Player Profiling using Hidden Markov Models Supported with the Sliding Window Method

Alper Kilic, Mehmet Akif Gunes and Sanem Sariel
Istanbul Technical University, Istanbul, Turkey

Keywords: Player Profiling, Hidden Markov Model, Sliding Window Method, the Vindinium Game.

Abstract: In this paper, we present a player profiling system applicable for both human players and bots in video games. The Vindinium artificial intelligence (AI) contest is selected as the test-bed for analyzing the performance of our system. In this game, AI bots compete with each other in a systematically generated environment to achieve the highest score. Our profiling method is based on Hidden Markov Model (HMM) constructed by using consecutive actions of AI bots and improved with the initial training phase and our sliding window approach. The method is evaluated for three different performance criteria: recognition of bots, grouping bots that have similar game styles and tracking changes in the strategy of a single bot through the game. The results indicate that the method is promising with 90,04% binary classification success in average.

1 INTRODUCTION

Video games have become to take a very important place in our daily lives as a form of entertainment especially in the last 10 years with the development of software and hardware technologies.

The growing increase in the number of users at every age group and the market share reaching billions of dollars have accelerated competition in the gaming industry to a high level and have led researchers and companies do advanced game/player analyses. These analyses have revealed out the importance of modelling and classifying players to provide them more engaging contents, to increase their play time, to present more intelligent non-player characters and to identify which game janras they are tending towards (D. Kennerly, 2003), (K.S.Y. Chiu, and K.C.C. Chan, 2008).

Data collection about players to create a successful player model by carrying out surveys have remained inadequate at present. It has been shown that in-game user interaction data are more valuable to analyse and model players (Yannakakis, G. N.; Spronck, P.; Loiacono, D.; and Andre, E. 2013). That has increased the number of scientific work on player modelling using supervised and unsupervised machine learning methods (S. C. Bakkes, P. H. Spronck, and G. van Lankveld, 2012).

In this work, we propose an HMM-based player profiling method. We selected the Vindinium game environment as a test-bed for evaluations. This is due to the extensive data that can be acquired through the website of the game and the ease of validation of our method. Our contributions include a sliding window approach for the evaluation of HMM results for player in-game action sequences and the flexibility provided in the HMM emission matrix formation.

The rest of the paper is organised as follows. Section 2 mentions related research works in the area. Section 3 and section 4 define the problem and present the selected test-bed game (Vindinium), respectively. Section 5 presents the proposed method and the implementation details. Section 6 introduces the uses of the method and the experiment results. We conclude the paper in section 7.

2 RELATED WORK

Matsumoto and Thawonmas (Yoshitaka Matsumoto, Ruck Thawonmas, 2004) have used temporal action sequences to classify players with a supervised Hidden Markov Model. In their study, there are three basic player classes which have distinguishing characteristics. They have more successful results

Kilic, A., Gunes, M. and Sariel, S.
Player Profiling using Hidden Markov Models Supported with the Sliding Window Method.
DOI: 10.5220/0005711403620369
In *Proceedings of the 8th International Conference on Agents and Artificial Intelligence (ICAART 2016) - Volume 2*, pages 362-369
ISBN: 978-989-758-172-4

from their previous work (Thawonmas, R., Ho, J.Y., and Matsumoto, Y. 2003) that uses action frequency analysis.

Etheredge et. al. (2013) proposed a method that combines clustering algorithms and HMMs. The clustering part is implemented on the Blindmaze game that they created through their study. The implementation of HMMs is provided as a future work for classification by labeling clusters beforehand.

In another study (Shin Jin Kang & Soo Kyun Kim, 2014), the authors presented an automated behavior analysis system using trajectory clustering and implemented it on one of the most successful massive multiplayer online game, World of Warcraft (WoW). They defined modes of players as socializing, exploring, in combat and idle just by using position and the camera angle of the player.

Harrison and Robert (Brent Harrison, David L. Roberts, 2011) proposed a model to predict future behaviours of players, using the previous behaviour data. The order in game actions, frequencies and correlations are used to develop a two-step probabilistic behaviour prediction model, and they tested their model on the World of Warcraft game.

As a different approach for player modeling, Drachen and Yannakakis used Self Organising Maps (SOM), on the commercial game Tomb Raider: Underworld data. Six fundamental features, death by opponent, by environment, by falling, total number of deaths, completion time and help on demand, are used for modeling high level player behaviours, and they obtained four clusters by unsupervised learning, and labeled them as Veterans, Solvers, Pacifists and Runners with the experts' opinions (Drachen, A. Canossa and G.N. Yannakakis, 2009).

3 PLAYER PROFILING IN GAMES

The research problem that we address is given as follows. a_1, a_2 ... ,a_T is the in-game action sequence of a player where each a_t ($0 \leq t \leq T$, T is the end of the game) consists of a feature set $\{f_1, \ldots, f_m\}$. Example features include *hit*, *beer*, *mine*, *death* and *kill* in our case. Given a training set of action sequences of different players, the problem is to distinguish new players competing with each other and determine their strategies. These action sequences must be considered temporally and noise in data must be handled. For this reason, temporal probabilistic models are suitable to model player patterns. According to the developed model, the player of a given game or it's strategies during the game can be classified. In this particular research, we focus on the classification of AI bots but the methods to be presented can be used to classify human players as well.

We address three main objectives as follows:

- To derive a model for every bot that the can be distinguished from each other.
- To group bots pursuing similar strategies.
- Determine different strategies used by a player during a given game episode.

4 A CASE STUDY ON THE VINDINIUM GAME

In the scope of our study, the Vindinium game (http://vindinium.org) is selected as the main platform where four artificial intelligence characters (AI bots) fight against each other. In Vindinium, competitor AI bots obtain mines that bring them gold each turn by killing mine goblins. Another way to gain gold is to kill enemy bots and take possession of their mines. Fighting against goblins and enemy bots decreases a certain amount of health and causes bleeding that lessens the health of a bot each turn. AI bots could use four taverns located on the map to increase their health by drinking beer. Game maps are generated randomly where the objects on each map are placed symmetrical, therefore, a fair competition is guaranteed. A typical game lasts about 1200 turns. In each turn, one of the bots is allowed to make an action, in total, each bot completes the game with 300 actions. An example scene from the Vindinium game is shown at Figure 1.

Completed games can be viewed from the Vindinium website by game ids, and game logs can be downloaded from the browser cache. Each game log consists of turn data, and each turn data contains bot positions, mine counts, gold info, health status of each bot, and also the current state of the map.

Game logs are transferred to the database system to be able to run queries to select appropriate data for our study. For example, interrupted or incomplete games can be eliminated by this way. Queries can also be made to select games of a certain hero or games that have a certain map size and contain a certain number of mines.

Figure 1: A screenshot from the Vindinium game. Bots names and scores are shown on the right side. The percentage distribution of the gained golds for each bot is shown on the colored bar. The bar on the bottom shows the progress of the game. Mines belonging to the bots can be seen on the tiled map with corresponding colors.

5 TEMPORAL-PROBABILISTIC PLAYER PROFILING

Hidden Markov Model (HMM) is selected as the main method to determine temporal relationships between states of the game probabilistically. The main reason HMM is selected as the main method for player profiling is temporal and sequence based action structure of Vindinium game. An HMM consists of an unobservable Markov chain and observable state probability transition definitions for each state in the chain. Observation o_t is produced by hidden s_t state at time t and holds the information to predict the next state which indicates that this process has Markov property meaning that an observation is only a function of the current state, and a state is only dependent on its previous state independent from the earlier states.

Figure 2 shows the basic components of an HMM. s_1, s_2, s_3 (s_i) represent hidden states. o_1, o_2, o_3 (o_j) represent the observations emitted by hidden states. π_1, π_2, π_3 ($\pi_i = P(s_i)$) are initial probabilities of hidden states. Observation probability in state s_i can be shown as $b_{ij} = b_i(o_m) = P(o_m \mid s_i)$, and the transition probabilities can be represented by $a_{ij} = P(s_i \mid s_j)$. In brief, an HMM is formed by three components $\lambda = \{A\ (a_{ij}), B\ (b_i\ (o_m)), \pi\ (\pi_i)\}$ (L. R. Rabiner, 1989) as A transition matrix, B emission matrix and π initial distribution matrix. An observation sequence to train HMM is shown as $q=q_1, q_2 \ldots, q_K$ and q_K is an element of observation set $\{o_1,\ldots,o_m\}$.

The learning phase of HMM uses Baum-Welch expectation maximization algorithm to find the best sets for state transition and emission probability matrixes (Luis Javier Rodriguez, Ines Torres, 2003). Local maximum likelihood estimate of HMM parameters are derived for given observation sequences.

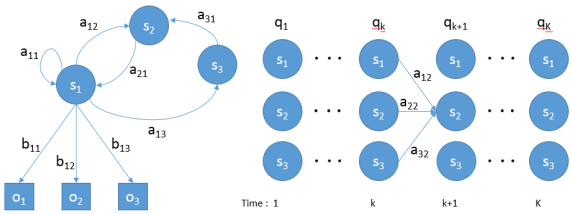

Figure 2: Representation of hidden state transitions and observation emissions.

5.1 HMM-based Bot Modeling in the Vindinium Game Environment

All basic bot actions can be used as observable states, and combinations of these actions form hidden states. The basic actions of bots are as follows:

hit: This action is observed when two bots are located in adjacent tiles on the game map. This results in reduced energy of the bots by a certain amount.

beer: This action is observed when a bot is adjacent to a tavern and chooses to stay on the same tile for a turn facing to the tavern.

mine: This action is observed when a bot is adjacent to a gold mine and chooses to stay on the same tile for a turn facing to that mine. As the result, the mine is acquired but the bot's energy is reduced by a certain amount.

death: This action is observed when the health of the bot is depleted to zero. Only fighting with the other AI bots could be the cause of death.

kill: This action is observed after a fight that results in the death of the enemy.

move: This action is observed when the player goes from one tile to an adjacent tile on the walkable areas of the map.

Likewise hidden states are explained below:

Fight Mode: This hidden mode indicates that the bot is in a war situation with the other bots. Accordingly, the bot executes one of the actions *hit*,

death or *kill* with a high probability, and it can be said that *beer* action is carried out rarely. The bots that are following a strategy to win the game by taking mines of enemy bots through killing them are most likely to be in Fight Mode.

Mine Mode: This hidden mode indicates that the bot has a tendency to collect mines one after another. Accordingly, the bot executes action *mine* with a high probability and one of the actions *beer*, *hit*, *kill* or *death* with lower probabilities. The bots that are following a strategy to win the game by holding as many mines as possible are most likely to be in Mine Mode.

Survive Mode: This hidden mode indicates that survival and keeping the possessed mines safe are considered at the highest priority to win the game by the bot. Accordingly, the bot executes action *beer* with a high probability and one of the actions *mine*, *hit*, *kill* or *death* with lower probabilities.

Move action is considered as a means to perform other actions in the game. For that reason, it is omitted from the formation of the observation history. The relationships between hidden and observable states ("B" emission matrix) are initially given as a matrix shown in Table 1. To construct HMM, "A" state transition and "P" initial matrix created with equal values. During training, the values of this matrix are evolved to learn a corresponding profile.

Table 1: "B" emission matrix explaining the relations between observations and modes.

%	Hit	Beer	Mine	Death	Kill
Fight	65	15	0	10	10
Mine	5	20	70	2,5	2,5
Survive	15	50	20	7,5	7,5

Another approach in this study is constructing an HMM by assigning a hidden state (such as HM-1, HM-2, HM-3, HM-4 and HM-5) for each observable state and assigning nearly probability 1 to the relation in the emission matrix as shown in Table 2. The values in this matrix are rearranged by the training process according to emission distribution probabilities and observation sequences acquired from the game data.

The purpose of HMM-based player profiling is to determine the relationship between actions and hidden states as behaviour modes in bots action sequences. By this way, a well trained HMM is used to recognize bots from their strategies, and classification of them is also possible.

Table 2: Alternative "B" emission matrix. Observations are shown in columns and the hidden states are in rows.

Hidden Mode	Hit %	Beer %	Mine %	Death %	Kill %
HM-1	100	0	0	0	0
HM-2	0	100	0	0	0
HM-3	0	0	100	0	0
HM-4	0	0	0	100	0
HM-5	0	0	0	0	100

5.2 Selection of Training and Test Matches

AI bots are able to compete in various maps with different sizes and different mine counts. The effect of these parameters on the behavior of the players is very high. For example, bots face each other and fight very often in small sized game maps in contrary, in large maps their meeting probabilities decrease. The total mine count of the map also effects the strategies of bots. Games are selected by taking these parameters into account.

5.3 HMM Training Process and the Sliding Window Method

Observation sequence of a selected game belonging to a bot is given to the initial HMM model and a hidden state sequence is obtained from the model. HMM matrices are updated using this sequence via the Baum-Welch algorithm. For each training game, a new HMM is constructed described above. To finish a training step, the average of HMM matrices are calculated, and this model is used to set initial values for the next training step. This process is illustrated in Figure 3.

In this work, we propose to use a sliding window approach for the consideration of actions of the player. Instead of taking individual actions as observations, action segments are considered as observation data. The sliding window method is illustrated in Figure 4 allows us to take each single observation into account considering previous and subsequent actions as episodes of game parts. Proposed method has used to visualise HMM results for grouping of bots and process monitoring which explained in detail in section 6.

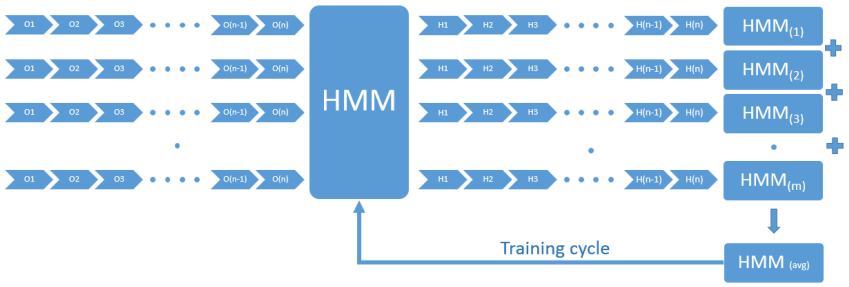

Figure 3: HMM training process.

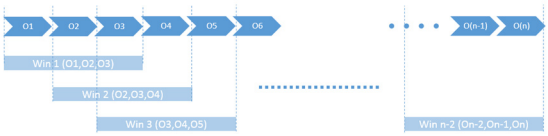

Figure 4: The observation sequence of a game is divided into windows with size 3. Each observation is taken into account with one previous and one subsequent action.

6 EXPERIMENTS

The HMM-based profiling method is evaluated in three different experiment sets. In these experiments, recognition of a certain bot, bot clustering and game strategy recognition performances are measured.

6.1 Experiment 1: Recognition of Bots

For classification of bots, 12 bots are selected as training data with over 100 games satisfying the completed game criteria. Final HMM's are created for each bot as shown in Figure 3 above.

Then, observation sequence of a test match is given to the trained HMMs of selected bots, and log probabilities are calculated via the Viterbi algorithm. The test match is considered to belong to the bot associated with the HMM model that generates the highest probability value. For all the test matches belonging to a bot, the log probability results are obtained. If the HMM belonging to the test bot gives the maximum log probability, it is counted as a true match. By this way, the percentages of recognition accuracy are calculated.

6.1.1 Selection of Test Groups

Recognition performance of each trained HMM for each bot is tested in groups. Group sizes are selected as 2, 4, 8 and 12. For each group size, all possible combinations of bots are constructed. All test matches of a bot in a group are given to the HMM models of other bots in the group, and recognition percentages are calculated. After calculation of results is completed for each group size, recognition performances are set as the average of recognition percentages for each bot.

6.1.2 Results

Recognition results for groups with sizes 2, 4, 8 and 12 are given in Table 3 are obtained by using the emission matrix values for HMM shown in Table 1. Emission matrix values given in Table 2 produce the results given in Table 4. Bots named "miner", "fighter" and "mybot" are programmed for test purposes in the process of this study. The rest of the 9 bots are selected using the criteria described above. "miner" bot tries to maximize its gold count by taking mines killing guardian goblins. "fighter" bot often kills other bots and possesses its mines by the rules to win the game. "mybot" tries to be alive mostly and takes mines or fights with other bots according to its situation. The main reason for high recognition rates of these bots is that their strategies are so simple, definitive and fixed.

Considering the results shown in Table 4, it was observed that recognition performances are increased compared to Table 3. It can be said that the main reason of this improvement is the increasing number of hidden states. Here we assumed that there is a hidden state for each observable state for HMM and have one to one relation. Emission matrix values are given accordingly as 99,99%. These values are adjusted in the process of training of HMM considering observation sequences of training games.

Table 3: Recognition results for groups sizes of 2, 4, 8 and 12. The emission matrix whose values are given in Table 1 is used.

Group size 2		Group size 4		Group size 8		Group size 12	
Bot	% Rec. rate	Bot	% Rec. rate	Bot	% Rec. rate	Bot	% Rec. rate
miner	100,00	miner	100,00	miner	100,00	miner	100
mybot	100,00	mybot	100,00	mybot	100,00	mybot	100
fighter	96,36	68dfq9s6	94,55	68dfq9s6	100,00	68dfq9s6	90
68dfq9s6	95,91	fighter	83,64	fighter	74,55	ar5xzyt6	65
ar5xzyt6	86,36	ar5xzyt6	81,94	ar5xzyt6	74,33	fighter	60
jjqna34p	80,91	jjqna34p	62,97	i4v2093x	39,73	6n638ovy	60
6n638ovy	80,91	scxsye1b	62,61	6n638ovy	32,12	i4v2093x	25
scxsye1b	79,55	6k31znk9	56,24	6k31znk9	30,33	iyo6u22k	25
i4v2093x	79,55	6n638ovy	50,06	jjqna34p	25,61	scxsye1b	10
p7bk68fw	75,91	p7bk68fw	45,73	iyo6u22k	23,00	jjqna34p	5
iyo6u22k	73,64	i4v2093x	43,24	scxsye1b	14,39	6k31znk9	5
6k31znk9	71,36	iyo6u22k	39,36	p7bk68fw	4,32	p7bk68fw	0
AVERAGE	**85,04**	**AVERAGE**	**68,36**	**AVERAGE**	**51,53**	**AVERAGE**	**45,42**

Table 4: Recognition results for groups sizes of 2, 4, 8 and 12. The emission matrix whose values are given in Table 2 is used.

Group size 2		Group size 4		Group size 8		Group size 12	
Bot	% Rec. rate	Bot	% Rec. rate	Bot	% Rec. rate	Bot	% Rec. rate
mybot	100,00	mybot	100,00	mybot	100,00	mybot	100
68dfq9s6	99,55	68dfq9s6	98,64	68dfq9s6	96,82	68dfq9s6	95
fighter	98,18	fighter	94,55	fighter	87,27	miner	80
miner	96,36	miner	90,18	miner	82,18	fighter	80
scxsye1b	93,64	scxsye1b	81,45	6n638ovy	63,62	ar5xzyt6	55
6n638ovy	91,82	6n638ovy	79,09	ar5xzyt6	62,67	6n638ovy	55
ar5xzyt6	91,36	ar5xzyt6	78,21	scxsye1b	59,27	scxsye1b	40
jjqna34p	89,09	jjqna34p	75,36	jjqna34p	53,00	p7bk68fw	40
i4v2093x	86,82	i4v2093x	66,18	p7bk68fw	46,50	jjqna34p	35
p7bk68fw	82,27	p7bk68fw	63,97	i4v2093x	41,24	i4v2093x	30
6k31znk9	81,36	6k31znk9	57,73	6k31znk9	37,50	6k31znk9	30
iyo6u22k	70,00	iyo6u22k	34,73	iyo6u22k	10,83	iyo6u22k	5
AVERAGE	**90,04**	**AVERAGE**	**76,67**	**AVERAGE**	**61,74**	**AVERAGE**	**53,75**

According to results shown in Table 3, recognition performance of our model evaluated as 85,04% which is 1,7 times better than random recognition performance 50% for groups with size 2. For groups with size 4, recognition performance increases from 25% to 68,36% which is 2,73 times better. Likewise, groups with size 8 recognition lift factor is 4,12 and for groups with size 12 it is calculated as 5,45. Recognition performance and lift factors are given in Figure 5 and Figure 6, respectively. Figure 7 and Figure 8 show the same results obtained by using performance values taken from Table 4.

6.2 Experiment 2: Grouping of Bots

Even though AI bots in the Vindinium game are written by different people, they may have very similar strategies. Considering the responses as log probability values obtained from HMM models for a given observation sequence, it could be shown that for some bots the values are very close to each other. If the games belonging to a bot are examined through the eyes of an expert, it could be said that certain bots adopted nearly same strategies.

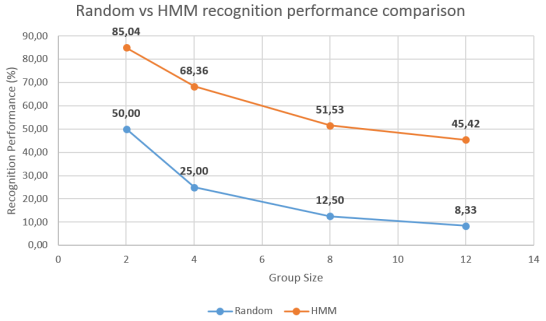

Figure 5: HMM and random recognition performance results for groups with sizes 2,4,8 and 12 according to results shown in Table 3.

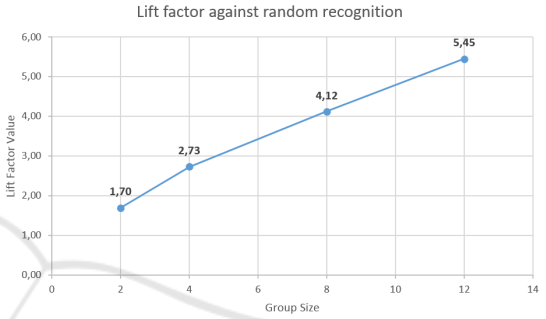

Figure 6: Lift factor values against random recognition performance according to results shown in Table 3.

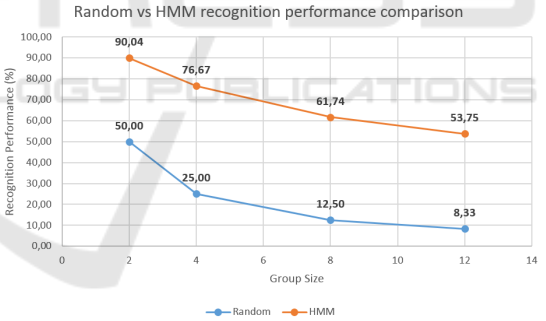

Figure 7: HMM and random recognition performance results for groups with sizes 2,4,8 and 12 according to results shown in Table 4.

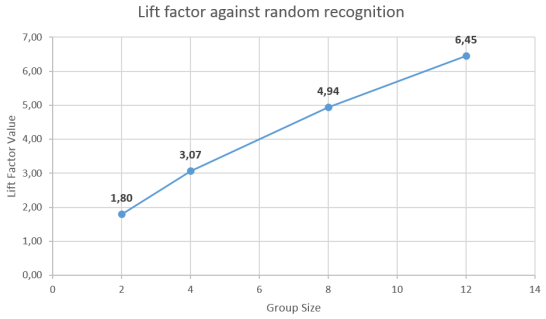

Figure 8: Lift factor values against random recognition performance according to results shown in Table 4.

For a detailed examination of strategies of these bots through the game, observation sequences belonging to their specific matches are divided into certain sized windows. Window sequences are given the HMMs of bots, and log probabilities are calculated for each of them.

In order to show that the window model could be used to detect similarities between bots with similar strategies for a given test match, four bots are selected two of each have similar gameplay strategies.

Log probability results obtained from a test match using HMM's of the selected bots for each window are given in Figure 9 and Figure 10. In this test case, the window size is selected as 10. x-axis shows the sequence number of windows, and y-axis shows the log probability values calculated using HMM's of each selected bot for given test match. As it can be seen in the figures, two of the bots have very similar log probability responses for each window. Human observers also state that these players have similar strategies analyzing the previous games of the selected bots.

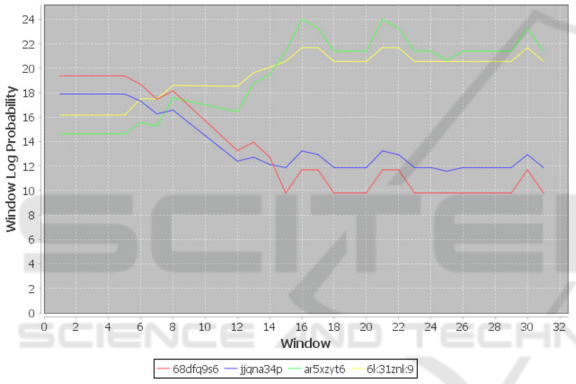

Figure 9: Observation sequence of the test game (Id:uxuykwpv) is given to the HMMs of 4 bots and window log probability results are shown in y-axis (This game can be viewed at: http://vindinium.org/uxuykwpv).

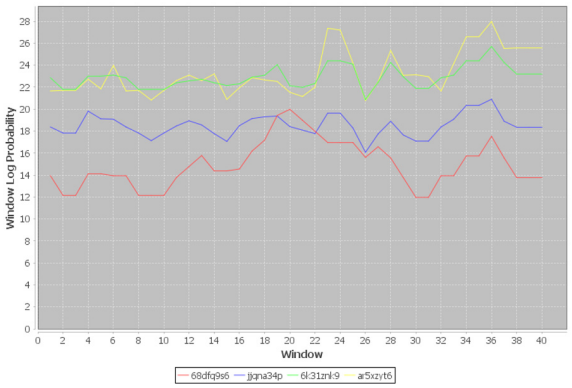

Figure 10: Observation sequence of the test game (Id: hiwqdgec) is given to the HMMs of 4 bots and window log probability results are shown in y axis (This game can be viewed at: http://vindinium.org/hiwqdgec).

6.3 Experiment 3: Process Monitoring

A single AI bot may follow different strategies in different games. It can even change its strategy during the same game in different time frames depending on the situation. To observe these strategy changes, the trained HMM of selected bots can be used in conjunction with the proposed window model. In Vindinium, three basic strategies can be adopted to win the game based on our experiences. The first one is to kill other bots and acquire their mines to collect more gold which we label as Fighter, the second strategy is to get as many as mines by killing goblin guardians and have the highest mine count through the game that could be labeled as Miner, the third one is trying not to die and hold a certain number of mines which could be characterized as Survivor.

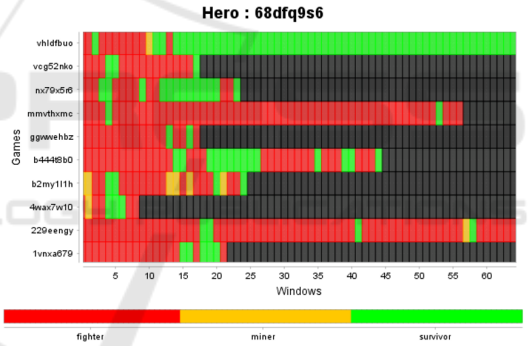

Figure 11: Strategy changes of a selected bot for 10 random games are shown with different colors. X-axis shows the window sequence numbers. Y-axis shows game ids. Strategies are colored as it is shown in the legend.

To represent the strategies described above, we generated the observation sequences for each of them with a human observer's (expert) opinion considering the possible actions and action sequences that could arise by the bots following these strategies. Using these observation sequences, HMMs are generated for each strategy. Using these HMMs that represent our three basic strategies, for each window of action sequence of a test match belonging to a bot we could assign a strategy for a specific window.

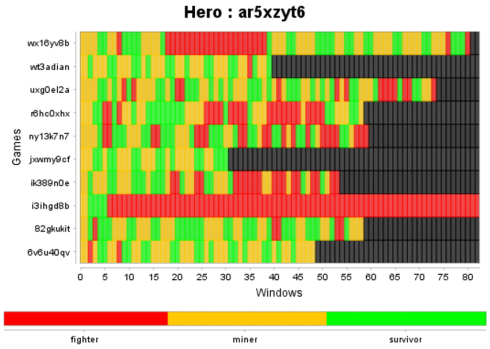

Figure 12: Strategy change of the selected bot for 10 random games is shown. X-axis shows the window sequence numbers. Y-axis shows game ids. Strategies are colored as shown in the legend. Games can be viewed at the corresponding http://vindinium.org/<game_id> link.

7 CONCLUSIONS

We have presented our HMM-based player profiling method supported with the sliding window approach. Hidden state representation which associated one to one observable state mapping for initial HMM emission matrix is another improvement over standard HMM. We selected the Vindinium game environment as our testbed. To the best of our knowledge, this is the first time Vindinium AI contest data are considered for bot player modeling. The method has been evaluated for three different performance criteria: recognition, grouping and strategy monitoring of AI bots. The results indicate that the method can identify bots with average success rate of 90,04% for groups with size 2, 76,67% for groups with size 4, 61,74% for groups with size 8 and 53,75% for groups with size 12. The method could also be used for grouping of the bots and it can determine strategy changes of bots during a game. This work forms a promising basis for our future studies on human player profiling. The method with its generic representation could be easily implemented for other games.

ACKNOWLEDGEMENTS

The authors wish to thank the developers and the community of the Vindinium AI contest which provided a rich content test-bed for the research presented in this paper. This work is supported by a project supported by the Scientific and Technological Research Council of Turkey (TUBITAK).

REFERENCES

Brent Harrison, David L. Roberts, (2011). Using sequential observations to model and predict player behavior. Proceedings of the 6th International Conference on Foundations of Digital Games, p.91-98, Bordeaux, France.

D. Kennerly, (2003). Better game design through data mining. Gamasutra.

Drachen, A. Canossa and G.N. Yannakakis, (2009). Player Modeling Using Self-Organization in Tomb Raider: Underworld. Proc. IEEE Symp. Computational Intelligence and Games, pp. 1-8.

Etheredge M., Lopes R. and Bidarra R. (2013). A Generic Method for Classification of Player Behavior. AAIDE - Artificial Intelligence in the Game Design Process.

K.S.Y. Chiu, and K.C.C. Chan, (2008). *Game engine design using data mining*. Proceedings of the 26th IASTED International Conference on Artificial Intelligence and Applications, pp. 352-357.

L. R. Rabiner, (1989). A Tutorial on Hidden Markov Models and SelectedApplications in Speech Recognition. Proc. IEEE, Vol.77 (2), pp. 257-285.

Luis Javier Rodriguez, Ines Torres (2003). Comparative Study of the Baum-Welch and Viterbi Training Algorithms Applied to Read and Spontaneous Speech Recognition.

S. C. Bakkes, P. H. Spronck, and G. van Lankveld, (2012). Player Behavioural Modelling for Video Games. Entertainment Computing, in press:1--9.

Shin Jin Kang & Soo Kyun Kim, (2014). Automated spatio-temporal analysis techniques for game environment. Springer Science+Business Media New York.

Thawonmas, R., Ho, J.Y., and Matsumoto, Y. (2003). Identification of Player Types in Massively Multiplayer Online Games. Proc. the 34th Annual conference of International Simulation and Gaming Association (ISAGA2003), Chiba, Japan, pp. 893-900.

Yannakakis, G. N.; Spronck, P.; Loiacono, D.; and Andre, E. (2013). Player Modeling. Dagstuhl Seminar on Game Artificial and Computational Intelligence.

Yoshitaka Matsumoto, Ruck Thawonmas, (2004). MMOG Player Classification Using Hidden Markov Models. ICEC-Entertainment Computing.