Clustering Gamers's Time Series Data : A Case Study

Yuanli Pei[*]
Oregon State University, Corvallis, OR
peiy@eecs.oregonstate.edu

Amy Tan
Zynga Inc, San Francisco, CA
atan@zynga.com

Moises Goldszmidt[*]
Affiliation, Address
moises.goldszmidt@gmail.com

Alexandros Ntoulas
Zynga Inc, San Francsico, CA
antoulas@zynga.com

Abstract

Game industry plays an important in improving our entertainment in digital life. One of the main tasks for the game companies is to develop user-specific features to improve their gaming experience. The key factor is to understand player's behaviors during their game-play. This paper considers the following two tasks: Given user's time series behavior data, 1) what are the underlying playing states that controls user's behaviors at each time period, and 2) what are the natural groups of players based on their state transitions. We apply Hidden Markov Model (HMM) to identify gamers' hidden playing states, and use a Dirichlet clustering method to find player groups based on their state transitions. Our case study on one of the strategy game developed by Zynga Inc. reveals interesting results that can help gamer designers to improve the game.

1 Introduction

As game industry immerses, playing games has become an important entertainment activity in our digital life. To improve players' experience, it is very important for game companies to understand player's behaviors and design user-specific features. As players may change their playing strategies during the game-play, understanding gamer's playing pattens over time becomes an important task.

In this paper, we study the problem of understanding users' behaviors from their time series data obtained during the game-play. In particular, we focus on two tasks: 1) what are the underlying playing states that controls user's behavior at each time period, and 2) what are the natural groups of players based on their state transitions (game playing strategy). We model user's time series data using Hidden Markov Model (HMM) [6], with the playing states as the hidden nodes and the players' statistics at each time period as the observed features. We fit the HMM model and find users' playing states by applying Viterbi decoding algorithm. We then find the groups of players using an unsupervised Dirichlet clustering method.

We experiment our method on a mobile strategy game developed by Zynga Inc. Our method found three underlying playing states, and the clustering results reveal three interesting groups that contains players with similar transition states within each group. All these results are helpful to understanding players' behaviors and improve the game design.

2 Related Work

Clustering time series data

[*]This work was done while Yuanli Pei and Moises Goldszmidt were at Zynga Inc.

3 Method

Let $X_{it} = [X_{it1}, \cdots, X_{itD}]^\top$ be the measurements of player i at the t-th time period, where D is the number of features. Suppose we have the measurements of player i from 1 to the T-th time period[1], denoted as $X_i = [X_{i1}, \cdots, X_{iT}]^\top$. For a total number of N players, we have their time series data $X = \{X_1, \cdots, X_N\}$.

Our method includes two steps corresponding to the two tasks: 1) using HMM model to identify gamers' playing states at each time period, and 2) discovering player groups based on their transitions over states by clustering. Below we describe these two steps in detail.

3.1 Identify Gamers' Playing States Using HMM Model

We assume that there is an underlying playing state that controls users behaviors at each time period, and the state evolves as the player changes his strategy over time. We represent each gamers' data using a Hidden Markov Model (HMM) [6] chain with length T, the total time periods. Let Y_{it} be the hidden state representing the i-th gamer's underlying playing state at time t, and Y_{it} is discrete which can take on S values $\{1, \cdots, S\}$.

The mechanism of the gamers' HMM model is as below: 1) initially, a gamer starts with state $Y_{i1} \in \{1, \cdots, S\}$ according to an initial distribution π, with π_s being the probability of starting at state s; 2) all the Y_{it}'s evolves according to the *Markov property*: given Y_{it-1}, the state Y_{it} is independent of all the states prior to $t-1$, and the transition matrix is A, with A_{rs} being the probability of transitioning from state r to state t; 3) at each time t, the observations X_{it} only depends on the state Y_{it} parametrized by B, with B_s controlling the probability of observing X_{it} at state $Y_{it} = s$.

Given the observed data X and the number of states S, we can estimate the parameters by maximizing the likelihood of the observations

$$\max_{\pi,A,B} P(X|\pi, A, B) = \prod_{i=1}^{N} P(Y_{i1}|\pi)P(X_{i1}|Y_{i1}, B) \prod_{t=2}^{T} P(Y_{it}|Y_{it-1}, A)P(X_{it}|Y_{it}, B) \; .$$

However, in the application of modeling gamers' playing states, the number of states S is usually unknown. We will estimate S using criteria such as changes of data likelihood, AIC or BIC.

After estimating the parameters, the state sequence $Y_{i1}, \cdots, Y_{iT}$ for each user can be found using Viterbi decoding [6] which maximizes $P(Y_{i1}, \cdots, Y_{iT}|X, \pi, A, B)$.

3.2 Clustering Gamers based on State Transitions

The next step is to clustering the gamers based on their trends of state transitions. Here we adopt a similar method that has been successfully applied to crisis identification in distributed computing system [7]. Let $Y = [Y_1, \cdots, Y_N]^\top$ be the state transitions for all the players, where $Y_i = [Y_{i1}, \cdots, Y_{iT}]^\top$ denotes the i-th player's states from 1 to the T-th time. We denote Z_i as the underlying cluster assignment for the i-th player. Let K be the total number of (unknown) clusters, which we will found automatically.

3.2.1 The Clustering Model

We adopt the mixture of Dirichlet process model (DP) for clustering [4]. Suppose the clusters evolves according to a Dirichlet distribution with parameter α. We assume that each cluster k generates a Markov chain parametrized by $\{\lambda^k, \Phi^k\}$, where λ^k is the S vector for the initial state distribution, and Φ^k is the $S \times S$ transition matrix. We use the prior distribution for parameters in each

[1]Note that for different users, the total time periods are not necessarily the same. Here for simplicity we assume that all users are measured during the same amount of time

cluster is $G_0(\{\lambda^k, \Phi^k\}) = Dir(\hat{\pi}) \prod_{s=1}^S Dir(\hat{B}_{s\cdot})$, where $\hat{\pi}$ and $\hat{B}$ are the estimated parameters at the first step. The conditional probability

$$P(\{\lambda^k, \Phi^k\}_{k=1}^K | Z) = \prod_k G_0(\{\lambda^k, \Phi^k\}) \ . \tag{1}$$

Given the clustering model, the likelihood of the data of state transitions for all players is

$$P(Y|Z, \{\lambda^k, \Phi^k\}_{k=1}^K) = \prod_{i=1}^N \left(\prod_{s=1}^S \lambda^{\mathbf{1}[Y_{i1}=s]} \prod_{r=1}^S \left(\Phi_{rs}^{Z_i}\right)^{n_{irs}} \right) \ , \tag{2}$$

where $\mathbf{1}[\cdot]$ is the indicator function, and n_{irs} is the number of transitions from state r to state s for the i-th player.

3.2.2 MCMC Sampling of Posterior Distribution

Here we use a collapsed-space sampling method [5, 3] to obtain samples from the reduced-spaced posterior distribution $P(Z|Y)$, instead of the full-space distribution $P(Z, \{\lambda, \Phi\}|Y)$. This allows for easy sampling steps and faster convergence rate. The reduced-space posterior distribution is

$$P(Z|Y) \propto P(Z, Y) = P(Y|Z)P(Z).$$

The likelihood $P(Y|Z)$ can be computed by integrating out the cluster-specific parameters $\{\lambda^k, \Phi^k\}_{k=1}^K$. Substituting (1) and (2), we obtain

$$\begin{aligned} P(Y|Z) &= \int P(Y|Z, \{\lambda^k, \Phi^k\}_{k=1}^K) P(\lambda^k, \Phi^k|Z) d\lambda^k d\Phi^k \\ &= \prod_{k=1}^K \left[\frac{\prod_s \Gamma(\bar{\pi}_s)\Gamma(\sum_s \hat{\pi}_s)}{\Gamma\left(\sum_s \bar{\pi}_s\right) \prod_s \Gamma(\hat{\pi}_s)} \right] \times \prod_{k=1}^K \prod_r \left[\frac{\prod_s \Gamma(\bar{B}_{rs})\Gamma(\sum_s \hat{B}_{rs})}{\Gamma\left(\sum_s \bar{B}_{rs}\right) \prod_s \Gamma(\hat{B}_{rs})} \right] , \end{aligned}$$

where $\bar{\pi}_s = \hat{\pi}_s + \sum_i \mathbf{1}[Z_i = k, Y_{i1} = s]$, and $\bar{B}_{rs} = \hat{B}_{rs} + \sum_i n_{irs} \cdot \mathbf{1}[Z_i = k]$.

Sampling Z from Dirichlet distribution can be equivalently done as below [5]: set $Z_1 = 1$; for subsequent players, sample Z_i according to the following distribution

$$\begin{aligned} P(Z_i = k|Z_1, \cdots, Z_{i-1}) &= \tfrac{|\{i'<i:Z_{i'}=k\}|}{i-1+\alpha} \ , \quad \text{for } k \in \{Z_{i'}\}_{i'<i} \\ P(Z_i = Z_{i'}, \forall i' < i|Z_1, \cdots, Z_{i-1}) &= \tfrac{\alpha}{i-1+\alpha} \ , \end{aligned}$$

where $|\cdot|$ denotes the number of elements in a set.

Note that our clustering method can be easily extended to incorporate side information such as pairwise constraints [1], by only considering the samples Z that satisfying the constraints.

4 Experiments

We apply our method to Zynga Inc's strategy game called "Empires and Allies". In the game, the goal is to conquer all the battlefields in a global map, either with machine or with other players. To conquer the battles, the players need to build/upgrade base resources such as weapons and troops, which in turn requires game points that can be obtained from winning battles. Thus, the players need to tradeoff between building resources and conquering battlefields.

We form a dataset that contains player's 10 consecutive days data after installation, and subsample users that are active at all the 10 days. This results in 1719 players. Each player is measured with 67 metrics at each day, and we select 5 important features based on prior experience: `PvP` (people vs people battle), `Pve` (people vs machine battle), `Points` (number of points gained), `Session` (number of session started), `LevelUp` (whether a player level up) and `isPayer` (whether the player paid). We model `Points` with Gaussian distribution, `Session` with Poisson distribution, and the rest features with Bernoulli distribution.

4.1 Identifying Players' Daily States

We fit HMM model with different number of playing states and found that 3 states is reasonable. Table 1 lists the discovered playing states, **Aggressive**, **Defensive**, and **Moderate**. The **Aggressive** state captures the mode where the players focus on conquering battles, while the **Defensive** state describes the stage that they build the resources. The **Moderate** is a mixture of the two. These states interestingly identified the design of the game explained previously, namely, the players have to conquer battles and build resources intermittently in order to improve.

We then decode player's states at each day. Figure 1 plots the transitions for all the users among the 10 days. The result shows that most of the users starts with the **Moderate** or **Defensive** state, and then gradually transition to the **Aggressive** state. This is also consistent with the game design since the players can not start with many battles at the beginning due to the resources restriction, but they ultimately need to conquer all the battlefields.

Table 1: Discovered Playing States

Feature / State	**Aggressive**	**Defensive**	**Moderate**
Prob. `Pvp`	0.2472	0.0295	0.0947
Prob. `Pve`	0.2430	0.0581	0.1044
Mean `Points`	88.10	1238.36	476.22
Mean `Session`	4.58	34.19	12.95
Prob. `LevelUp`	0.1664	0.0177	0.0553
Prob. `Pay`	0.0031	0.0956	0.0320

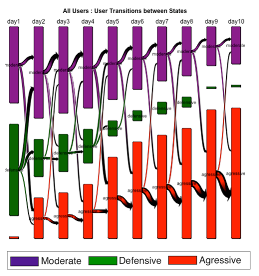

Figure 1: State transitions of all the Players at the first 10 days of installing the game.

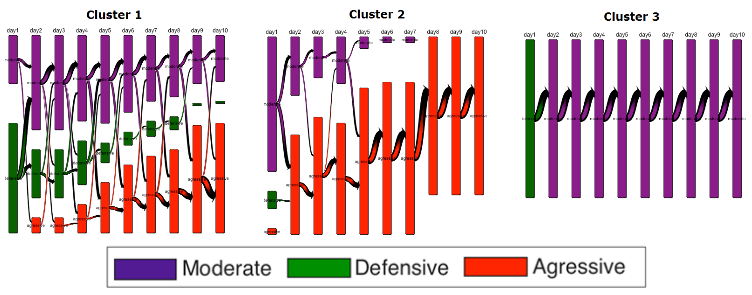

Figure 2: State transitions within three clusters found by our method.

4.2 Clustering Results

Next we clustering users based on their state transitions with the method explained in Sec. 3.2 . We compare with two baselines: Kmeans and Gaussian Mixture Model (GMM). Due to the fact that we do not have actual cluster labels, we evaluate the results using a polular internal evaluation method Davis-Bouldin (DB) index [2]. We tune the number of clusters for Kmeans using DB index, and report the one that has the best (the smallest) DB index value. Table 2 report the clustering results of all methods, where our method outperforms the baselines. We also plot the state transitions within each clusters and found that our method produces the most meaningful results. Here we show the within cluster transitions found by our method at Figure 2.

Table 2: Clustering results.

Method	**#Cluster**	**DB**
Our	**3**	**0.968**
Kmeans	5	2.628
GMM	4	1.803

5 Conclusion

In this paper, we study the problem of clustering gamer's time series. In particular, we solved two tasks: 1) identifying gamers' underlying playing states using HMH, and 2) clustering gamers' based on their state transitions using a Dirichlet clustering method. We experimented on one of the strategy game developed by Zynga Inc, and the results reveal interesting results that can help improve the game design.

References

[1] Sugato Basu, Ian Davidson, and Kiri Wagstaff. *Constrained Clustering: Advances in Algorithms, Theory, and Applications*. Chapman & Hall/CRC, 1 edition, 2008.

[2] David L Davies and Donald W Bouldin. A cluster separation measure. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (2):224–227, 1979.

[3] Michael D Escobar. Estimating normal means with a dirichlet process prior. *Journal of the American Statistical Association*, 89(425):268–277, 1994.

[4] Michael D Escobar and Mike West. Bayesian density estimation and inference using mixtures. *Journal of the american statistical association*, 90(430):577–588, 1995.

[5] Radford M Neal. Markov chain sampling methods for dirichlet process mixture models. *Journal of computational and graphical statistics*, 9(2):249–265, 2000.

[6] Lawrence R. Rabiner and Biing-Hwang Juang. An introduction to hidden markov models. *ASSP Magazine, IEEE*, 3(1):4–16, 1986.

[7] Dawn B. Woodard and Moises Goldszmidt. Online model-based clustering for crisis identification in distributed computing. *Journal of the American Statistical Association*, 106(493):49–60, 2012.