Using Computer Gaming Models to Understand the Behavior of Industrial Machines

Silvia Melicchio  
University of Pisa  
Pisa, Italy  
smelicchio@gmail.com

Marco Rocchetti, Gustavo Marfia and Alessandro Amoroso  
University of Bologna  
Bologna, Italy  
{marco.roccetti, gustavo.marfia, alessandro.amoroso}@unibo.it

Abstract—The exponential growth of audiovisual and textual data is aggressively leading to the creation of tools and systems capable of selecting and recognizing given patterns of information. Such urge is at the basis of the development of machine learning techniques, techniques that are today employed to analyze all kinds of information. Computer gaming is, with no doubt, one of the fields where such techniques have been put to good use most pervasively. For example, in order to provide better opponents in human vs. machine games, machine learning techniques are being extensively utilized to model how players behave during gaming sessions. These same ideas, however, could be put to good use when realizing completely different systems and applications, as for example when analyzing how industrial machines behave while running complex tasks. We here show that techniques that have been well accepted and deployed through entertainment technologies can also be applied to industrial ones. In particular, we will show how it is possible to model the operation of an industrial machine as a game in order to assess whether the machine is following its regular behavior path. The result of this study is that it is possible to model also complex machines by patterns where divergences from such patterns can represent indicators of malfunctioning or unexpected working states.

Keywords—machine learning; games; game modeling; industrial applications.

I. INTRODUCTION

Almost as effective as military research, entertainment has been a forerunner in many different ways in the development of advanced technologies. Game development has often been the reason for the creation of new techniques and algorithms that could improve the usability (e.g., Nintendo Wii console) and the graphical realism (e.g., GPUs) of computer systems.

With no doubt, an important stream of research in computer gaming amounts to the creation of computer opponents in human vs. machine games [1]-[3]. In fact, when engineering computer opponents, substantial work has been done to create opponents that could be perceived as credible, i.e., opponents that could be fought while truly entertaining a player, appearing neither too strong nor too weak for humans. Clearly, this has required the devise of techniques that could help predicting how a human player would behave in real time and the creation of algorithms that could generate moves and strategies that guided, following some type of criteria, computers according to the expected actions of humans.

All this is done assuming that a human player will stick to play a game as long as his/her computer opponent is compatible in terms of its skillset. Clearly, the moment a computer opponent is too strong, a human player will not feel comfortable when playing a game that would end too soon. On the other side, when a computer opponent is too weak, a human player would feel annoyed and unchallenged from winning too easily. This is why given human player patterns should lead to the adoption of corresponding computer player patterns in the mind of game designers.

Now, with the increase of their complexity, mainly due to the adoption of many different components controlled by electronic parts and software, industrial machines exhibit characteristics that resemble how computer games work. In fact, many industrial machines may be modeled as a mix of different players interacting among each other, as they are often composed of different subsystems that behave differently, producing different outputs, depending on their particular final outcome. Today, in particular, this happens because industrial machines have grown to become complex ecosystems composed by an excess of sensors and actuators that are managed by different software components (e.g., SCADA components in industrial control systems). As the amount of information that industrial machines as also grown, and this information can be easily seen as the composition of information coming from different sources and actors, this is why we believe that the same techniques that have been so far adopted to model the online behavior of computer game players can also be applied in the world of industrial machines.

In order proceed with this study, we considered a real family of industrial machines for our study: packaging machines. Packaging machines are particular machines that are utilized to package any given goods (e.g., medications, food, cigarettes, etc.). These are particularly interesting machines as they are typically very flexible and can be set to package different goods depending on their particular initial setup. For what concerns our study, this characteristic is meaningful as it means that under different settings different components of a given machine may exhibit different behaviors.
The particular contribution of this work is that of developing a computer gaming inspired model for industrial machines. To do this we resort to well known mathematical tools such as Bayesian networks. Bayesian networks, in fact, have been thoroughly utilized in computer gaming for the forecasting of gamer behavior. We here apply the same procedures and show that complex industrial machines can be modeled utilizing the same class of techniques. The final result is that it is possible to develop fine grained models that exhibit industrial machine behavior, which let operator forecast well in advance, in comparison the currently employed techniques, when normal operations are not being followed, possibly leading to faults.

The remainder of this paper is organized as follows. In Section II we provide a brief overview of the works that fall closest to our approach. We then proceed providing our model in Section III and preliminary results of our experimentation in Section IV. We finally conclude with Section V.

II. RELATED WORK

Although industrial machines have often evolved, from predominantly mechanical and chemical ones to machines rich of electronic components and controlling software systems, no well-established techniques have been so far engineered for modeling their behavior. The most advanced systems adopt Field Failure Data Analysis (FFDA) techniques, techniques that have been seen their birth for the troubleshooting and error recovery of complex computer systems. We will hence here describe three of FFDA related works that are relevant for our study.

In [4], the authors analyzed the behavior of a Networked Windows NT System that included over five hundred machines. Per each server, the authors collected log data that reported on its outage time, its cause and its reboot time. With such information the authors aimed at providing an overview of the outage times of the whole network, in order to understand their main causes. In addition, they provided error propagation matrices exhibiting how different domains within the set of servers depend upon each other.

The work carried out in [5] performs a 9-month reliability analysis for the packaging of beer production. The authors here identified the most important failure modes and calculated the descriptive statistics that best fit actual failures rates. A study that generalizes such work reviewing relevant research that has been published regarding the reliability, availability and maintainability analysis in food production lines is available in [6]. In this work the authors aimed at identifying the critical points of production systems resorting to total quality management tools as failure mode and effect analysis, Pareto analysis, statistical process control, etc.

All of the techniques that have here been briefly mentioned, however, build upon the presence of a fault in a system. None of such models aimed at characterizing a normal operational mode, in order to be able to more closely predict the occurrence of a fault. This is why we here aim at standing one step behind the mentioned analysis building automatic models that may show when a system is behaving normally and hence signal as soon any deviation from such behavior occurs.

III. MODEL

The model that we here propose builds upon the fact that many modern industrial machines (and as such, many packaging ones) can be thought of as information sources, where the information that is being produced pertains the status of the parts that compose the system (e.g., actuators, sensors, software systems, etc.). While running, hence, many of such machines produce logs that summarize the operations of all of their information source components, as represented in Figure 1.

However, differently than how they are represented in Figure 1, log data sources will often have some type of hierarchical relationship. For example, the trigger of an actuator could cause a physical movement that could in turn lead to a change to the quantity (e.g., pressure) that is sensed by a given sensor. Rather than belonging to a flat hierarchy, as represented in Figure 1, log data sources could easily depend upon each other during the normal operation of a machine.

Such relations very closely resemble those that typically exist between players in computer games. When a human player makes a move, for example, computer ones react accordingly depending on: (a) what move the human player performed, and, (b) the estimated skillset exhibited, up to that point, by the human player. Clearly, causal relations are those that can be described and understood the most, but the nature of such relations might be way more complex and intertwined depending on the number of variables that are involved.

Now, one of the mathematical tools that have been used most, in literature, to model the evolution of a player in computer gaming is Markov Models. With Markov Models, in fact, the present state of a system, which described by a set of stochastic variables, depends non-deterministically upon previous states. Bayesian networks, instead, represent a generalization of Markov Models, as Markov Models are simply Bayesian networks for modeling time series data. We will here hence resort upon such mathematical tool, which naturally incorporates the concept of representing conditional independencies between a set of random variables.
When modeling the logs of an industrial machine as a Bayesian network, a few steps are involved. The first step amounts to associating a binary random variable (i.e., true or false) to all the events that are being logged. In order to understand why this is done, consider the following exemplar case, where the space of all possible events is given by events A, B, C and D. Let us now assume that we want to represent a step where the sequence of events “AAACBA” appears with this model. We will represent such step as true, true, true, false, as all events, except D, whose absence is given by the false entry, appeared at the given step. A natural objection to such procedure is that with such step we are losing any information provided by the frequency of each event. Our results will show that such loss does not jeopardize the predictive accuracy of our model, although capturing event frequency could be starting point for future refinements of our work.

Once log data has been turned into binary random variables, we can proceed constructing a Bayesian Network following two steps. The graph nature of a Bayesian network makes, in fact, its construction process amount to a two-step process: the first step entails learning the structure of the Bayesian network (i.e., learn which nodes of the network are connected), the second the probability tables (i.e., the weights of the connections).

Different techniques can be found in literature for learning the structure of a Bayesian network. In the context of this work we resorted to the well-known K2 algorithm [7]. K2 is a greedy “hill climbing” algorithm that adds connections (i.e., conditional causality relations) between nodes (i.e., events) based on a scoring function. Compared to other popular algorithms, K2 allows finding more than a single parent node, providing a greater ability to capture the reality with the Bayesian network model.

Once the network structure is known, the second step is given by the learning of the probability tables (i.e., the conditional probabilities that link the two events) of the network. These probabilities have been determined resorting to the simple estimator implemented by the Weka statistical package [8]. Figure 2 briefly summarizes the process that takes from logs to the final Bayesian network construction.

Now, before we go any further, we must finally consider that the chosen model (model of transition from log to causality between events) has one additional important parameter that should be considered: the time frame within which relations between events can take place. In fact, one can be interested in how the events evolve in very short (e.g., five seconds) or long (e.g., five hundred seconds) time horizons, with an impact on the complexity of the Bayesian network (the longer the interval, the higher number of events that are being considered, the greater the complexity of the network). We will hence consider such parameter in the experimental Section showing how it affects the predictive power of the constructed model.

This said, in the following Section we will show how the described model applies to a real working machine.

IV. RESULTS

In order to carry out our experiments we had the opportunity of experimenting our methodology on a real industrial machine, a packaging machine. The packaging machine that has been taken into consideration is used to package food as well as medications and is composed of many different actuators and sensors, representing hence an interesting source of log data.

As described in the preceding Section, well-accepted methodologies exist in literature for the construction of Bayesian networks directly from data. Now, the problem moves to assessing the performance of the constructed model. A typical procedure in statistics for the verification of the validity of a model is known as cross-validation (cross-validation estimation or rotation) and works as follows. The k-fold cross-validation amounts to divide the total dataset into k parts (also called k-fold validation) and, at each step, the k-th part of the dataset is utilized as a test dataset (the remaining part is the training dataset). Such procedure avoids over fitting problems and asymmetric sampling (and therefore suffering from bias) of training datasets, which typically occur when a dataset is divide in only two parts.

In the case of interest we adopted a value of k equal to five, this meaning that in order to assess the validity of the model, four fifths of the available data were used in turn as a training set, and one fifth as a set of events to test the model. Given that the ultimate goal of the model is to verify the presence or not of a variable of interest (i.e., a log of interest) on the basis of the events that precede it, all experiments assess the success rate in predicting such presence, for the whole class of events under consideration and under two different time windows, fifty seconds and five hundred seconds.

Figures 3 and 4 report the results of our experimentation. Both graphs exhibit common trend: the use of a smaller number of samples leads to a lower accuracy. This phenomenon is intuitively easy to understand, the smaller the number of samples that are used during the training set entails that a smaller amount of information is utilized to build the model: it is therefore more difficult to obtain an accurate prediction of an event.
Comparing the two graphs we can also note that the larger the time window, the poorer the prediction quality in the presence of fewer training observations. Even this is easily understandable considering that the use of larger time windows leads to, in general, a looser coupling between successive observations and to the inclusion of a greater number of events in each observation.

Fig. 3. Number of correct classification when utilizing a fifty second interval.

Fig. 4. Number of correct classification when utilizing a five hundred second interval.

V. CONCLUSION

The aim of this work is that of demonstrating that techniques that are well understood and deployed in the computer entertainment industry [9]-[13]. In our case, opponent modeling through Bayesian networks, can be profitably applied into other fields, such as the industrial packaging one. With the hyper utilization of digital controlling systems and sensors, in fact, industrial machines are becoming more and more sources of great amounts of data. This data can be put to good use to improve the operational efficiency of such machines, not only ex post, i.e., after a failure has occurred, but also ex ante, i.e., before a failure occurs, studying more in depth the patterns that describe how such machines work.

REFERENCES


