [13].

A generational genetic algorithm (GA) is implemented, using a fitness function that measures the difference between the subject's reported affective preferences and the relative magnitude of the corresponding model (ANN) output. More specifically, the logistic (sigmoidal) function $g(\delta_e, \epsilon) = 1/(1 + e^{-\epsilon\delta_e})$ is used where $\delta_e = e(A) - e(B)$ is the difference of the ANN output values (investigated affective state) between game A and game B; $\epsilon = 30$ if A is preferred, $A \succ B$, and $\epsilon = 5$ when B is preferred, $A \prec B$. Both the sigmoidal shape of the objective function and its selected ϵ values are inspired by its successful application as a fitness function in neuro-evolution preference learning problems [21], [2].

A population of 1000 individuals is used and evolution runs for 5 generations. Due to the large GA population and the small ANN size used, running the algorithm for a higher number of generations does not produce a significant improvement on the classification accuracy of the SLPs. A probabilistic rank-based selection scheme is used, with higher ranked individuals having higher probability of being chosen as parents. Reproduction is performed by uniform crossover, followed by Gaussian mutation with a 5% probability relying upon earlier successful parameter tuning experiments [21], [2].

The performance of SLPs is measured through the average classification accuracy of the model in three independent runs using 3-fold cross validation. In each run of the cross-validation, 3 networks are trained and the highest classification accuracy is used in the calculation of the performance.

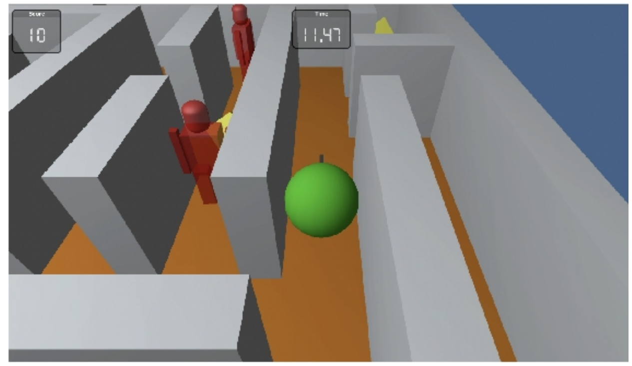


Fig. 2. Variant 3 of the computer game MazeBall. The camera parameters for this variant are as follows: height equals 15, distance equals 6, and frame coherence equals 0.1. Height and distance are measured in *Unity*[3] meters whereas frame coherence is a weight that affects the speed of the camera transitions.

IV. Data Set

The data set used in this study was gathered via user survey experiments of 36 subjects playing the MazeBall game[2]. MazeBall is a 3D prey/predator game where the player guides a ball through a maze. Golden tokens can be collected to increase the player's score, while red enemies that move around the maze decrease the player's score when they come in contact with the ball. The virtual camera embedded in the game is defined by three parameters (height, distance and frame coherence) that can be modified by the designer/experimenter to create different variants of the game (see Fig. 2).

Sequences of two 90-second game variants with different camera parameters are played. After each pair of game variants is completed, players are questioned to express their preferred game with regards to six affective states, *anxiety, challenge, excitement, frustration, fun* and *relaxation*, via 4-alternative forced choice (4-AFC) questionnaire items. 4-AFC allows subjects to either express their clear preference — i.e. either the first or the second game (pairwise preference or 2-AFC) is preferred — or to express their preference for both games equally or neither game. More details about the experimental protocol and the self-reported data can be found in [22].

A. Extracted Features

Several game metrics are logged for each game, including elements of the game state and the player's inputs (keystrokes), and various statistical features are extracted. These gameplay features are listed in detail in this section.

- **Performance**: the final score (S), the percentage of the grid explored (G), and the percentage of paths covered

[2] the game is available online at http://www.itu.dk/~yannakakis/MazeBall.html

[3] http://unity3d.com/

several times (P) (calculated by dividing the number of explored cells of the grid by the times the player leaves a cell).

- **Time**: average and standard deviation of time intervals the player stays in certain cell (t^c) and the number of these intervals that are greater than 0.5, 0.6, 0.7, 0.8, 0.9 and 1.0 seconds ($t^c_{0.5}$, $t^c_{0.6}$, $t^c_{0.7}$, $t^c_{0.8}$, $t^c_{0.9}$, $t^c_{1.0}$ respectively).
- **Space**: average and standard deviation of the Euclidean distance between the ball and the closest token (D^ϵ_t) and between the ball and the closest enemy (D^ϵ_e), average and mean of the standard deviation of the Euclidean distance to all enemies ($D^\epsilon_{\forall e}$), average and standard deviation of the manhattan distance between the ball and the closest token (D^μ_t) and between the ball and the closest enemy (D^μ_e), average and the mean of the standard deviation of the manhattan distance to all enemies ($D^\mu_{\forall e}$).
- **Input**: number of right ($90°$), left ($-90°$) and $180°$ turns divided by the times the right, left and down key arrows were pressed respectively (ω_{90}, ω_{-90}, ω_{180}), number of times the up arrow (K_{up}) or space bar (K_{space}) key were pressed, average and standard deviation of the time that either the right or the left arrow keys were held down (t^K).
- **Camera**: the average and standard deviation of visible enemies (V_e), visible tokens (V_t) and visible paths (V_p), camera profile parameters: height (H), distance (D), and frame coherence (F_c).

V. Experiments

This section first presents the types of MazeBall players found with the ESOM player modeling tool and then analyzes the preference predictors assisted by the player types found. All extracted gameplay features are pre-processed and presented to automatic feature selection as follows: data is uniformly normalized to [0,1] after setting outlier values to 0 and 1 if they are, respectively, below a minimum or above a maximum threshold defined by the 95% of the values.

The data set used contains 122 pairs of games with dissimilar number of clear (2-AFC) preferences for each affective state reported. Note that ESOM is trained on all games, 244 samples in total, while the SLP is trained only on pairs of clear (2-AFC) preferences reported for the corresponding affective state: 97, 83, 86, 92, 90 and 90 pairs of data samples are utilized, respectively, for challenge, excitement, anxiety, fun, relaxation and frustration.

A. *Player Modeling with ESOM*

Player modeling considers all gameplay features related to the player's behavior in the game; i.e. we exclude the *camera* and *input* features from our investigations to make the model dependent on the player performance and not on the specific game configuration played. As mentioned in Section III, we apply SFS to find the minimal set of features that can provide a good clustering of the data samples. Because of the non-deterministic nature of the algorithm, the ESOM is trained

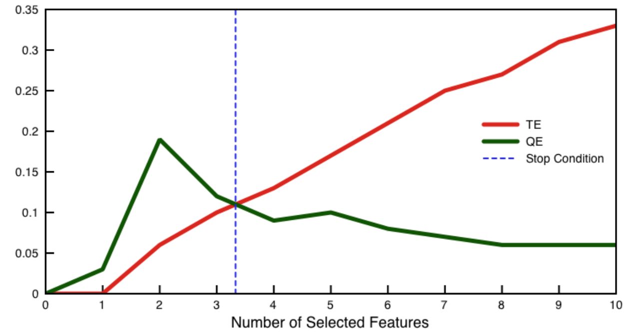


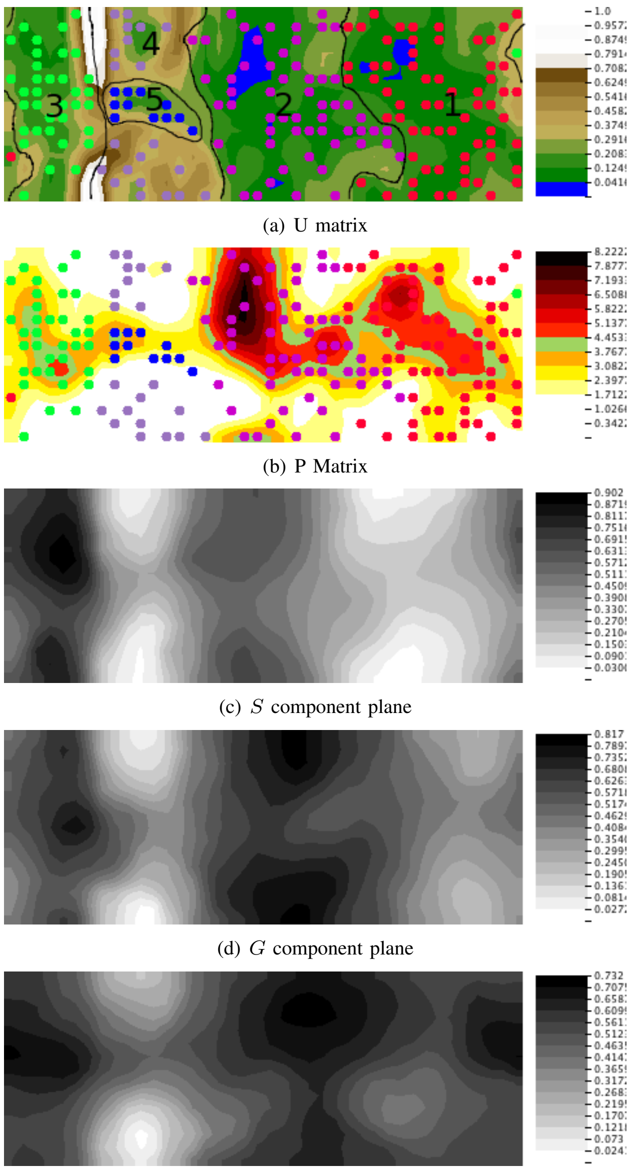
Fig. 3. Average topographic error (TE) and average quantization error (QE) for the best subset of features in consecutive iterations of SFS.

10 times each time a new subset of features is considered by SFS. The topographic error and the quantization error are averaged and used to assess the quality of the feature subset considered. In each iteration of SFS, the subset of features that generates the minimum average TE is selected. The algorithm stops when the average QE is greater than the average TE (see Fig. 3).

Toroid-shaped ESOMs consisting of neurons organized in a 15×40 rectangular lattice are trained with the batch algorithm. Fig. 3 shows the evolution of average TE and QE along 10 iterations of SFS which is stopped after selecting 3 features: final score (S), percentage of grid explored (G) and average number of times the player stays in a cell more than than 0.6 seconds ($t^c_{0.6}$).

Five types of players are defined by observing the U-matrix and P-matrix of the ESOM depicted in Fig. 4). The values of the U-matrix are normalized into [0,1] and plotted as a topographic map in which higher values of the Euclidean distances between the weight vectors of neighboring neurons are shaded with white and brown colors creating mountains on the map and neurons with smaller neighbor weight vector distances are depicted as sea (blue color) and green valleys. On the other hand, the P-matrix is a heat map in which black and dark red colors represent areas of high density of best-matching neurons whereas low density areas are colored in white. Finally, the component planes depicted in Fig. 4 show the values of the connection weights of every neuron on the ESOM as a grayscale spectrum: the highest values of the connection weight are colored black and the lowest values are colored white.

Table I and the component planes in Fig. 4 show the characteristics of the five classes of players. The samples of cluster 1, named $Wary$, correspond to players that stop for more than 0.6 seconds a high number of times (high $t^c_{0.6}$ values) and have a low score (S) at the end of the game. Players of type $Explorer$ (cluster 2) tend to explore a big fraction of the maze (high G values) while stopping frequently. The $Winner$ (cluster 3) players present similar characteristics to the $Explorer$ players but with higher final scores. The cluster $Impetuous$ (cluster 4) is assigned to players characterized by continuous movement (low $t^c_{0.6}$



(a) U matrix

(b) P Matrix

(c) S component plane

(d) G component plane

(e) $t^c_{0.6}$ component plane

Fig. 4. Visual representations of the best-performing ESOM: U-matrix, P-matrix and the three component (feature) planes. The highlighted small circles correspond to the best-matching neurons for the samples in the input dataset. The legends depict the color assigned in the representation to each value in the map.

values), low final scores and small fraction of the maze explored. Finally, $Neutral$ players (cluster 5) achieve (on average) higher final scores than $Wary$ players and demonstrate moderate values for G and $t^c_{0.6}$.

B. Player Model-Driven Preference Learning

This section presents a comparative study between standard neuro-evolutionary preference learning and preference learning relying on the player types identified via self-organization. For modeling player preferences (without the use of player models) we run automatic feature selection to

TABLE I

CHARACTERISTICS OF THE CLUSTERS DEFINED. S, G AND $t^c_{0.6}$ VALUES PRESENTED ARE AVERAGES ACROSS ALL MEMBERS OF THE CORRESPONDING CLUSTER.

Cluster	Name	# samples	S	G	$t^c_{0.6}$
1	Wary	73	0.24	0.54	0.75
2	Explorer	71	0.46	0.78	0.78
3	Winner	44	0.76	0.76	0.81
4	Impetuous	23	0.15	0.31	0.4
5	Neutral	13	0.35	0.58	0.64

select features for the SLP model (see Section III). For each reported affective state we run SFS 25 times on the complete set of statistical gameplay features. The performance value is the average of all 25 3-fold cross validation accuracies (see Section III-C) and SFS stops when adding a new feature does not increase the performance. At the end of each SFS run, SLPs are trained ten times, with the selected subset of features as input, and the one with the highest performance is selected.

A similar experiment runs for the player model-driven preference learning approach as proposed in Section III. The ESOM is treated as an additional triplet of features for SFS since it is connected to the SLP via five additional SLP binary inputs representing the five different player clusters identified.

The average performance of the 25 SLPs for each affective state on the two experiments is shown in Table II. For comparison purposes, we train 25 SLPs whose input solely consists of the five classes derived from the ESOM and present their average performance at the same table. Results obtained generate similar conclusions for all six affective states investigated. It is apparent that using only the type of the player to predict preferences results to very low accuracy values: the lowest is found for challenge where only a 30.23% of the unseen reported preferences are estimated correctly; the performance for other 5 affective states is over 40% and reaches chance level for frustration (49.06%). These results indicate that additional gameplay features are required for the training of the preference models and furthermore that the ESOM approximation of player types is not sufficient for successful preference prediction.

Evidently when a subset of features selected through SFS is used to train SLPs accuracy on predicting preferences improves greatly. For excitement, fun, relaxation and frustration cross validation accuracy is around 80% which suggest that SLPs are rather successful predictors of those affective states given the complexity of the affective modeling task and the noisy nature of the self-reported data. The performances across affective states follow the trend noted earlier. Frustration and challenge demonstrate, respectively, the maximum and minimum model performance obtained indicating the dependency between reported affective states and problem complexity.

When combining player modeling with preference modeling, expressed as the ESOM feeding the SLP model, the prediction performance of the resulting preference model is further improved for all six affective states when compared to the performances generated by the SLP per se. These performance improvements are statistically significant (significance is 5% in this paper) for all affective states excluding the fun state. These statistically significant effects suggest that player model-driven preference learning can indeed improve the performance of the player preference predictions. Furthermore it is derived that the combination of self-organization (as a player modeling technique) and neuro-evolution (as a preference learner) is successful in capturing the association between gameplay features and reported preferences of players.

C. Inherited Knowledge in the Player Model

In the experiments presented earlier the SLPs are trained using 2/3 of the data (training set) while the ESOM is trained on the whole dataset. Therefore, when the ESOM feeds the SLP an implicit knowledge of the validation set might be introduced. Despite no preference information is included in that dataset we investigate the effect of this inherited knowledge on the prediction accuracy for challenge (the state with the most clear preferences) and fun (the affective state where our methodology produced lower improvement in the performance as presented in earlier results).

In the experiments presented in this section we apply player model-driven preference modeling on clear (2-AFC) preferences. The performance of the model is assessed using 1-fold cross validation: 2/3 of the clear preferences are randomly picked and used for training and the remaining third for validation.

It is noteworthy that the dissimilar number of clear preferences across the different affective states prevents us from using a single ESOM for all different investigated states, therefore we apply the same methodology followed in Section V-A to train two separate ESOMs — one for each affective state.

The results presented in Table III indicate that statistically significant performance improvements are achieved even when samples in the validation set are not used to train the player model. As a matter of fact, the performance improvement with the use of the player model-driven preference learning approach is greater now; this could be explained, in part, due to the use of only one fold in these case-studies but also due to the existence of noise incorporated through the unclear preference data that the ESOM was previously trained on. Additionally, the difference in performances due to different clusters defined suggests that the quality of the player classification may have an impact on the degree of preference prediction improvement.

TABLE III
AVERAGE PERFORMANCE OF 15 RUNS OF SFS FOR CHALLENGE AND FUN USING STANDARD NEURO-EVOLUTIONARY PREFERENCE LEARNING (SLP), AND PLAYER MODEL-DRIVEN PREFERENCE LEARNING (SLP+ESOM). THE P-VALUES PRESENTED CORRESPOND TO A 2-TAILED INDEPENDENT T-TEST BETWEEN SLP AND SLP+ESOM PERFORMANCE VALUES.

	SLP	SLP+ESOM	p-value
Challenge	65.63	68.33	0.0004
Fun	83.87	87.53	0.0003

VI. DISCUSSION AND CONCLUSIONS

This study examines whether it is possible to enhance the accuracy of a preference predictor (built using neuro-evolutionary preference learning) by embedding a player model (represented through a self-organizing map) to the predictor. The data our hypothesis was tested on is collected from a game experiment survey and consists of statistical features of game-player interaction with corresponding self-reported emotional preferences for the games played. Results obtained with the player model-driven preference learning approach proposed demonstrate a small, yet statistically significant, performance improvement in five out of the six affective states investigated.

Results suggest that including information of players who did not express a clear (2-AFC) preference for the game in the training data set helps towards predicting the clear preferences. One could claim that the performance improvement could be, in part, elicited by a implicit knowledge of the SLP validation set embedded in the data used for training the ESOM model. However, the ESOM is trained on all gameplay data corresponding to all possible preference choices of the 4-AFC (i.e. *both equally* and *neither* are included) which generates noise for the preference learning mechanism. Moreover, preferences are not taken into account during the training of the ESOM which suggests that no association between gameplay features and preferences is learnt. The results presented in Table II support, in part, these arguments, showing that preference predictors fail when relying solely on the type of the player inferred by the ESOM. In addition, Table III shows that the improvement remains when validation data is not used to train the ESOM in the two affective states tested: fun and challenge. All above suggest that the fraction of improvement due to the inherited knowledge of the validation set is minimal. We believe, however, that future empirical analysis will be required for this effect to be investigated further.

Results in Sections V-A and V-C suggest that the quality of the player classification has a high impact on the preferences prediction. For future work, different methods for clustering player data (e.g. decision tree clustering) will be examined and the impact of non-clear preferences on their combination with preference learning will be investigated.

The ANNs used in this paper for learning preferences are single layer perceptrons with one output neuron. When ESOMs are combined with SLPs 600 more neurons are added to the model; thus it might be that similar performance improvements could be achieved using multi layer perceptrons (MLP) on the preference learning process instead of a two-module system. Therefore, a future study will investigate the impact of an MLP-ESOM combination to the performance of preference prediction. Nevertheless, the combination of preference learning and self-organization maps investigated in this paper already presents an advantage regarding the

TABLE II

AVERAGE PERFORMANCE OF 25 RUNS OF SFS FOR EACH AFFECTIVE STATE USING STANDARD NEURO-EVOLUTIONARY PREFERENCE LEARNING (SLP), AND PLAYER MODEL-DRIVEN PREFERENCE LEARNING (SLP+ESOM). THE AVERAGE PERFORMANCE OF 25 SLPS FED SOLELY WITH ESOM GENERATED CLUSTERS (SLP$_{ESOM}$) IS PRESENTED FOR COMPARISON. THE P-VALUES PRESENTED CORRESPOND TO A 2-TAILED INDEPENDENT T-TEST BETWEEN SLP AND SLP+ESOM PERFORMANCE VALUES.

	Challenge	Excitement	Anxiety	Fun	Relaxation	Frustration
SLP	67.03	80.52	70.67	80.34	79.47	82.44
SLP+ESOM	69.49	82.47	73.52	80.77	81.42	84.15
p-value	0.005	0.001	0.0002	0.43	0.01	0.02
SLP$_{ESOM}$	30.22	46.07	41.52	42.62	41.96	48.67

interpretation and expressiveness of the data. While MLPs define black boxes, both the ESOM characteristics of the player types can be easily inferred and the impact of statistical features selected on the corresponding preferences can be easily analyzed within an SLP [1].

Future work also includes the investigation of more sophisticated methods of feature selection for fully exploiting the potential of player modeling for preference learning. Furthermore, the approach proposed needs to be evaluated across different data sets deriving from dissimilar player input modalities, interaction modes and game genres to test the extent of its generalizability.

ACKNOWLEDGMENTS

The authors would like to thank Martin Schwartz for the implementation of the camera control system in the MazeBall game and all participants of the survey experiment.

REFERENCES

[1] C. Pedersen, J. Togelius, and G. N. Yannakakis, "Modeling Player Experience for Content Creation," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 2, no. 1, pp. 54–67, 2010.

[2] G. N. Yannakakis and J. Hallam, "Real-time Game Adaptation for Optimizing Player Satisfaction," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 1, no. 2, pp. 121–133, June 2009.

[3] D. Thue, V. Bulitko, M. Spetch, and E. Wasylishen, "Interactive storytelling: A player modelling approach," in *Proceedings of the third Artificial Intelligence and Interactive Digital Entertainment conference (AIIDE-07)*, 2007.

[4] G. N. Yannakakis and M. Maragoudakis, "Player modeling impact on player's entertainment in computer games," *Lecture notes in computer science*, vol. 3538, p. 74, 2005.

[5] P. Spronck, M. Ponsen, I. Sprinkhuizen-Kuyper, and E. Postma, "Adaptive game AI with dynamic scripting," *Machine Learning*, vol. 63, no. 3, pp. 217–248, 2006.

[6] K. M. Gilleade and A. Dix, "Using frustration in the design of adaptive videogames," in *Proceedings of the 2004 ACM SIGCHI International Conference on Advances in computer entertainment technology*. ACM, 2004, p. 232.

[7] J. Doyle, "Prospects for preferences," *Computational Intelligence*, vol. 20, no. 2, pp. 111–136, May 2004.

[8] J. Fürnkranz and E. Hüllermeier, "Preference learning," *Künstliche Intelligenz*, vol. 19, no. 1, pp. 60–61, 2005.

[9] D. Charles, A. Kerr, M. McNeill, M. McAlister, M. Black, J. Kcklich, A. Moore, and K. Stringer, "Player-centred game design: Player modelling and adaptive digital games," in *Proceedings of the Digital Games Research Conference*, vol. 285, 2005.

[10] R. Thawonmas, M. Kurashige, K. Iizuka, and M. Kantardzic, "Clustering of Online Game Users Based on Their Trails Using Self-organizing Map," *Lecture Notes in Computer Science*, vol. 4161, p. 366, 2006.

[11] A. Drachen, A. Canossa, and G. N. Yannakakis, "Player Modeling using Self-Organization in Tomb Raider: Underworld," in *Proceedings of the IEEE Symposium on Computational Intelligence and Games (CIG2009)*, 2009.

[12] M. Sharma, S. Ontanón, C. Strong, M. Mehta, and A. Ram, "Towards player preference modeling for drama management in interactive stories," in *Proceedings of the Twentieth International FLAIRS Conference (FLAIRS07)*, 2007.

[13] G. N. Yannakakis, "Preference Learning for Affective Modeling," in *Proceedings of the Int. Conf. on Affective Computing and Intelligent Interaction (ACII09)*, Amsterdam, The Netherlands, September 2009.

[14] M. Schwartz, H. P. Martinez, G. N. Yannakakis, and A. Jhala, "Investigating the Interplay between Camera Viewpoints, Game Information, and Challenge," in *Proceedings of Artificial Intelligence and Interactive Digital Entertainment (AIIDE09)*. AAAI, October 2009.

[15] J. Togelius, T. Schaul, J. Schmidhuber, and F. Gomez, "Countering poisonous inputs with memetic neuroevolution," in *Parallel Problem Solving From Nature 10*, 2008.

[16] T. Kohonen, "The self-organizing map," *Neurocomputing*, vol. 21, no. 1-3, pp. 1–6, 1998.

[17] R. M. Gray, "Vector quantization," *IEEE ASSP Magazine*, pp. 4–29, 1984.

[18] A. Ultsch, *Kohonen Maps*, 1999, ch. Data Mining and Knowledge Discovery with Emergent Self-Organizing Feature Maps for Multivariate Time Series, pp. 33–46.

[19] A. Ultsch and F. Mörchen, *ESOM-Maps: tools for clustering, visualization, and classification with Emergent SOM*. Dept. of Mathematics and Computer Science, University of Marburg, No. 46, Tech. Rep., 2005.

[20] A. Ultsch, "Maps for the visualization of high-dimensional data spaces," in *Proc. Workshop on Self organizing Maps*, 2003, pp. 225–230.

[21] G. N. Yannakakis, M. Maragoudakis, and J. Hallam, "Preference learning for cognitive modeling: a case study on entertainment preferences," *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, vol. 39, no. 6, pp. 1165–1175, 2009.

[22] H. P. Martinez, A. Jhala, and G. N. Yannakakis, "Analyzing the Impact of Camera Viewpoint on Player Psychophysiology," in *Proceedings of the Int. Conf. on Affective Computing and Intelligent Interaction*. Amsterdam, The Netherlands: IEEE, September 2009, pp. 394–399.