Play-style Identification through Deep Unsupervised Clustering of Trajectories

Branden Ingram, Benjamin Rosman, Clint van Alten, Richard Klein
School of Computer Science and Applied Mathematics
University of the Witwatersrand
Johannesburg, South Africa
{branden.ingram, benjamin.rosman1, clint.vanalten, richard.klein}@wits.ac.za

***Abstract*—In any game, play-style is a concept that describes the technique and strategy employed by a player to achieve a goal. Being able to identify the play-style of a player is desirable as it can enlighten players on which approaches work better or worse in different scenarios, as well as inform developers of the value of design decisions. In this paper, we propose a novel approach to play-style identification based on an unsupervised LSTM-autoencoder clustering approach for multi-dimensional trajectory-based data of variable length. We evaluate our approach on two domains and show that not only is our model capable of identifying these play-styles from entire trajectories but it is also capable of this during gameplay from partial trajectories. Additionally, it is demonstrated through state frequency analysis that the properties of each of the play-styles can be identified and compared. Through these processes, we can extract useful information which describes the different behaviours or play-styles present within a domain useful to both players and developers.**

***Index Terms*—play-style identification, player modelling**

I. Introduction

A play-style is defined as a particular way of playing a game and, typically, reflects the preferences with which a player may engage in progressing through the game. For example, a player may have a preference for exploring the minutiae of a game, or for completing it as quickly as possible. The diversity of possible play-styles can be significant, meaning that different players could play a game in very different ways. There is much potential benefit in identifying the play-style of an individual player as they engage in the game [1]. From the player's perspective, identifying their style could be used to assist in the tailoring of game mechanics in real-time for the needs and preferences of the player. Additionally, designers gain insight into how their players are interacting with gameplay features and mechanics. There is an added benefit of being able to tailor tutorial mechanics and in-game tips to the type of player identified.

Multiple studies have looked into modelling the style in which an individual engages with a problem and is not a concept unique to games. For example, in learning, studies have been conducted to identify learning styles [2]. Similar studies, specifically in games, have looked to identify player archetypes as well as personality characteristics and motivations [3], [4], [5]. Although work has been done in terms of play-style identification, there has been little exploration of applying these techniques to complex trajectory-based data. Without considering the temporal dimension of a user's playthrough we lose the ability to understand how the player arrived at a certain state and instead the focus is purely on that final state. By doing so we could reveal further insights lost when only utilising high-level features. This approach has the added value of allowing us to assign labels at different times along the trajectory that more accurately represent the individual. In addition, video games by nature are time series domains with decision making occurring and changing through the course of gameplay. Therefore, it is fitting to model play-styles using temporal models.

One possible reason for a limited amount of existing work focusing on temporal data could be the lack of this kind of trajectory data labelled with respect to style. This makes applying supervised learning approaches difficult and as a result, we developed and evaluated our unsupervised model on a generated dataset where ground truths were known. The goal of this approach is to improve the credibility of the results obtained from natural datasets where play-styles are unknown.

Limited access to data is not the only issue in trajectory analysis that requires further investigation. Another issue is how to preprocess and analyse large quantities of multidimensional trajectories of variable length. Non-machine learning applications of trajectory analysis have traditionally only been performed on points moving in 1D, 2D or 3D spaces [6]. Video game state information, however, is not limited to such low dimensions. In machine learning, the development of Recurrent Neural Networks has allowed for solving complex problems using sequence-based data. Most notable applications are in video tagging [7], generating image descriptions [8], speech modelling [9], language modelling [10] as well as video-games [11]. These architectures allow us to be able to handle high-dimensional trajectory data.

Interpreting generalised behaviours from large datasets is another issue when trying to identify play-style. This has been done with non-trajectory based data (summary statistics) within video games to describe player roles [12]. Creating such a technique for trajectory data requires compressing both the spatial and temporal distributions. Such descriptions have recently been obtained by utilising clustering techniques [13], [14]. These descriptions offer useful insight into characteristics and trends present in the underlying data, however, they are yet to be applied to video game domains as an informative tool for both player and designer.

In this paper, we address both the problem of processing complex data and that of finding interpretive descriptions of behaviours within a dataset. These descriptions should look to describe some general characteristics or patterns exhibited by a group. To that end, we propose a novel system for play-style identification through the clustering of multi-dimensional variable-length trajectories in video games and demonstrate that these clusters represent varying styles of behaviours. The core of this system is a specialised LSTM-autoencoder that utilises the benefits of Recurrent Neural Network [15] architectures to handle sequence-based data. We evaluate this model on a generated benchmark dataset as well as a natural domain. We also demonstrate the unique ability of our model to identify the style in both complete and partial trajectories without the need for any additional engineering or training time. Lastly, the characteristics of each play-style are recovered through the analysis of each cluster and their relationships to each other.

II. Related Work

Traditionally, play-style identification has been approached through player modelling, which is the study of computational models of players in games. This includes the detection, modelling and prediction of human player traits which are manifested through cognitive, affective and behavioural patterns [16]. The techniques utilised in player modelling usually fall into one of two groups: model-based or model-free.

A. Model-Based Approaches

In model-based approaches, a player model is built on a theoretical framework whereby the preexisting understanding of the domain is leveraged [16]. Model-based approaches have been inspired by cognitive frameworks [17] as well as general theoretical frameworks of behavioural analysis such as usability theory [18]. Additional examples are models which utilise theories of "fun" [19], [20]. These top-down approaches have also been used to dynamically affect player experience [21], [22], [23]. Although model-based techniques like these are useful, our work seeks to avoid the use of any prior knowledge of the domain and in essence, learn the heuristic classifier in a model-free setting. Doing so has two benefits, with the first being we are not required to impart any domain-based bias, and the second is the ability to learn a more accurate model than a handcrafted one.

B. Model-Free Approaches

Model-free approaches refer to the construction of a mapping (model) between (player) input and a player state representation. In this case, there is no pre-existing understanding of the model; rather, it is learned through an iterative process [16]. To achieve this, observations are collected and analysed to generate a model without strong initial assumptions of its structure. In model-free approaches we see attempts to model and predict player actions and intentions [24], [25]. Thue et al. [24] implemented a system that learns a label for the player representing their style. However, this required the manual annotation of the different "encounters" with their corresponding style. Our proposed approach looks to learn these play styles in a completely unsupervised setting.

Data mining efforts to identify different behavioural playing patterns within a game have also been implemented using bottom-up approaches [5], [26]. Drachen, Canossa and Yannakakis [5] trained emergent self-organising maps on high-level game behaviour data to classify players into four categories representative of their style. Here no pre-existing information was required for the analysis of the data, which also meant that the clusters that were identified were not influenced by external factors. However, this process required identifying the characteristics that resided within the data of each separated cluster. It is this statistical analysis of the separated data that revealed the semantics behind each cluster and allowed for the identification of player archetypes. We perform a similar step, however, the results of this process are dependent on the features present in the original dataset. These techniques have been effective, however, these studies have utilised meta-data summaries over raw user data experiences. We aim to achieve the same goal by analysing raw sequence-based trajectory data, but without the need for extensive feature engineering. This analysis is based on trajectory clustering.

C. Trajectory Clustering

An approach to the analysis of trajectory data is comparing and grouping based on whole or partial trajectory attributes using a similarity measurement. However, it has been demonstrated that clustering using sub-sequences lacked meaning as the generated cluster centres are not dependent on the input [27]. Additionally, when clustering trajectories, the choice of similarity measure is important as it should consider both the spatial and temporal features [28]. Common distance measures include Hausdorff distance, Dynamic Time Warping (DTW) [29], Euclidean distance and Longest Common Sub-Sequence (LCSS) [30]. The choice of metric is dependent on the structure of the data. DTW and LCSS in particular can measure similarities between varying length trajectories.

Techniques such as k-Means and hierarchical clustering have been utilised to perform the clustering step [31]. However, these techniques tend to be ineffective when input dimensions are high [32]. Additionally, model-based methods which use statistical approaches (COBWEB [33]), Neural Network approaches (ART [34]), or self-organising maps (SOM [35]) have been utilised. SOM clustering of time-series features is unsuitable for trajectories of unequal length, as the dimensions of the weight vectors are fixed [36]. Furthermore, data compression techniques utilising Latent Dirichlet Allocation (LDA) have been applied to play-style clustering [37]. However, by considering timesteps as individual data points, LDA approaches ignore the structural significance of temporal data.

In addition to these, deep learning techniques have been applied to unsupervised clustering [13]. Xie, Girshick, and Farhadi [13] used an autoencoder which allowed for the important data compression aspect which was required to make the previously mentioned clustering techniques more feasible. This makes the clustering task easier since clustering can be performed on the encoded data. Xie, Girshick, and Farhadi [13] pre-train a network to minimise the reconstruction loss and then separately minimise the clustering loss. This separate clustering step used the KL-Divergence [38] between a target distribution and an estimated distribution generated from the encoded data. More recently, LSTM-autoencoders have been implemented which are capable of handling time-series data effectively [39]. This is a similar approach to Xie, Girshick, and Farhadi [13] with the added LSTM layers to handle time-series data as well as joint minimisation of the reconstruction and clustering aspects. We utilise a similar approach to handling our video game-based trajectories which have the added characteristic of being variable length as well as being higher dimensional.

III. Methodology

We aim to solve the following problem: given a set of trajectories (X) can we identify a label (y) that categorises each trajectory ($x_i \in X$) according to a unique style of play.

A. Play-style Identification

Our unsupervised approach is based on two key steps. First, we utilise an autoencoder network to project a trajectory into a lower-dimensional latent representation. We then perform clustering on this latent space to discover clusters corresponding to related trajectories in this space. The full system is depicted in Fig. 1.

1) Trajectory Encoding: An autoencoder works to reconstruct each original input trajectory ($x_i \in X$) after first encoding it as a lower-dimensional state ($z_i \in Z$). Formally, this is given by (1).

$$z_i = \texttt{Encoder}(x_i) \texttt{ and } x_i' = \texttt{Decoder}(z_i) \tag{1}$$

Our specific model is a temporal autoencoder containing non-stacked LSTM layers similar to Xie, Girshick, and Farhadi [13]. This allows the processing of varied length trajectories by feeding the state at each time-step into its own LSTM cell. These cells ("A" in Fig. 1), pass on the important information (h_t, c_t) in sequence until finally outputting a fixed sized vector representative of our latent space (Z). The network is trained using back-propagation through time [15].

2) Trajectory Clustering: Having projected the trajectories into the latent space, we then cluster them. This clustering step is performed on the set of all generated pairs (x, z), where $x \in X$ and z is the output of the Encoder in (1). Each pair (x, z) is clustered with respect to z to form predicted labels y'. Since z_i is a representation of x_i we can use y_i' as the cluster label for the original data. Clustering using Z enables the use of most clustering algorithms as there is no longer an issue of varying length or temporal features.

IV. Experiments

A. Datasets

To validate the robustness of our method, we evaluate our model on two different datasets. The first is derived from a

Algorithm 1 Preference-Based Trajectory Generation (PBTG)

```
 1: procedure PBTG(Environment $E$)
 2:     Define a set of reward functions $R$
 3:     Initialise our set of trajectories $T = \{\}$
 4:     for all reward functions $r \in R$ do
 5:         Use Q-learning to learn optimal policy $\pi_r^*$
 6:         for $n$ number of required Trajectories do
 7:             $\pi_r^n \leftarrow perturb(\pi_r^*)$
 8:             Generate $t(n, r)$ from $\pi_r^n$ and append to $T$
 9:         end for
10:     end for
11: end procedure
```

grid-world game where a player seeks out a goal with the opportunity of completing two additional optional objectives. By generating this set of trajectories we have access to the ground truth play-styles and as a result, we use this domain to obtain quantifiable measure of performance. The second is an unlabelled set of trajectories from the game Super Mario Bros [40]. This data set was collected from individuals and we use it to demonstrate our model's performance in natural domains.

1) Grid World: To evaluate an algorithm for play-style identification, it is important to have multiple trajectories from a set of different play-styles. Trajectory based datasets labelled according to style do not exist and therefore we generate data to account for this. We distinguish two play-styles as being different goals that could be reached by an agent. These we model as different reward functions in the reinforcement learning paradigm. This idea of reward shaping has been used to train a set of human-like bots with differing styles [41]. We utilise this approach to generate a set of trajectories with differing performance levels for multiple styles.

Our PBTG trajectory generation approach (Algorithm 1) was used to generate 5 individual datasets T_n from 5 different environments ($E_1, \ldots, E_5$) with 4 varying play styles (reward functions) present. Each environment is a 10$\times$10 grid world, as depicted in Fig. 2. The environments each have a start state (S, in blue) and a goal state (G, in green). Walls (black tiles) cannot be traversed and trap states (red tiles) result in failure. The variety in play-styles is introduced through the addition of two bonus states (B_1, in gold and B_2, in cyan). These are the optional objectives that a player with certain preferences might wish to complete. The set of actions is the movement in any of the 4 primary cardinal directions. Using this design we generated data with 4 play-styles, as described in Table I. The set of reward functions R used to emulate these behaviours is also defined in Table I. Here we defined large positive rewards for the objectives we wished the agent to accomplish. The respective bonus rewards were only given the first time an agent reached either B_1 or B_2. Following the procedure outlined in Algorithm 1 we trained an agent for each of the combinations of R and E for 20000 episodes with discount factor $\gamma = 0.99$ and linear ϵ decay in order to ensure our agent converges to the global optimal. The state is given

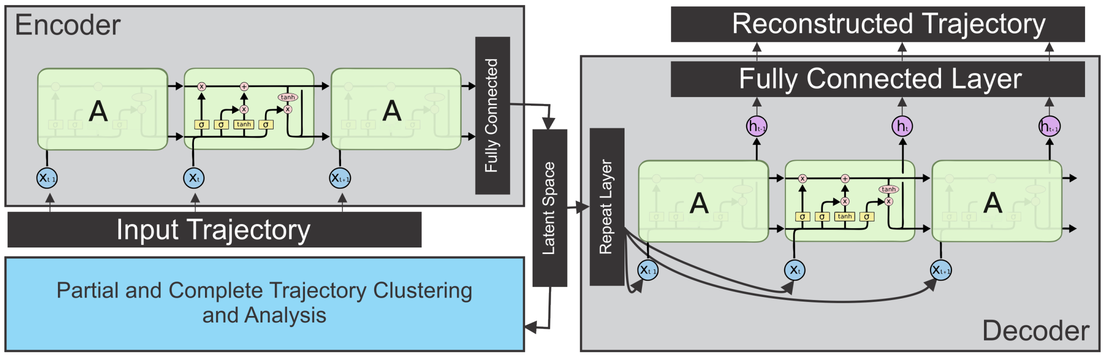

Fig. 1. Proposed architecture

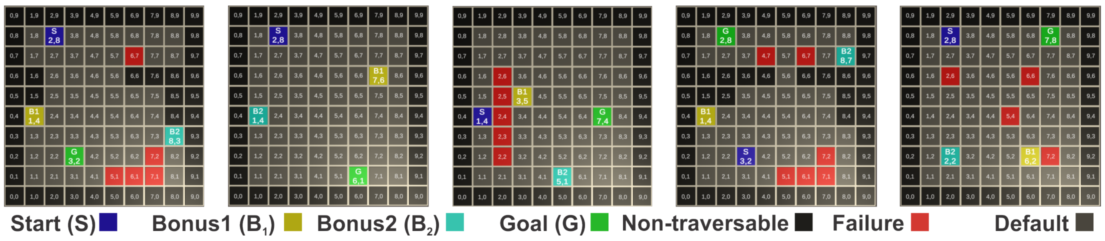

Fig. 2. Randomly generated grid world environments $E_1, ..., E_5$

TABLE I
OBSERVABLE PLAY-STYLES AND REWARD STRUCTURE

R	Behaviour	G reward	B_1 reward	B_2 reward
1	Moves directly to G	100	0	0
2	Visits B_1 before G	100	50	0
3	Visits B_2 before G	100	0	50
4	Visits B_1 and B_2 before G	100	50	50

by the tuple (x, y, b_1, b_2) where x and y are the Cartesian grid coordinates and b_1 and b_2 indicate whether an agent has visited B_1 or B_2, respectively. Our dataset consists of 8000 randomly selected trajectories per R and E. Therefore our trajectory is a sequence of time steps in the form of (x, y, b_1, b_2).

2) Mario: This dataset consists of 74 playthroughs across 11 different levels of Super Mario Bros. These playthroughs were each captured by logging the actions of a unique human participant [40]. We then refactored this data into a trajectory, where each time step represents the current state of the playthrough at that point. We defined a state as a tuple given by (j, k, r, c, d, e) where j is the number of jumps, k is the number of enemies killed, r number of times the player has started running, c the number of coins collected, d the number of times the player died and e the unique encoding for each action.

B. Training

We trained an individual model for each of our 6 environments $(E_1, \ldots, E_5, \texttt{Mario})$. To avoid dataset-specific tuning, we used the same parameters across all domains with ReLU as the activation function. Dimensions $(20, 8)$ were used to define the (Hidden, Output) sizes of the autoencoder. The output dimension is the size of our latent space. The models were trained for 10000 episodes using the Adam optimiser with a learning rate 0.001 using (2) to calculate the loss.

$$\frac{dL_{\texttt{Recon}}}{dz} = \frac{d(\frac{1}{2}||(x - x')||_2^2)}{dz} \tag{2}$$

C. Clustering

For the clustering step, we evaluated both k-means and Gaussian Mixture Models (GMMs). For both algorithms, the number of clusters were 4 and 8 for the Grid World and Mario domains, respectively. For k-means and GMM we fit our data using 100 restarts and a maximum iteration of 10000 . In addition, k-means used a tolerance of 0.0001 and GMM used a “full” covariance type. Although our model is completely unsupervised, we do compare the ground truth cluster labels (y_i) with the predicted labels (y_i') to validate accuracy. This validation step uses (3) and (5). Firstly (3) finds the best match between the cluster assignments from an unsupervised algorithm (y_i') and a ground truth assignment (y_i), where m ranges across all possible one-to-one mappings and n is the number of data points [42].

$$\texttt{Accuracy} = \texttt{max}_m \left(\frac{\sum_{i=1}^{n} 1\{y_i = m(y_i')\}}{n} \right) \tag{3}$$

Secondly, (5) defines a confidence measure $(\overline{k})$ giving the probability vector that a given trajectory x belongs to a particular cluster. This is determined using (4) which calculates the Euclidean distance vector $\overline{d}$ between x and C, where C is the set of centroids determined through the clustering step.

$$\overline{d}(x) = ||(C - \texttt{Encoder}(x)||_2^2 \tag{4}$$

$$\overline{k}(x) = \frac{exp(\overline{d}^{-1}(x))}{\sum exp(\overline{d}^{-1}(x))} \tag{5}$$

Additionally since identifying play-styles on partial trajectories is a key feature, we need to perform a similar evaluation step on these partial trajectories. To this end, we calculate the total accuracy using Algorithm 2 as the average correctly labelled predictions for every trajectory $(t \in T_n)$. In this case, our predicted label (y') is determined by first calculating the weighted moving average confidence ($\texttt{WMAC}$) [43] over all partial trajectories $(s \in t)$ using (6). Here $\overline{k}_i$ represents the confidence calculated using (4) and (5) as $\overline{k}(s)$ where $s \leftarrow t[0 : i]$ and the weighting $w \leftarrow \frac{m(m+1)}{2}$ where m is the length of t. The cluster with the highest confidence is then selected as our predicted label and compared to the corresponding ground truth label y. This process is then repeated for all $(t \in T_n)$ with our final partial trajectory accuracy (PTA) being the percentage of correct predictions.

$$\overline{\texttt{WMAC}} = \sum_{i=0}^{m} \left(\frac{\overline{k}_i \times i}{w}\right) \tag{6}$$

Algorithm 2 Partial Trajectory Accuracy (PTA)

```
1:  procedure PTA($T_n$)
2:      for all Trajectories $t \in T_n$ do
3:          $w \leftarrow \frac{(m)(m+1)}{2}$                ▷ $m$ is the length of $t$
4:          Calculate $\overline{\texttt{WMAC}}$ with all partial trajectories $s \in t_m$
    using (6)
5:          $y' \leftarrow \operatorname{argmax}_{1 \le j \le N} \overline{\texttt{WMAC}}$        ▷ $N$ being the number of
    clusters
6:          if $y'$ matches ground truth $y$ then
7:              $PTA \leftarrow PTA + 1$
8:          end if
9:      end for
10:     $PTA \leftarrow \frac{PTA}{n}$
11: end procedure
```

V. Results and Discussion

Through the results of our experimental analysis, we demonstrate the ability of our model to accurately cluster game trajectories into their respective play-styles on both complete and partial trajectories. Additionally, we also show how unique characteristics can be recovered from these clusters by identifying similarities and differences across clusters.

A. Complete Trajectory Clustering

To demonstrate the effectiveness of our model in identifying play-styles, we plotted heat maps of the trajectories in each cluster as well as the original trajectories for each $r \in R$ for E_1. In Fig. 3 we observe that there is a correlation between heat maps for both the clustered trajectories and the original trajectories separated by reward function. This shows firstly, that the desired behaviours in Table I are represented in the data through the use of the rewards in Table I. Secondly,

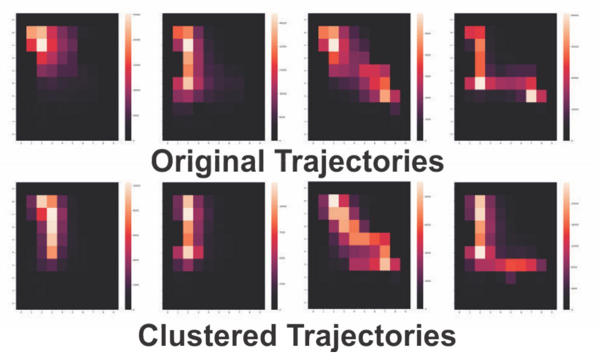

Fig. 3. Heatmap of visited states comparing 8000 clustered trajectories separated by our model (bottom) and original trajectories separated by reward function (top) for E_1.

TABLE II
Complete and Partial Trajectory Clustering Accuracy

E	Complete GMM	Complete kmeans	Partial GMM	Partial kmeans
E_1	**0.653**	0.598	0.499	**0.534**
E_2	0.707	**0.714**	0.602	**0.706**
E_3	**0.778**	0.705	**0.749**	0.670
E_4	**0.709**	0.706	0.576	**0.639**
E_5	**0.778**	0.652	**0.686**	0.678

we observe that the clustered trajectories depict the same behaviour. For example, R_4 sees the agent move to both bonus objectives (top-right) and the same behaviour is observed in the corresponding clustered set (bottom-right).

For quantitative analysis, we directly compared the set of all predicted labels (y') with the set of all ground truth labels (y) for each Environment $(E_1, \ldots, E_5)$ using (3). This resulted in the clustering accuracies shown in Table II for both k-means and GMM clustering algorithms. Table II demonstrates our model's ability to accurately cluster play-styles from completed trajectories across multiple varying environments.

B. Partial Trajectory Clustering

To investigate the ability to identify play-styles during gameplay, we clustered partial trajectories and measured the change in clustering confidence as a function of time. This was achieved using Algorithm 2 with the results depicted in Table II. We observe that an environment where trajectories are initially similar such as E_1 have a lower clustering performance. This indicates that the play-styles are initially well aligned while only diverging after some time.

Fig. 4 depicts the change in cluster assignment over time for two particular trajectories. We observe that the cluster assignments are non-volatile after an initial fluctuation. However, two noticeable shifts occur, namely $3 \rightarrow 0$ which corresponds to the shift $R_1 \rightarrow R_2$ and $0 \rightarrow 2$ which corresponds to the shift $R_2 \rightarrow R_4$. This change in clustering assignment is valid as these trajectories are both from R_4 where the optimal behaviour was to move $B_1 \rightarrow B_2 \rightarrow G$. This also

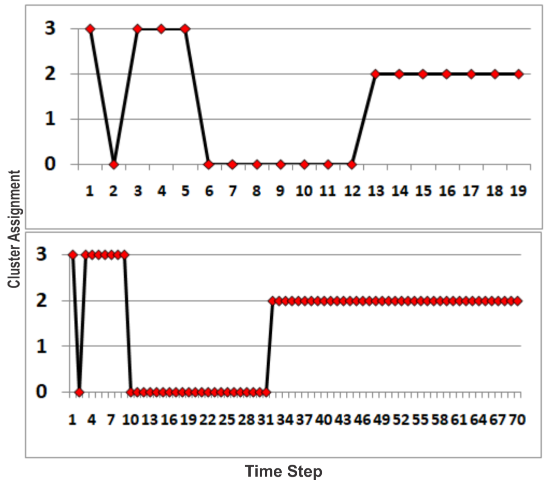

Fig. 4. Change in play-style prediction over time for two particular trajectories with behaviour corresponding to R_4 with 2 being the correct cluster assignment for both

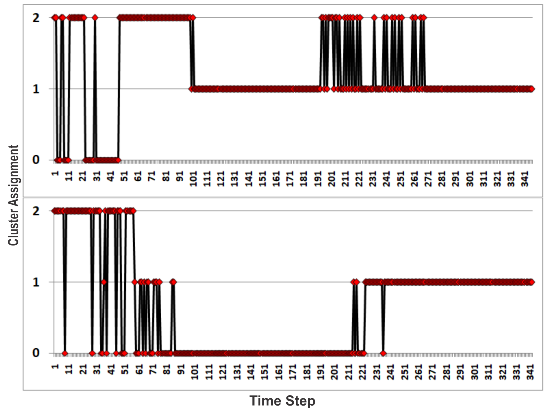

Fig. 5. Change in play-style prediction over time for two particular trajectories from the Mario dataset

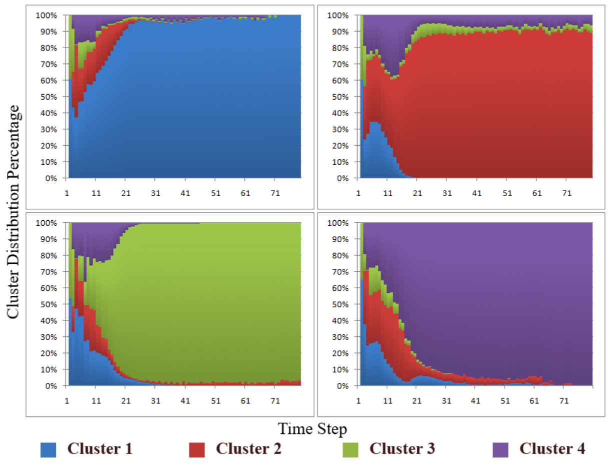

Fig. 6. Clustering assignment over time separated by identified clusters over all 5 grid environments

demonstrates that the autoencoder is separating data according to the reward functions. Therefore, this form of analysis can identify how a player's play-style changes over time as a result of performing some objective in some fashion.

To demonstrate the efficacy of our partial clustering over time we analysed all trajectories separated by the identified clusters across all 5 grid environments $(E1, \ldots, E5)$. This aggregation is possible since the play-styles we are trying to recover are the same across these different environments, as they would be across multiple levels in a video game. Fig. 6 depicts the results of this, where we observe that for all environments the clustering assignment converges to a unique cluster. This demonstrates that across multiple trajectories our approach to partial trajectory clustering produces consistent desired results.

We applied the same analysis in the Mario domain; the results are shown in Fig. 5. Here we observe that two initially different trajectories converge to the same play-style. Based upon initially differing behaviours it is natural for them to change over time and even converge, based on the changing dynamics of a video game. Additionally, we once again see the same initial fluctuations as observed in Fig. 4. The occurrence of fluctuations after the convergence is indicative of cross-over regions between play-styles. This is the main benefit of identifying style temporally as developers or players have a far denser signal indicating where and when their style changes.

The partial trajectory clustering accuracy (PTA), calculated using Algorithm 2, is shown in Table II. The accuracy does decrease using partial trajectories, but not by a significant amount. This decrease, however, is to be expected with play-styles sharing similarities, most notably the start location. This demonstrates the robustness of the clustering model to unseen data within the domain as well as demonstrating our ability to quickly identify the correct play-style.

VI. Identifying Unique Style Characteristics

In the case of our grid world domain, we have shown that we can group trajectories based upon an expected semantic meaning encoded into the data. However, it is not commonly the case that the semantic meaning behind clusters is known beforehand. The characteristics of each cluster need to be identified for the grouping to be useful for developers or players. We analyse the frequency (f_k) of state (s) occurrences across the identified clusters (k) to identify unique characteristics defined by (7). A state is an individual time step within a trajectory. In (7), $x, s_i \in S$, where S is the set of all possible states, and n denotes the number of different states.

$$f_k(x) = \sum_{i=1}^{n} 1[s_i = x] \tag{7}$$

In particular, we observe in Fig. 7 that the unique states for cluster 1 reside in the top right of the map for E_4 when $B_1 = 0, B_2 = 1$ as well as when $B_1 = 0, B_2 = 0$ which expectantly corresponds to behaviour 4 in Table I. These unique states for cluster 1 $(k = 1)$ were calculated as $f_1(x) - (\sum_{k=0}^{3}[f_k(x)]; k \neq 1)$. Here uniqueness emerges when a particular state occurs more than the total frequency for that state in all the other clusters. Similar forms of analysis can be used to determine similarities and differences between particular play-styles. For example, the similar states between clusters 1 and 3 as seen in Fig. 8 are calculated as the combined frequency for each common state. A state is said to be common if it exists within trajectories found in both clusters. Frequencies are combined by selecting the minimum frequency between the two clusters for each particular state (x). Here it is observed that both play-styles first go to B_1 before heading towards the middle in the direction of the goal. We saw in Fig. 7 the behaviour of cluster 1 was to move to B_2 before heading to the goal, however, this is not the case for cluster 3. Through observing the unique states as well as where the clusters are similar and different we can formulate a deeper understanding of the behaviours associated with each cluster. Applying the same form of analysis to the Mario dataset, shown in Table III, allowed us to discover the meaning behind the identified clusters. For example, we note that cluster 0 corresponds to players who die the least while cluster 2 contains players who are more likely to collect coins while also killing enemies. By engaging in this behaviour we see that they are more likely to die. Lastly, cluster 1 players tend to jump the least while also ignoring picking up any coins. The unique state counts were also used to influence the number of clusters chosen when using k-means. By considering whether the frequency of the most unique state was too large or small we could raise and lower the number of clusters we identified. For example, if the frequency of unique states was below a certain threshold, the particular cluster would be considered too similar to another. By doing this we settled on 3 clusters that had both unique cluster characteristics and a suitable amount of data.

TABLE III
THE 3 MOST FREQUENT UNIQUE STATES FOR EACH IDENTIFIED CLUSTER FOR ALL THE TRAJECTORIES IN THE MARIO DATASET

	State Description					
	jumps	kills	runs	coins	deaths	action
Cluster 0	17	1	2	3	0	9
	17	1	2	3	0	2
	12	1	0	2	0	2
Cluster 1	11	7	0	0	1	9
	11	7	0	0	1	2
	8	7	0	0	1	2
Cluster 2	38	8	0	8	2	9
	38	8	0	8	2	9
	49	8	0	8	1	9

VII. Conclusion and Future Work

This paper presents an approach to play-style identification using an unsupervised LSTM-autoencoder clustering model on variable-length trajectory data. Through empirical analysis, we demonstrated that the generated clusters matched the ground-truth underlying play-styles in our generated benchmark domain for both partial and complete trajectories. Similar results were demonstrated in the Mario domain demonstrating the model's ability to work in natural domains. The capacity of our model to recover the underlying play-style from partial trajectories demonstrates its usefulness during game-play. We were also able to show how our model could describe how a player's style evolved throughout their game-play using our partial trajectory analysis. This is particularly useful as our model requires no further engineering or training to perform this operation on partial trajectories. This abstraction technique was effective at identifying these behaviours using raw low-level data rather than hand-tuned features. Although clustering over time for play-styles has been attempted before [44] we achieve this result in an unsupervised setting. Through state frequency analysis we identified the characteristics which define our play-styles. This component makes it easier for developers and players not only to understand how an indi-

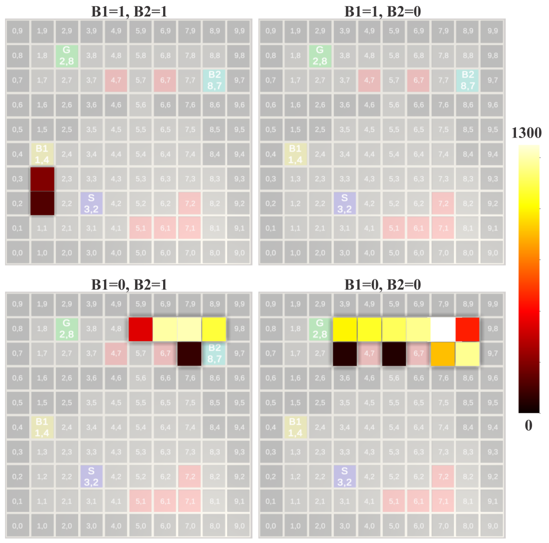

Fig. 7. Unique states observed for Cluster 1 for E_4 (excluding non-visited states)

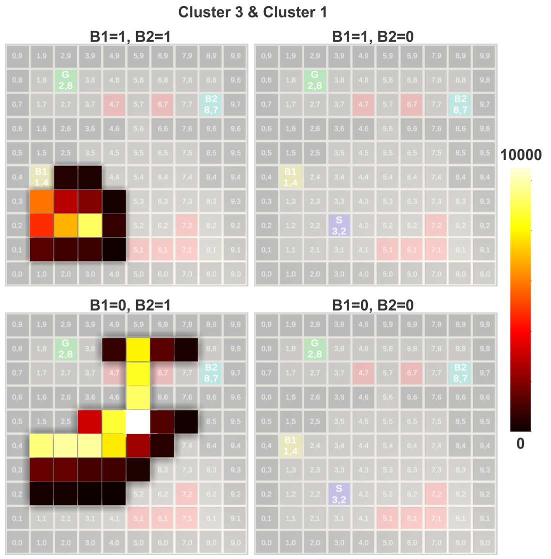

Fig. 8. Shared states observed between Cluster 3 and 1 for E_4 (excluding non-visited states)

vidual plays but also how that individual compares to others of differing styles, thereby allowing players or developers to make more informed decisions on which aspects they should change. A natural progression for this work would be to test the effectiveness of our model on higher dimensional trajectories more representative of modern games. Furthermore, another possible improvement may arise from the application of "attention" based models [45] as they are capable of handling temporal data.

References

[1] D. Charles, M. Mcneill, M. McAlister, M. Black, A. Moore, K. Stringer, J. Kücklich, and A. Kerr, "Player-centred game design: Player modelling and adaptive digital games," 2005.

[2] R. Dunn, "Learning style: State of the science," *Theory into practice*, vol. 23, no. 1, pp. 10–19, 1984.

[3] R. Bartle, "Hearts, clubs, diamonds, spades: Players who suit muds," *Journal of MUD research*, vol. 1, no. 1, p. 19, 1996.

[4] N. Yee, "Motivations of play in mmorpgs," 2005.

[5] A. Drachen, A. Canossa, and G. N. Yannakakis, "Player modeling using self-organization in tomb raider: Underworld," in *2009 IEEE symposium on computational intelligence and games*. IEEE, 2009, pp. 1–8.

[6] W. Helland-Hansen and G. Hampson, "Trajectory analysis: concepts and applications," *Basin Research*, vol. 21, no. 5, pp. 454–483, 2009.

[7] S. Ilyas and H. U. Rehman, "A deep learning based approach for precise video tagging," in *2019 15th (ICET)*. IEEE, 2019, pp. 1–6.

[8] A. Karpathy and L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3128–3137.

[9] Z. Yu, V. Ramanarayanan, D. Suendermann-Oeft, X. Wang, K. Zechner, L. Chen, J. Tao, A. Ivanou, and Y. Qian, "Using bidirectional lstm recurrent neural networks to learn high-level abstractions of sequential features for automated scoring of non-native spontaneous speech," in *2015 IEEE Workshop (ASRU)*. IEEE, 2015, pp. 338–345.

[10] M. Sundermeyer, R. Schlüter, and H. Ney, "Lstm neural networks for language modeling," in *Thirteenth annual conference of the international speech communication association*, 2012.

[11] P. Bertens, A. Guitart, P. P. Chen, and A. Perianez, "A machine-learning item recommendation system for video games," in *2018 IEEE (CIG)*. IEEE, 2018, pp. 1–4.

[12] C. Eggert, M. Herrlich, J. Smeddinck, and R. Malaka, "Classification of player roles in the team-based multi-player game dota 2," in *International Conference on Entertainment Computing*. Springer, 2015, pp. 112–125.

[13] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *International conference on machine learning*. PMLR, 2016, pp. 478–487.

[14] A. Drachen, M. Yancey, J. Maguire, D. Chu, I. Y. Wang, T. Mahlmann, M. Schubert, and D. Klabajan, "Skill-based differences in spatio-temporal team behaviour in defence of the ancients 2 (dota 2)," in *2014 IEEE Games Media Entertainment*. IEEE, 2014, pp. 1–8.

[15] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[16] G. N. Yannakakis, P. Spronck, D. Loiacono, and E. André, "Player modeling," 2013.

[17] J. R. Anderson, C. F. Boyle, and B. J. Reiser, "Intelligent tutoring systems," *Science*, vol. 228, no. 4698, pp. 456–462, 1985.

[18] K. Isbister and N. Schaffer, *Game usability: Advancing the player experience*. CRC press, 2008.

[19] T. W. Malone, "What makes things fun to learn? heuristics for designing instructional computer games," in *Proceedings of the 3rd ACM SIGSMALL symposium and the first SIGPC symposium on Small systems*, 1980, pp. 162–169.

[20] R. Koster, *Theory of fun for game design*. "O'Reilly Media, Inc.", 2013.

[21] G. Andrade, G. Ramalho, H. Santana, and V. Corruble, "Extending reinforcement learning to provide dynamic game balancing," in *Proceedings of the Workshop on Reasoning, Representation, and Learning in Computer Games, 19th (IJCAI)*, 2005, pp. 7–12.

[22] J. K. Olesen, G. N. Yannakakis, and J. Hallam, "Real-time challenge balance in an rts game using rtneat," in *2008 IEEE Symposium On Computational Intelligence and Games*. IEEE, 2008, pp. 87–94.

[23] P. Spronck, I. Sprinkhuizen-Kuyper, and E. Postma, "Difficulty scaling of game ai," in *Proceedings of the 5th International Conference on Intelligent Games and Simulation (GAME-on 2004)*, 2004, pp. 33–37.

[24] D. Thue, V. Bulitko, M. Spetch, and E. Wasylishen, "Interactive storytelling: A player modelling approach." in *AIIDE*, 2007, pp. 43–48.

[25] C. Thurau, C. Bauckhage, and G. Sagerer, "Learning human-like movement behavior for computer games," in *Proc. Int. Conf. on the Simulation of Adaptive Behavior*, 2004, pp. 315–323.

[26] B. G. Weber and M. Mateas, "A data mining approach to strategy prediction," in *2009 IEEE Symposium on Computational Intelligence and Games*. IEEE, 2009, pp. 140–147.

[27] E. Keogh and J. Lin, "Clustering of time-series subsequences is meaningless: implications for previous and future research," *Knowledge and information systems*, vol. 8, no. 2, pp. 154–177, 2005.

[28] S. Kisilevich, F. Mansmann, M. Nanni, and S. Rinzivillo, "Spatio-temporal clustering," in *Data mining and knowledge discovery handbook*. Springer, 2009, pp. 855–874.

[29] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series." in *KDD workshop*, vol. 10. Seattle, WA, USA:, 1994, pp. 359–370.

[30] M. Vlachos, G. Kollios, and D. Gunopulos, "Discovering similar multidimensional trajectories," in *Proceedings 18th international conference on data engineering*. IEEE, 2002, pp. 673–684.

[31] M. Nanni, *Clustering Methods for Spatio-temporal Data: Ph. D. Thesis*, 2002.

[32] M. Steinbach, L. Ertöz, and V. Kumar, "The challenges of clustering high dimensional data," in *New directions in statistical physics*. Springer, 2004, pp. 273–309.

[33] D. H. Fisher, "Knowledge acquisition via incremental conceptual clustering," *Machine learning*, vol. 2, no. 2, pp. 139–172, 1987.

[34] G. A. Carpenter and S. Grossberg, "A massively parallel architecture for a self-organizing neural pattern recognition machine," *Computer vision, graphics, and image processing*, vol. 37, no. 1, pp. 54–115, 1987.

[35] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990.

[36] T. W. Liao, "Clustering of time series data—a survey," *Pattern recognition*, vol. 38, no. 11, pp. 1857–1874, 2005.

[37] J. Gow, R. Baumgarten, P. Cairns, S. Colton, and P. Miller, "Unsupervised modeling of player style with lda," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 3, pp. 152–166, 2012.

[38] C. Frogner, C. Zhang, H. Mobahi, M. Araya-Polo, and T. Poggio, "Learning with a wasserstein loss," *arXiv preprint arXiv:1506.05439*, 2015.

[39] N. S. Madiraju, S. M. Sadat, D. Fisher, and H. Karimabadi, "Deep temporal clustering: Fully unsupervised learning of time-domain features," *arXiv preprint arXiv:1802.01059*, 2018.

[40] M. Guzdial and M. Riedl, "Game level generation from gameplay videos," in *Twelfth Artificial Intelligence and Interactive Digital Entertainment Conference*, 2016.

[41] C. Arzate Cruz and J. A. Ramirez Uresti, "Hrlb: A reinforcement learning based framework for believable bots," *Applied Sciences*, vol. 8, no. 12, p. 2453, 2018.

[42] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[43] Y. Zhuang, L. Chen, X. S. Wang, and J. Lian, "A weighted moving average-based approach for cleaning sensor data," in *27th International Conference on Distributed Computing Systems (ICDCS'07)*. IEEE, 2007, pp. 38–38.

[44] J. Valls-Vargas, S. Ontanón, and J. Zhu, "Exploring player trace segmentation for dynamic play style prediction," in *Eleventh Artificial Intelligence and Interactive Digital Entertainment Conference*, 2015.

[45] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.