Real-Time Prediction of Gamers Behavior Using Variable Order Markov and Big Data Technology: A Case of Study

[1]Alejandro Baldominos, [1]Esperanza Albacete, [2]Ignacio Marrero and [1]Yago Saez

[1]Universidad Carlos III, Madrid, [2]Universidad de Granada, Spain

***Abstract* — This paper presents the results and conclusions found when predicting the behavior of gamers in commercial videogames datasets. In particular, it uses Variable-Order Markov (VOM) to build a probabilistic model that is able to use the historic behavior of gamers and to infer what will be their next actions. Being able to predict with accuracy the next user's actions can be of special interest to learn from the behavior of gamers, to make them more engaged and to reduce churn rate. In order to support a big volume and velocity of data, the system is built on top of the Hadoop ecosystem, using HBase for real-time processing; and the prediction tool is provided as a service (SaaS) and accessible through a RESTful API. The prediction system is evaluated using a case of study with two commercial videogames, attaining promising results with high prediction accuracies.**

***Keywords* — User Behavior, Prediction, Variable-Order Markov, Big Data, Real-Time.**

I. Introduction

WHEN gamers are playing, they are performing a sequence of actions which are determined by the game mechanics and vary from game to game. These actions are generally performed across the whole gaming time, making users switch from one game state to another presenting many similarities across different players. In order to incentivize their engagement, it is common to send messages, events or invitations to the users even when they are not playing. These engagement strategies are aimed to keep gamers interested in the game and active.

If the sequence of actions ends and the user does not play the game during a specific period of time (which depends on the game itself) then these gamers will be referred as churners.

The history of actions of a gamer will depend on how he interacts with the game. While the possible ways of interaction with a game may be unlimited, it is expected that some users will behave in a similar way, thus having similar historical actions records and similar game state transitions. Grouping these users would lead to the appearance of gamer profiles, i.e., groups of users sharing similar patterns of gaming behaviors. Having the ability to correctly identify those behavioral patterns will give videogame companies a chance to accurately predict which the most likely next user action is and to act in consequence.

Identifying gaming profiles can help companies to gain better understanding on how users interact with their games, to adapt their products to the customers and to determine the following metrics:

- The customer engagement: what the degree of commitment of gamers playing the game is.
- The churn rate: a measure for gamers that abandon the game.
- The conversion rate: how much revenue is the company able to earn from a certain profile of gamers, such as those who upgrade their license, make purchases, click on advertising, etc.

Understanding these profiles and learning from them comprise the first step for conditioning the behavior of gamers to optimize these metrics, and to ultimately increase the overall performance of the game. To do so, the company can take an active role so that users from a certain profile can move to other profiles providing better metrics or higher revenues.

In this paper we consider a situation where game events occur in sequences. In such cases, the ability of predicting future events based on current events can be valuable, and in consequence this paper presents a novel approach for predicting the behavior of gamers taking advantage of a Variable-Order Markov Model (VOM) which is built and used for training the model and for forecasting future events. The proposed prediction system will be applied to a case of study using data extracted from two commercial datasets.

In practice, this approach is similar to that of "sales funnel" that can be found in sales process engineering, inasmuch in sales funnels the company wants to know what are the specific sequences or processes that lead to a sale or conversion; and in this paper we want to know what will be the following action performed by a gamer (which can be or not a conversion) given the sequence of actions he/she has carried out.

In addition, it should be noticed that while playing, gamers usually perform a sequence of many actions very fast. This introduces two main challenges: in the first place, storing all of these sequences for hundreds of thousands users will result in a very large dataset; and secondly predictions must be provided in real-time in order to become valuable. In order to face these challenges introduced by the high volume and velocity of data, the proposed user behavior prediction system is designed and tested over a big data platform with high scalability in terms of storage and processing power.

The reminder of this paper is organized as follows. First, relevant papers related to this work are presented and discussed, which also use Markov-based techniques in order to face prediction problems in a variety of domains. Later we provide a formal and sound description of the problem to be tackled in this paper, as well as a proposal for solving this problem. After that, experimental results are shown and discussed. Finally, the paper concludes with a discussion of the contribution in this paper, open problems and future lines of work.

II. State of the Art

The high availability of data in recent years, due to new technologies enabling distributed storage and processing of information, has motivated the study of, among many others, the analysis of users' behavior. This analysis allows to better understand the user profiles in an application and how they will respond to different events. In particular, a subset of these analysis techniques has to do with trying to understand how the user will behave in the future, i.e., predicting his behavior.

Prediction of users' behavior is a topic of extensive research in a wide range of domains. The main reason is that this knowledge can be a useful asset when trying to improve user experience, and in some cases

DOI: 10.9781/ijimai.2016.367

can lead to higher revenues. For instance, a recent work from Yuan, Xu, & Wang [27] have presented a framework that studies the relationship between user behavior and sales performance in e-commerce.

Moreover, a very diverse set of techniques have been developed and applied to face the problem of behavior prediction in very different domains. The most extensively reviewed domain is probably online social networks. In this field, Wang & Deng [24] have recently stated that "being able to predict user behavior in online social networks can be helpful for several areas such as viral marketing and advertisement", and have proposed and evaluated a model for performing this task. A survey of techniques for understanding user behavior in online social networks has been published by Jin et al. [13], which also reviews user behavior in mobile social networks. More interestingly, this work reveals the main advantages of studying user behavior in online social networks for different stakeholders: Internet service providers, OSN service provides and OSN users. These advantages can be extrapolated for domains different than online social networks. Finally, a work from Kim & Cho [16] proposed a system for providing recommendations to mobile phone users by predicting their behavior using Dynamic Bayes networks, which can handle time-series data. Finally, in the field of online social games (which are closer to the present work), Zhu et al. [22] have recently studied theoretically the influence of user behavior in order to determine continuance intention, which is an issue related to churn prediction.

Moreover, techniques derived from Markov models have been extensively used in this kind of prediction problems, as it will be reviewed later. These are probabilistic models which, in essence, should fulfil the Markov property which states that the conditional probability of a future event depends only on the present, and not on past events. This property is usually referred as the model being memory-less, as the probability distribution of events in the future is known even when no information about the past is available.

The specific Markov model fulfilling this property, also known as first-order Markov model, only takes into account the present state to predict the future ones. However, this approach may turn out to be insufficient to carry out the task of behavior prediction. This fact has been shown with certain domains, such as web, as the next action cannot be accurately predicted by only considering the last action performed by the user [8].

More complex models arose from this one, which focuses in introducing memory to the original first-order model, thus considering the last k states in order to predict the future ones. This approach is named the kth-order Markov model. These kind of models evolving from the original concept of Markov models has been successfully used in different domains. For instance, one of the earliest applications of Markov models to the prediction of users' behavior in the web is described by Zukerman et al. [28] and later by Deshpande & Karypis [10], which also uses higher-order Markov models. Other, more recent works regarding the application of such models to the same domain include those of Maheswara-Rao & Valli-Kumari [19] and Madhuri et al. [5], where different variable kth-order Markov models are used in order to face the problem of the big amount of data. Moreover, in the works of Awad & Khalil [3] and Vishwakarma et al. [23], so-called all-kth models are used for similar purposes. Additionally, some works can be found in the fields of social networks, such as those by Lerman & Hogg [17] and Hogg et al. [12] where Markov models are used to predict the behavior of users in Digg and Twitter respectively; e-commerce, such as works by Rendle et al. [20] and Wu et al. [25] where Markov chains are used for recommending items to users (the second one using distributed processing to accommodate the presence of big data); education, such as the work of Marques & Belo [18] where Markov chains are applied for discovering user profiles in e-learning sites in order to customize the learning experience; or even in distributed systems, such as the works from Bolivar et al. where matchmakers based on hidden Markov models are proposed predict the availability of grid resources with the ultimate purpose of decentralized grid scheduling [6, 7].

Beyond the scope of web users' behavior prediction, techniques based on Markov models have also been successfully applied to a wide range of different domains. Some recent works involve using variable-order Markov models for predicting the behavior of drivers in vehicular networks, such as in the work from Xue et al. [26]; predicting the future location of mobile users, as in the work from Katsaros & Manolopoulos [15] or applying Markov-based models to the prediction of the inhabitants' activity in smart ubiquitous homes, as proposed by Kang et al. [14] or Alam et al. [1]. Also, an attempt to build a domain-independent predictor of user intention based on Markov model and considering fixed attributes from users is described by Antwarg et al. [2].

In the domain of games, several works have addressed the issue of predicting the gamers' behavior. For instance, Harrison & Roberts [11] describe a data-driven technique for modelling and predicting players' behavior in a game-independent manner. Moreover, some recent works use Markov-based techniques and sequence analysis to attain this objective, such as those published by Shim et al. [21] where the past performance of players is used to build a model from the observed patterns in users behavior to later allow prediction in MMORPGs (with a specific application to World of Warcraft) or by Dereszynski et al. [9] where hidden Markov models are used to learn models from past experience in order to predict the future behavior of players in real-time strategy games (with an application to StarCraft).

III. Problem Definition

Along this paper, an assumption is established expecting that the interaction of a user with the game can be described as a sequence of events (also referred in this paper as actions, as those events are performed by the user). Formally, let sequence $S = \{e_1, e_2, \dots, e_{t-1}, e_t\}$ be the chronologically sorted sequence of all events performed by a certain user in a certain game. This paper seeks a prediction algorithm A, such that when provided with a subsequence S' of S such that $S' = \{e_i, \dots, e_j\}$ for arbitrary values of i and j fulfilling $0 < i \leq j \leq t$, the algorithm outputs an event prediction: $e'_{j+1} = A(S')$.

If the value of e_{j+1} is known in advance (which happens when $j < t$), then the accuracy of this algorithm A can be computed and is defined as follows: for a given sequence $S' = (e\_i, \dots, e\_j)$ it can be stated that A is correct on S' if $A(S') = e_{j+1}$, i.e., if the predicted value e'_{j+1} matches the real value e_{j+1}. When working with more than one subsequences of a sequence S, the accuracy is the average fraction of subsequences in S for which the prediction of the algorithm A is correct (note that for computing this value e_t is excluded from S', as otherwise the value of e_{t+1} would be required to be known a priori).

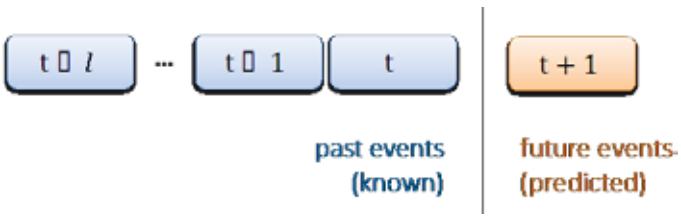

Fig. 1. Conceptual framework for the gaming prediction system. Past behavior events are known, and the system will predict the next event based on the behavior of the community of gamers.

Additionally, A could also be used to predict states of the game under certain circumstances. Let s_t be the state of the game at a time t, where this state is a set of attributes formally defining in an unambiguous way all aspects of the game at a certain point in time. If the state of the game changes only under the gamer's actions, then a new state s_{t+1} can be defined such that $(s_t, e_{t+1}) \rightarrow s_{t+1}$, i.e., the performance of event e_{t+1} under the game state s_t leads to the state s_{t+1}. As far as the game is fully observable (the current state s_t is always known), it is not stochastic, i.e., $\nexists s'_{t+1}$ such that $s_{t+1} \neq s'_{t+1}$ and $(s_t, e_{t+1}) \rightarrow s'_{t+1}$ and the game state is only affected by the gamer's actions (and not by the actions of other gamers or by the system itself), then algorithm A can be used interchangeably to predict either the next action performed by the gamer (e_{t+1}) or the next state of the game (s_{t+1}).

Notice that for games not fulfilling these conditions, then A will not be able to predict the next state of the game, but it will still predict the next user's action, as it depends only on the previous actions and not the states of the game.

This paper seeks an algorithm A providing high prediction accuracy.

IV. Proposal

The system proposed in this paper relies in three different technologies, which are (A) a prediction system for inferring the next action carried out by a gamer given the previous ones; (B) a big data platform for supporting the analysis of huge amounts of information; and (C) a web service for providing the prediction functionality as a service.

This section details how each of these technologies has been applied to the system.

A. Prediction System

The proposed system aims to predict the next action a gamer will carry out given his history of past actions, as depicted in Fig. 1.

In order to provide a prediction, the system will be given a Variable-Order Markov model (VOM) which will store the probabilities that an action occurs just after a particular sequence of actions, and which will be trained from historical data on gamers actions. In order to bound the length of the sequence of actions (as it could be potentially very long), a parameter l is introduced, which determines the maximum number of actions used for training the model, starting from the most recent one and going into the previous ones.

The process for training the model is as follows:

1. The system is introduced a sequence S of length $n+1$ (where $n < t$) containing several actions from a gamer: $S = \{e_{t-n}, \dots, e_{t-1}, e_t\}$.
2. For every different subsequence S' of actions $S' = \{e_i, \dots, e_j\}$, where $i \in [t-n, t-1]$ and $j \in [i, t-1]$ with maximum length l, the conditional probability $p(e_{j+1} | e_j, \dots, e_i)$ is computed.
3. The conditional probabilities computed in step (2) are used to update those already stored in the model. To ease this aggregation, absolute frequencies $n(e_{j+1} | e_j, \dots, e_i)$ may replace probabilities, which would require only additions to update the model.

We will show a very simple example which is not really representative but serves for the purpose of illustrating this process. Let's suppose a gamer performs the next sequence of actions during gameplay in chronological order: *Login*, *StartChallenge*, *KillEnemy*, *KillEnemy*, *KillEnemy*, *CompleteChallenge*. The system is configured to accept a sequence length $l = 2$. As a result, the next frequencies are computed:

$$n(StartChallenge \mid Login) = 1$$
$$n(KillEnemy \mid StartChallenge) = 1$$
$$n(KillEnemy \mid KillEnemy) = 2$$
$$n(CompleteChallenge \mid KillEnemy) = 1$$
$$n(KillEnemy \mid StartChallenge, Login) = 1$$
$$n(KillEnemy \mid KillEnemy, StartChallenge) = 1$$
$$n(KillEnemy \mid KillEnemy, KillEnemy) = 1$$
$$n(CompleteChallenge \mid KillEnemy, KillEnemy) = 1$$

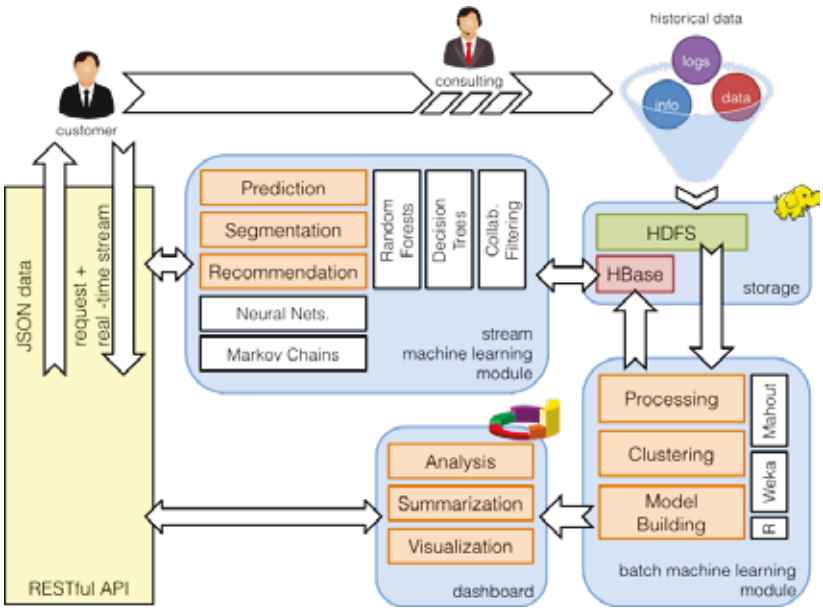

Fig. 2. General framework for online scalable machine learning. This framework comprises different modules which enable storage of Big Data over Apache Hadoop and HBase, as well as batch and streaming machine learning modules; and is operated through a RESTful API.

It is worth noting that, when computing the frequencies, the order of the actions is reversed, as they will be considered in inverse chronological order.

Once the probabilistic model is trained, it can be used for predicting future actions of gamers based on the last $k \leq l$ actions performed by them. In particular, given a known sequence of actions $s = \{e_{t-k}, \dots, e_{t-1}, e_t\}$ the action to be predicted, e_{t+1}, is computed given the next formula, where it should be noted that either relative (f) or absolute (n) frequencies can be used interchangeably:

$$e_{t+1} = \underset{e_{t+1}}{\operatorname{argmax}} f(e_{t+1}|e_t, e_{t-1}, \dots, e_{t-k})$$

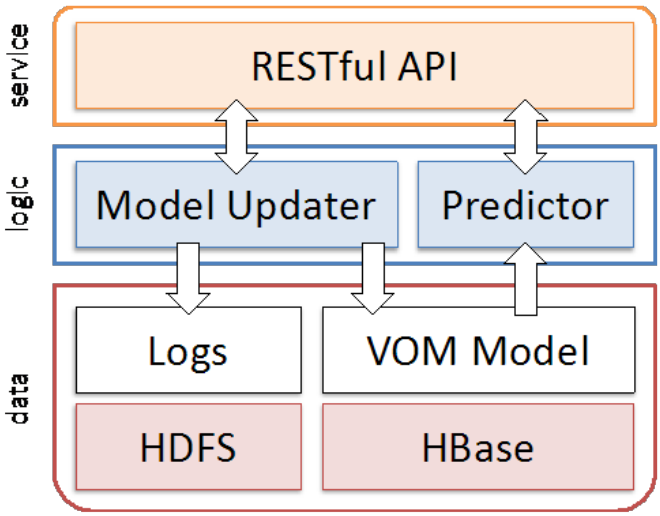

Fig. 3. Architecture of the user behavior prediction system. A web service provides an interface for both updating the probabilistic model stored in HBase (which also stores the incoming data in HDFS) and using the learnt model in order to return a prediction.

It should be noticed that it could be the case where the sequence s s is new and, as it never appeared before, the probabilistic model is unaware of it. To provide support for these cases, the previous formula can be generalized as follows:

$$e_{t+1} = \underset{e_{t+1}}{\operatorname{argmax}} f(e_{t+1}|e_t, e_{t-1}, \dots, e_{t-l'})$$

In this formula, $l' \in [1, l]$ and $s' = \{e_{t-l'}, \dots, e_t\}$ is the longest sequence which was already contained in the probabilistic model, i.e., it was also found previously during training. In the literature, this prediction mode is also called prediction by partial match or by longest prefix matching. In the case $s = e_t$ is not contained by the model, then the prediction cannot be performed.

TABLE I
DATA SCHEMA OVER HBASE FOR THE PREDICTION SYSTEM

rk	*cf:predict*	
$S^1 = e_t^1, e_{t-1}^1, \dots, e_{t-l^1}^1$	$q = e_{t+1_1}^1$	$v{:}n(e_{t+1_1}^1 \mid S^1)$
	$q = e_{t+1_2}^1$	$v{:}n(e_{t+1_2}^1 \mid S^1)$
	…	…
$S^2 = e_t^2, e_{t-1}^2, \dots, e_{t-l^2}^2$	$q = e_{t+1_1}^2$	$v{:}n(e_{t+1_1}^2 \mid S^2)$
	…	…
…	…	…

The next example illustrates the prediction process. The system receives the sequence of actions *CompleteChallenge*, *KillEnemy*, which have been carried out by the gamer in chronological order. The system will first try to predict e_{t+1} such that it maximizes the frequency $n(e_{t+1}|KillEnemy, CompleteChallenge)$. However, as that sequence was not already learnt by the model, the system will retry with the longest matching subsequence, which turns out to be *KillEnemy*.

Then, the system will look for the action e_{t+1} that maximizes the frequency $n(e_{t+1}|KillEnemy)n(e_{t+1}|KillEnemy)$. This action is actually *KillEnemy*, as $n(e_{t+1}|KillEnemy) = 2$ $n(e_{t+1}|KillEnemy) = 2$ and there are no higher frequencies for this particular sequence. Thus, the system will predict *KillEnemy* as the next action to be performed by the gamer.

This example, however, is extremely simple only serves the purpose of illustrating the proposed algorithm. It can be seen that there are many entries with the same frequency, and that can lead to draws. For instance, if the sequence *KillEnemy, KillEnemy* is considered, then both *KillEnemy* and *CompleteChallenge* are equiprobable predictions. In most cases, when very big datasets are used, draws are very infrequent. However, in the case they do happen and a most likely next event cannot be predicted, then the set of different events is returned.

B. Big Data Technology

When a videogame is popular, it will likely generate huge amounts of data at a high speed, involving the actions performed by its gamers. For this reason, it is important to design the prediction system with scalability in mind, in order to keep processing bounded and to be able to grow in storage capacity and efficiency as it is needed.

To meet these requirements, the prediction system will be built over big data technology. In particular, the Hadoop ecosystem is used following the architecture for big data real-time analysis described in Baldominos et al. [4]. The general framework supporting online scalable machine learning over Big Data is shown in Fig. 2, and the architecture showing how the different components relate to each other is shown in Fig. 3. It can be seen that the web service provides means for both updating the model with new data and providing predictions. The subsystem in charge of updating the model (*model updater* in the figure) also writes the logs to HDFS, so that batch analytics can be performed in later stages.

In order to store the probabilistic model Apache HBase will be used. This tool provides features for storing semi-structured information and is part of the Hadoop ecosystem. Table 1 shows how the model information is structured across the HBase table. As it is shown, the row key (*rk*) comprises the sequence of actions performed by the user, with a maximum length of *l*. There is only one column family (*cf:predict*) which will store the next action to be performed as the qualifier (q) along with the frequency of that action happening after the particular sequence, which is stored as the value (v).

The main advantage of HBase is that it provides indexed row keys, thus enabling efficient query over that field. As the row key in this application stores the subsequences of actions performed by users, we know that the maximum number of queries to be performed for returning a prediction is l, which will typically be a small value. This fact will enable the system to be able to provide real-time predictions.

C. Web Service

The prediction system proposed in this paper is offered as software-as-a-service (SaaS), so that external stakeholders can access it for

their own businesses. Moreover, the web service allows selecting one model in each request, so that the system can be reused for different videogames. The web service is provided through a REST API, which provides the following functionalities:

- Recording a new sequence of actions in the model (`record`), so that the frequencies matrix is updated to incorporate the new sequence. The following parameters must be specified when calling the service:
  - `chainLength`: the maximum chain length (l).
  - `sequence`: the sequence of actions performed by the gamer, in chronological order.
- Predicting the most likely future action (`predict`), for a specified sequence. The next parameter is required for calling the service:
  - `sequence`: the sequence of actions performed by the gamer, in chronological order, with maximum length *l*.
- Returning all the possible next actions along with their probability to occur (`predict_full`), for a specified sequence. This functionality allows the user to get better information about the shortcoming behavior of gamers. The next parameter is required for calling the service:
  - `sequence`: the sequence of actions performed by the gamer, in chronological order, with maximum length *l*.

V. Evaluation

In order to evaluate the proposed prediction system, two different datasets each within the particular domain of social videogames have been considered, as a part of a case of study. The next section provides a description of these datasets. Later, the methodological approach used for conducting the experiments is detailed, and the results obtained are discussed.

A. Datasets Description

Two different commercial games have been used as datasets for evaluating the prediction system. Each of those belong to a different domain, and have a different set of events, so the purpose is to validate that the system is able to generalize well when facing different domains. The commercial names of the games cannot be revealed due to a non-disclosure agreement, but the set of events is described. Although these events could be specified to a higher level of detail, the main purpose for defining these sets was to keep a bounded and reduced number of events while at the same time providing business value from their prediction.

Game #1

It is a social game where users compete to become the best sports director. The set of actions gathered for this game are the next ones:

- `StartSession`, when the user starts a new session.
- `BuyItem`, when the user buys an item in the game.
- `SendNeighbor`, when the user sends an invitation to a friend in the social network to join the game.
- `AcceptNeighbor`, when the user joins the game following an invitation sent by a friend.
- `HelpNeighbor`, when the user helps a neighbor's sport.
- `SendRequest`, when the user sends a social request.
- `AcceptRequest`, when a user accepts a social request sent by a neighbor.
- `UnlockContent`, when the user unlocks content in the game, which can happen after leveling up.
- `OfferAction`, when the user is offered an action by the game.
- `BuyHelp`, when the user buys help in the game (rather than asking neighbors for it).
- `StartTask`, when the user initiates a task associated with a building.
- `TaskCollect`, when the user collects a finished task from a building.
- `UserReferalInfo`, when the user clicks on an advertisement or promotion from a given campaign.

Game #2

It is a social game where users manage an amusement park, and they aim to receive as many visitors as possible. The next actions are logged for this game:

- `StartSession`, when the user starts a new session.
- `BuyItem`, when the user buys an item in the game.
- `SendRequest`, when the user sends a social request.
- `AcceptRequest`, when a user accepts a social request sent by a neighbor.
- `QuestConditionCompleted`, when the user completes a step of a multi-step quest.
- `QuestCompleted`, when the user completes a quest.
- `CollectionCompleted`, when the user completes a collection of items.
- `UserReferalInfo`, when the user clicks on an advertisement or promotion from a given campaign.

B. Experimental Setup

The experiments carried out for the evaluation follow a methodological approach, which is shared between all the datasets used. The purpose is to have the raw data subjected to a preprocessing phase, in order to conform the input format required by the training algorithm to build the probabilistic model. Then, the resulting data is again processed in order to avoid bias in the training or evaluation phases.

For the first preprocessing phase, the raw data is converted to sequences which can be inputted directly to the training algorithm to fed the Variable-Order Markov model. In particular, the original data is contained in the form of logs collecting information about the gamer (including the user identifier), the session (including the session identifier), the action carried out by the gamer, a timestamp and some additional parameters which fall out of the scope of this work. This raw data is converted to a file containing a sequence of actions for each gamer, which can be extracted from the logs just by considering the user ID and the timestamp and which already comply with the format expected by the training algorithm.

After the sequences are produced and before the model is trained, an additional phase takes place in order to guarantee that the evaluation results are not biased. To do so, first the complete set of sequences is randomly shuffled and divided in a training set and a test set, each having 70% and 30% of the original data respectively.

- **No cut**: no cut is performed whatsoever, so the sequence is kept intact and contains all the actions performed by the gamer up from the very beginning to the current moment.
- **Session cut**: the cut is performed randomly, but ensuring that a session is not split, i.e. the cut is performed before a `StartSession` event. The final sequence is kept from the very beginning up to the cut point.
- **Random-fixed cut**: the cut is performed in a random place of the sequence (may break a gaming session) and the final sequence will start at the cut point and have at most length l.

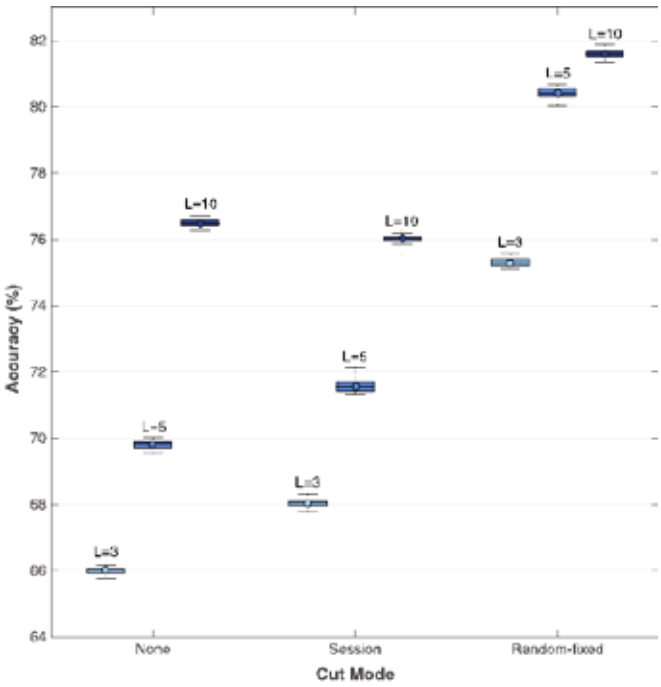

Fig. 4. Boxplot chart for Game #1, showing the distribution of the results of 30 experiments for each cut mode and length value.

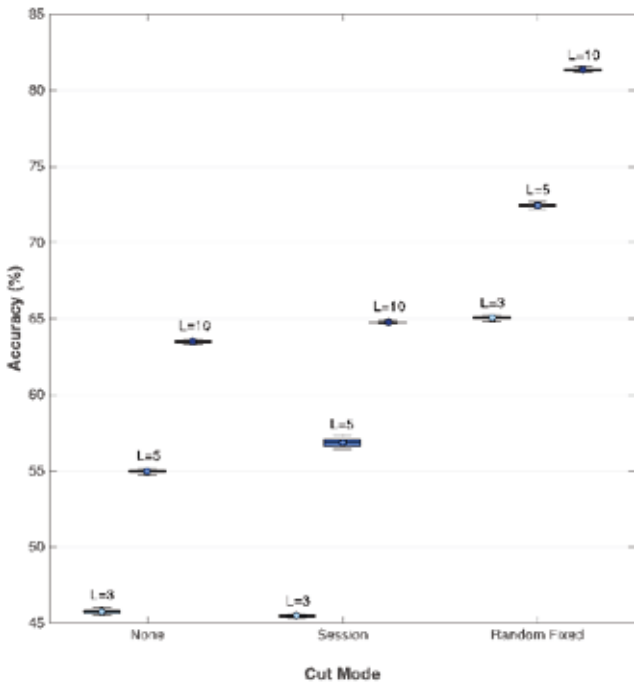

Fig. 5. Boxplot chart for Game #2, showing the distribution of the results of 30 experiments for each cut mode and length value.

TABLE II
PRECISION AND RECALL OBTAINED WHEN PREDICTING EVENTS IN THE GAME #1 DATASET WITH RANDOM-FIXED CUT AND $l = 10$.

Events	*Frequency*	*Precision*	*Recall*
StartSession	02.63%	42.14%	24.49%
BuyItem	06.22%	80.07%	90.75%
SendNeigbor	00.42%	98.60%	98.40%
AcceptNeighbor	00.04%	35.67%	09.75%
HelpNeighbor	01.80%	75.80%	72.19%
SendRequest	06.61%	90.83%	93.01%
AcceptRequest	00.81%	62.41%	24.32%
UnlockContent	02.54%	85.62%	65.00%
OfferAction	00.29%	84.28%	72.96%
BuyHelp	01.64%	69.77%	68.52%
StartTask	40.05%	83.84%	84.80%
TaskCollect	35.79%	63.03%	69.43%
UserReferalInfo	01.17%	64.14%	28.06%

For the experiments, different values of the sequence length (l) will be tried in order to study how the accuracy changes with the evolution of this parameter, in particular, values of $l \in \{3,5,10\}$ will be tested.

For each particular setup, i.e., each combination of sequence cut mode and length, 30 experiments are executed in order to obtain statistically significant results.

C. Results

The resulting distribution for the prediction system accuracy is described by its boxplot in Fig. 4 for the Game #1 dataset and in Fig. 5 for the Game #2 dataset.

While the results are further discussed in the next section, they

TABLE III
PRECISION AND RECALL OBTAINED WHEN PREDICTING EVENTS IN THE GAME #2 DATASET WITH RANDOM-FIXED CUT AND $l = 10$.

Events	*Frequency*	*Precision*	*Recall*
StartSession	04.96%	51.96%	15.62%
BuyItem	22.82%	82.89%	84.77%
SendRequest	19.92%	79.38%	65.07%
AcceptRequest	01.40%	50.48%	38.32%
QuestConditionCompleted	37.83%	82.36%	90.07%
QuestCompleted	11.25%	80.60%	88.74%
CollectionCompleted	00.02%	27.14%	02.50%
UserReferalInfo	01.81%	74.19%	42.74%

clearly show an evidence that for both datasets the same setup achieves a higher accuracy: the one combining a random-fixed cut and a sequence length of $l = 10$. More details about the specific per-event precision and recall, as well as the frequency of occurrence, are shown in table 2 for the Game #1 dataset and in table 3 for the Game #2 dataset.

D. Discussion

As it can be seen in figures 4 and 5, the best results are achieved using $l = 10$ and a random-fixed cut, as this setup outperforms the others for both datasets.

The fact that the accuracy increases with the value of l seems to have a simple explanation: when l is small, then very few past events are considered in order to predict the next ones, and the system will have higher chances to miss the prediction. However, higher values of l does not imply better results in all cases. Moreover, it should be noticed that the time required for training the system grows substantially with the value of l, but the accuracy grows asymptotically.

Regarding the cut mode, results show that higher accuracies are achieved with the random-fixed cut. In this case, the explanation is not trivial and is domain-dependent. As the datasets used in this paper are social games, it has been observed that the behavior of the users stabilize over time. When users start playing for the first time, they will in most cases perform some random actions in order to get used to the game and explore the scenarios. As time goes by, they stop exploring and start exploiting their resources to achieve higher scores. It should be noticed that the experiments with no cut and with session cut train the probabilistic model using data from the very beginning of the game, while the random-fixed cut may not. This explains why the random-fixed cut achieves higher accuracies, as it has a higher probability of training the model with data extracted when users have already stopped from exploring the game.

Tables 2 and 3 show how precision and recall are compared to the relative frequencies of occurrence of each event in each dataset. With the only exception of the *PayTransaction* event in the Game #1 dataset (an event with an exceptionally small frequency) precision and recall are always higher than the frequency. Also, it can be seen that the system has an outstanding ability to predict some events, such as *SendNeighbor*, even when this event occurs less than 1% of the times. These predictions about future users' behavior could be used to provide a better gaming experience as well as offering players customized offers or experiences to influence their behavior. Moreover, this information can be of great value for the videogame companies, due to the interest of predicting when the users will likely perform some events, such as spending money in in-game purchases (*BuyItem*) or involving other potential users in the game (*SendRequest* and *SendNeighbor* events).

Finally, while it is outside the scope of this evaluation, the time required by the system to provide a prediction is shown in previous work from Baldominos et al. [4]. The experiments were performed in a single-node cluster with 8 Intel Xeon processing cores and 16GB of RAM virtualized over VMWare ESXi 5.0, and Hortonworks HDP 2.1 as the Hadoop distribution, which includes Hadoop 2.4 and HBase 0.98; and JBoss AS 7 as application server. The use of Big Data technology along with a scalable web service allows an average response time of 20.29 ms. when requests are sequential, 98.43 ms. with 10 concurrent requests or 235.04 ms. with 30 concurrent requests. These times are far below one second and enables using this system for real-time prediction as a service. Moreover, the proposed architecture allows to easily scale horizontally just by adding nodes to the cluster, thus being able to perform concurrent reads among these nodes and improving concurrency. Also, the application server could be set in a cluster, and a load balancer could be used for improving performance when many requests are performed at the same time.

VI. Conclusions and Future Work

This paper has presented a novel approach for predicting the future behavior of users of social videogames given their historic behavior. This approach uses variable-order Markov models (VOM) in order to store the frequencies of an event happening after a certain sequence of actions. This model is then able to provide a real time estimation of the next event performed by a gamer.

This prediction system is useful for videogame companies to gain a better understanding on how their users and customers interact with their products, predicting whether certain users will probably make purchases in the shortcoming future or, on the other hand, will abandon the game to become *churners*. This knowledge will enable the company to provide specific experiences and offers customized for each gamer in order to condition his/her behavior; and will provide business insights to discover how their users engage with their products as well as what steps lead them to *convert* into customers.

In a more general way, the system is developed in a general-purpose fashion, so it could be used as a prediction system for any data that can be represented as sequence of events. While this paper has only evaluated the application of this prediction system to the field of videogames, in practice it could be applied to a broad variety of domains, including but not limited to sales processes, web navigation, smartphone usage or even general human behavior analysis.

The system is deployed over Big Data technology, in particular, the Hadoop ecosystem is used in order to scale out horizontally thus being able to tackle the problem of big volume and velocity of data which is inherent to this domain. Data is stored in HBase, thus taking benefit from indexed row keys in order to be able to perform queries on real-time. The prediction system is then provided as a service, which is able to attend 30 concurrent requests in about 200 ms. in a single-node cluster.

In order to evaluate the developed prediction system, experiments have been designed and conducted using two real-world datasets which are in production stages as a part of a case of study. Experiments involved testing the system with different setups of the chain length and the cut mode, this last parameter serving as a way to prevent bias. Results have shown that better results are achieved with a chain length of $l = 10$ and random-fixed cut mode, attaining accuracies of about 82% in both datasets. It makes sense that higher chain lengths lead to better results, as it delves more into the past behavior, being the main disadvantage that the time training the model grows substantially with the value of l. Also, the fact that random-fixed cut provides better results may be explained as the behavior of users in the game stabilize over time, and this cut mode often ignores the behavior at the very beginning of the gaming history of a user.

Per-event results show that precision and recall improve significantly over the frequency of each event, except in very rare cases. This points out that the model is able to learn and extract predictable patterns from data, and thus to provide accurate predictions.

The results from this case of study have not been compared with those resulting for the application of different techniques; as it is the case that other machine learning algorithms are not suitable for providing both training (model update) and prediction in real time. Even when some techniques such as Naive Bayes may be suitable, they must be developed fitting the described Big Data Real-Time architecture, which requires substantial effort. For this reason, this comparative evaluation is left for future work.

Other future lines of work may involve clustering users in order to learn specific models for each cluster or group, so that different types of gamers are detected and prediction models are specialized for each of these. Moreover, it would be interesting in delving in churn prediction, which so far could be supported by the system as long as "*churn*" is defined as an event. Finally, it would be interesting to provide methods not only to predict the next events performed by gamers, but also to condition their behavior so that they will have a higher probability to perform an event which is beneficial to the game company, such as purchasing a certain item. This would require the system to know which are the possible actions to be performed by the game itself in order to provide the more suitable action to achieve a certain behavior in the user, thus providing a more customized gaming experience.

Acknowledgment

This work is part of *Memento Data Analysis* project, co-funded by the Spanish Ministry of Industry, Energy and Tourism with identifier TSI-020601-2012-99 and is supported by the Spanish Ministry of Education, Culture and Sport through FPU fellowship with identifier FPU13/03917.

References

[1] Alam, M. R., Reaz, M. B. I., & Mohd Ali, M. a. (2012). SPEED: An inhabitant activity prediction algorithm for smart homes. *IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans*, *42*(4), 985–990.

[2] Antwarg, L., Rokach, L., & Shapira, B. (2012). Attribute-driven hidden markov model trees for intention prediction. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, *42*(6), 1103–1119.

[3] Awad, M., & Khalil, I. (2012). Prediction of User's web - browsing behavior : Application of Markov Models. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, *42*(4), 1131–1142.

[4] Baldominos, A., Albacete, E., Saez, Y., & Isasi, P. (2014). A scalable machine learning online service for big data real-time analysis. *2014 IEEE Symposium on Computational Intelligence in Big Data (CIBD)* (pp. 1–8).

[5] Bindu Madhuri, C., Anand Chandulal, J., Ramya, K., & Phanidra, M. (2011). Analysis of Users' Web Navigation Behavior using GRPA with Variable Length Markov Chains. *International Journal of Data Mining & Knowledge Management Process*, *1*(2), 1–20.

[6] Bolivar, H., Martinez, M., Gonzalez, R., & Sanjuan, O. (2012). A multi-agent matchmaker based on hidden Markov model for decentralized grid scheduling. *Proceedings of the 2012 4th International Conference on Intelligent Networking and Collaborative Systems (INCoS)* (pp. *62*–69).

[7] Bolivar, H., Martinez, M., Gonzalez, R., & Sanjuan, O. (2014). Complexity analysis of a matchmaker based on hidden Markov model for decentralized grid scheduling. *International Journal of Grid and Utility Computing, 5(3)*, 190–197.

[8] Chierichetti, F., Kumar, R., Raghavan, P., & Sarlos, T. (2012). Are web users really Markovian? *Proceedings of the 21st international conference on World Wide Web - WWW '12*, WWW '12 (p. 609). ACM.

[9] Dereszynski, E., Hostetler, J., Fern, A., Dietterich, T., Hoang, T.-T., & Udarbe, M. (2011). Learning Probabilistic Behavior Models in Real-Time Strategy Games. *Proceedings of the Seventh AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE-11)* (pp. 20–25).

[10] Deshpande, M., & Karypis, G. (2004). Selective Markov Models for Predicting Web Page Access. *ACM Transactions on Internet Technology*, *4*(2), 163–184.

[11] Harrison, B., & Roberts, D. L. (2011). Using sequential observations to model and predict player behavior. *Proceedings of the 6th International Conference on Foundations of Digital Games (FDG)* (pp. 91–98).

[12] Hogg, T., Lerman, K., & Smith, L. (2013). Stochastic Models Predict User Behavior in Social Media. *arXiv preprint arXiv:1308.2705*. Retrieved from http://arxiv.org/abs/1308.2705

[13] Jin, L., Chen, Y., Wang, T., Hui, P., & Vasilakos, a V. (2013). Understanding user behavior in online social networks: a survey. *IEEE Communications Magazine*, *51*(9), 144–150.

[14] Kang, W., Shine, D., & Shin, D. (2010). Prediction of state of user's behavior using Hidden Markov Model in ubiquitous home network. *Industrial Engineering and Engineering Management (IEEM), 2010 IEEE International Conference on* (pp. 1752–1756).

[15] Katsaros, D., & Manolopoulos, Y. (2009). Prediction in wireless networks by Markov chains. *IEEE Wireless Communications*, *16*(2), 56–63.

[16] Kim, Y., & Cho, S.-B. (2009). A recommendation agent for mobile phone users using Bayesian behavior prediction. *3rd International Conference on Mobile Ubiquitous Computing, Systems, Services, and Technologies, UBICOMM 2009* (pp. 283–288).

[17] Lerman, K., & Hogg, T. (2012). Using Stochastic Models to Describe and Predict Social Dynamics of Web Users. *ACM Trans. Intell. Syst. Technol.*, *3*(4), 1–33.

[18] Marques, A., & Belo, O. (2010). Discovering student web usage profiles using Markov chains. *9th European Conference on eLearning 2010, ECEL 2010*, *9*(1), 335–342.

[19] Rao, V. V. R. M., & Kumari, V. V. (2011). An Efficient Hybrid Successive Markov Model for Predicting Web User Usage Behavior using Web Usage Mining. *International Journal of Data Engineering*, *1*(5), 43–62.

[20] Rendle, S., Freudenthaler, C., & Schmidt-Thieme, L. (2010). Factorizing personalized Markov chains for next-basket recommendation. *Proceedings of the 19th international conference on World Wide Web (WWW)* (p. 811).

[21] Shim, K. J., Sharan, R., & Srivastava, J. (2010). Player Performance Prediction in Massively Multiplayer Online Role-Playing Games (MMORPGs). *Advances in Knowledge Discovery and Data Mining* (pp. 71–81).

[22] Song-Zhu, D., Jon-Kuo, M., & Hao-Liou, S. (2014). The Study of Continuance Intention for Online Social Games. *Proceedings of the 3rd International Conference on Advanced Applied Informatics* (pp. 230–235).

[23] Vishwakarma, S., Lade, S., Kumar-Suman, M., & Patel, D. (2013). Web User Prediction by Integrating Markov Model with Different Features. *International Journal of Engineering Research and Science & Technology*, *2*(4), 74–83.

[24] Wang, P., & Deng, Q. (2014). User Behavior Prediction: A Combined Model of Topic Level Influence and Contagion Interaction. *Proceedings of the 2014 20th IEEE International Conference on Parallel and Distributed Systems* (pp. 851–852).

[25] Wu, T., He, H., Gu, X., Peng, Y., Zhang, Y., Zhou, Y., & Xu, S. (2013). An intelligent network user behavior analysis system based on collaborative Markov model and distributed data processing. *Proceedings of the 2013 IEEE 17th International Conference on Computer Supported Cooperative Work in Design (CSCWD)* (pp. 221–228).

[26] Xue, G., Li, Z., Zhu, H., & Liu, Y. (2009). Traffic-known urban vehicular route prediction based on partial mobility patterns. *Proceedings of the International Conference on Parallel and Distributed Systems (ICPADS)* (pp. 369–375).

[27] Yuan, H., Xu, W., & Wang, M. (2014). Can online user behavior improve the performance of sales prediction in E-commerce? *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)* (pp. 2347–2352).

[28] Zukerman, I., Albrecht, D. W., & Nicholson, A. E. (1999). Predicting Users' Requests on the WWW. *Proceedings of the 7th International Conference on User Modeling* (pp. 275–284).

Alejandro Baldominos is Computer Scientist and Engineer since 2012 from Universidad Carlos III de Madrid, and got his Master degree in 2013 from the same university. He is currently working as a researcher in the Evolutionary Computation, Neural Networks and Artificial Intelligence research group (EVANNAI) of the Computer Science Department at Universidad Carlos III de Madrid, where he is currently working in his Ph. D. thesis with a studentship granted by the Spanish Ministry of Education, Culture and Sport. He also works as Professor of the Master in Visual Analytics and Big Data at Universidad Internacional de la Rioja. He has published several conference and journal papers in the fields of context-aware systems, artificial intelligence and big data; and has been involved in several national and European research projects.

Esperanza Albacete received the B.S. in Computer Science from Universidad Carlos III de Madrid in 2009 and the M.Sc. in Computer Science in 2010 in the same university. Currently, she is a Ph. D. student in Computer Science and works as research assistant at the Computer Science Department of Universidad Carlos III de Madrid. Her research interests include Human-Computer Interaction, Ontologies and Advanced Databases Technologies.

Ignacio Marrero is M.Sc. in Astrophysics from Universidad de Granada. He worked in CSIC (most relevant Spanish research center) until 2000, in the areas of experimental data analysis and modelling; and until 2011 worked as a Solution Architect and Manager of Innovation and Research in diverse projects and companies from different sectors. Since 2011 he is working as Project Manager in different companies in the field of Big Data, Business Intelligence, Internet of Things and Advanced Data Analytics. The words "Data Economy & Innovation" clearly summarizes his interests, professional life and overall, his passion; allowing him to be in the front line of Big Data in Spain.

Yago Saez received the degree in computer engineering in 1999. He got his Ph.D. in Computer Science (Software Engineering) from the Universidad Politécnica de Madrid, Spain, in 2005. Since 2007 is vice-head of the Computer Science Department from the Carlos III University of Madrid, where he got a tenure and is associate professor. He belongs to the Evolutionary Computation, Neural Networks and Artificial Intelligence research group (EVANNAI) and member of the IEEE Computational Finance and Economics Technical committee. Nowadays is involved in a European Project in collaboration with public and private companies and he is affiliated with Auctionomics (a US-based World leader auction advisory company). His main research areas encompass the evolutionary computation techniques applied to Computational Economic and Finance and Computational Intelligence to Games fields.