What did I do Wrong in my MOBA Game?: Mining Patterns Discriminating Deviant Behaviours

Olivier Cavadenti, Victor Codocedo, Jean-François Boulicaut, Mehdi Kaytoue

What did I do Wrong in my MOBA Game?: Mining Patterns Discriminating Deviant Behaviours

Olivier Cavadenti
Actemium
F-42000, Saint-Étienne
France

Victor Codocedo
Université de Lyon
CNRS, INSA-Lyon, LIRIS
UMR5205, F-69621, France

Jean-François Boulicaut
Université de Lyon
CNRS, INSA-Lyon, LIRIS
UMR5205, F-69621, France

Mehdi Kaytoue
Université de Lyon
CNRS, INSA-Lyon, LIRIS
UMR5205, F-69621, France

***Abstract*—The success of electronic sports (eSports), where professional gamers participate in competitive leagues and tournaments, brings new challenges for the video game industry. Other than fun, games must be difficult and challenging for eSports professionals but still easy and enjoyable for amateurs. In this article, we consider *Multi-player Online Battle Arena* games (MOBA) and particularly, "Defense of the Ancients 2", commonly known simply as DOTA2. In this context, a challenge is to propose data analysis methods and metrics that help players to improve their skills. We design a data mining-based method that discovers strategic patterns from historical behavioral traces: Given a model encoding an expected way of playing (the norm), we are interested in patterns deviating from the norm that may explain a game outcome from which player can learn more efficient ways of playing. The method is formally introduced and shown to be adaptable to different scenarios. Finally, we provide an experimental evaluation over a dataset of $10,000$ behavioral game traces.**

***Keywords*—video game; moba game; contextualized trajectory; expert model; discriminative pattern mining**

I. Introduction

The video game industry has been dramatically expanding over the last few years, targeting several populations (new, casual and extreme players) and electronic devices (PC, consoles, smartphones, etc.). Indeed, over the last decade, the annual turnover generated by the electronic entertainment industry went beyond those of both cinema and music industries, making video games production one of the most lucrative business ever developed. In the meantime, a new scene called electronic sports (eSports) has emerged, where the most skilled players (also referred to as *gamers*) are hired by professional teams and supported by sponsors to take part in large international competitions [31] widely followed on live streaming platforms such as *Twitch.tv* [23].

The success of eSports has also brought also new challenges for developers and gamers alike. Although eSport games represent only a tiny proportion of all video games, they gather masses around an game title and its sponsors, thus coming with new and lucrative business models. For example, the *League of legends 2015 World Finals* counted 36 million unique viewers and awarded the winner with a US$ 1M prize [4]. From the point of view of the game publisher, a video game designed to be an eSport should not be only interesting to professional players, but it also requires to attract a fan base of amateur players. Fans should know the game, while understanding and appreciating the style of playing of professional gamers (mechanics and strategy [13]).

Consequently, video games should be designed to be interesting for a wide variety of skills, a task that have proven difficult for publishers. In a nutshell, we can use the famous board game Othello's slogan to describe the required learning curve of eSports: *"A minute to learn, a lifetime to master"*. A well known example of the difficulties involved in the design of such learning curve can be found in the game StarCraft 2 (Activision/Blizzard) which has been constantly patched since its release in 2012. David Kim, the game designer responsible of balancing the game states this difficulty as: *"Obviously we want both ends of the spectrum. On the one end the pro players, they want the game harder to show off their skill better. Other end is casual players, who want an easier game than Starcraft 2 is currently"* [1].

In order to support players in overcoming this learning curve, our goal in this article is to analyze player behaviors (movements and strategic choices) and compare them to expected behaviors under different circumstances. When a behavior is detected as abnormal (deviating from expected behaviors), we seek the context of such abnormality.

We consider Multiplayer Online Battle Arena games (MOBA) and particularly *Defense Of The Ancients 2*, commonly known as **DOTA2**. This game is a high level eSport strategy game with rich behavioral data containing both, mobility traces and strategic choices. Given a set of game logs (historical data), our goal is to discover strategic patterns that can help the player to understand his game decisions and improve his skill. For example, considering all the games of a player in the current season (or patch), we would like to answer questions such as: What are the particular choices (e.g. build orders, map trajectories, team composition, etc.) that discriminate victory? What did I do wrong in my game, that is, that deviates from the norm (how other similar gamers play) no matter the outcome?

Simple database queries can hardly answer these questions, as the number of possible queries/contexts is exponential w.r.t. such features. Hence, we have designed a data mining-based method that aims to discover patterns composed of strategic choices that explain strong deviations w.r.t a Reference Model.

A Reference Model represents a background of matches with common characteristics, e.g. all of them ending in victory. Reference Models can also be adapted to handle different

Contact: mehdi.kaytoue@insa-lyon.fr

pid	Traces	Global information
1	(t_1, -6973,-6428, buy_X),(t_2, -6742,-5290, ab_{A_1}),(t_3, -6440,-750, move) ,(t_4, -4801,5725, move) ,(t_5, -938,563, move), ...	winner=Yes, length=42,...
2	(t_1, -6980,-6440, buy_X),(t_2, -2077,-1550, move),(t_3, -6922,-6185, ab_{A_1}) ,(t_4, -620,-5963, move),(t_5, -6650,-6338, buy_Y), ...	winner=No, length=60,...
3	(t_1, -6928,-6436, buy_X),(t_2, -6050,-902, move),(t_3, -4825,-6130, move) ,(t_4, 10,5962, ab_{A_1}),(t_5, -4900,-6045, move), ...	winner=Yes, length=54,...
4	(t_1, -6806,-6334,buy_X),(t_2, 25,100, move),(t_3, 50,-160, ab_{A_1}) ,(t_4, -702,-5802, move),(t_5, -7105,-6593, buy_Z), ...	winner=No, length=35,...
5	(t_1, -6896,-6450, move),(t_2, -650,-6004, move),(t_3, 4890,-5986, ab_{A_1}) ,(t_4, -598,-5785, move),(t_5, -7445,-6985, ab_{B_2}), ...	winner=Yes, length=38,...

Table I: Simplified example of rough game traces

scenarios. Finally, we provide an experimental evaluation of pur approach over a dataset of $10,000$ behavioral game traces.

The paper is organized as follows. We introduce MOBA games principles and our problem setting in Section II. Basics from pattern mining are introduced in Section III which are necessary for understanding the our methodology discussed in Section IV. An experimental evaluation along with the description of the dataset used is presented in Section V. Finally, related work is detailed in Section VI.

II. Problem Settings

A. Multiplayer Online Battle Arenas (MOBAs)

MOBA is a specific video game type that mix aspects of real-time strategic game and role-play game. Whole in this article we focus on *Defense of the ancient 2* (DOTA2, [2]), there are several other well-known MOBA games that share the same principles with slight differences (e.g. *Heroes of the storm*, *League of Legends*, etc.). The choice of game was made because of data availability and has no impact on the methodological aspects developed hereafter.

A DOTA2 match is played on a map where two teams of five players battle each other in real time. Each team has to defend their own stronghold and destroy the opponent's one to win. Each player controls a *hero* which he moves through the map by providing instructions using mouse clicks. Furthermore, the player needs to *improve* the hero by collecting gold, new items, abilities, and by fighting heroes of the opposing team. Figure 1 displays the map of the game along with the initial influence zone for both teams. Team red (called *Dire*) and team blue (called *Radiant*) defend their strongholds located at the top right corner and bottom left corner of the map, respectively. Three lanes on the map (*top*, *mid*, *bot*) separate both teams with series of defensive towers. Players have well defined roles depending on the hero they initially picked from a current pool of 110 available heroes. For example, one role consists of defending and extending the influence zone in a specific lane while another role is to quickly switch lanes to attack by surprise. Like rugby, certain positions and moves are really important and can determine the outcome of a match (strong defense of towers and base, attacking players by surprise, make a concerted and synchronized attack on certain targets, etc.). Knowing that a team only sees controlled map zones, estimating enemy positions and triggering team fights at well-chosen times and in strategic areas is key to success.

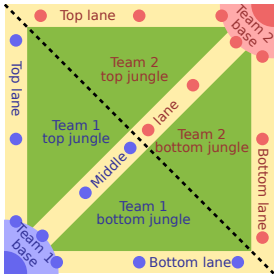


Figure 1: MOBA map

B. MOBA behavioural data

DOTA2 provides a log system based on replay files which players can recover after each match. In a nutshell, each match is completely recorded in a single file (called *replay*) for all of the 10 players involved. A replay contains all users' actions and positions as well as other system generated events. We will refer to the information of a single player's hero in a replay as a *trace*, however this notion will be later extended and formalized. Replays have been extensively used by developers,publishers and players alike to analyze behaviors and strategies that allow better understanding the in-game interactions of the entities involved (human and non-human controlled). Concerning DOTA2, replays contain among other pieces of information:

- **Positions**: For each hero expressed in x,y coordinates on the map where (0,0) is the center (Figure 1).
- **Builds**: Upgrades applied by a player to his hero. These should be carefully chosen as they are definitive and interdependent. A wrong build usually leads to poor performing heroes.
- **Items**: Items apply temporal boosts (of strength, defence or magic) or abilities to heroes. Items cost gold.
- **Gold/Experience time series**: Amount of gold or experience of a hero at a given time. Gold and experience are obtained through different in-game actions.
- **Global information**: Winner team, game length, team composition, players' ID, heroes picked, etc.

Example. Table I presents some trace examples where each row of the table is a series of actions made by one player during one game. The first action in the first row for player with *pid* 1 can be read as "At time t_1, located at coordinates $-6973, -6428$ bought item X."

C. Mining patterns explaining deviant behaviors

A new player learning how to properly use a given hero is likely to lose several matches if his team is not able to compensate for his lack of experience. A useful way for the new player to learn what he is doing wrong is looking at skillful players' replays. He will encounter that some strategies are different to what he has been doing so far and thus he will adopt those new strategies while dropping his own incorrect ones.

We can consider the last scenario in the context of data mining. Indeed, the new player is looking for frequent patterns in a *reference model* and contrasting them to frequent patterns in his own unsuccessful attempts to play the game. We will consider that a pattern that is prevalent in the latter set and not in the former *corresponds to what the new player is doing wrong*. An example (extracted from [3]) is given in Figure 2 depicting positions as points in the map in a given stage of

the game for a different MOBA. Zones densely populated can be considered as *normal behavior*. In section IV, we describe and illustrate this more sophisticated approach.

III. Background on Pattern Mining

In this section, we introduce the some basic notions of pattern mining through the use of an example. Firstly, we introduce the problem of *Frequent itemset mining* [5] considered the simplest form of frequent pattern mining. This choice was made for the sake of simplicity as it makes the method easier to follow, although generalizable to other kind of patterns. Secondly, we explain how to discover *Emerging patterns* that is, itemsets that discriminate a class label [17].

A. Mining frequent itemsets

Consider a set of transactions, where each transaction is composed of a set of items (in the original problem formulation, a set of items bought in a supermarket [5]). An arbitrary itemset is said to be frequent if the items it contains appear jointly in several transactions (i.e. more transactions than a user defined threshold). One can easily draw a parallel with MOBA: a transaction could be the set of items bought by a single player during a given game.

Definition 1 (Frequent itemset): Let $\mathcal{I}$ be a set of items. A transaction t is a set of items $t \subseteq \mathcal{I}$. A (transaction) database is a set of transactions $\mathcal{D} = \{t_1, t_2, ..., t_n\}$ with $t_i \subseteq \mathcal{I}, \forall\ i \in [1, n]$. Given an arbitrary itemset $X \subseteq D$, its support is given by the transactions that contain all items in X:

$$supp_{\mathcal{D}}(X) = |\{t \,|\, t \in \mathcal{D}, X \subseteq t\}|$$

The frequency of an itemset is the proportion of transactions in the database that contain the itemset:

$$freq_{\mathcal{D}}(X) = supp_{\mathcal{D}}(X)/|\mathcal{D}|$$

The *frequent itemset mining problem* consists in efficiently finding all the frequent itemsets in a transaction database $\mathcal{D}$, that is, with a frequency higher than a minimum frequency threshold θ: all $X \subseteq \mathcal{I}$ such that $freq_{\mathcal{D}}(X) > \sigma$. Equivalently, a minimum threshold on the support size is denoted by min_sup.

Note that several constraints other than minimal frequency can be used, such as minimal size of the itemset, minimal cost of an itemset (in the case where each item has a cost assigned), maximization of a function given by a mathematical model or several condensed representations that reduce the number of patterns [9]. Indeed, as there is an exponential number of itemsets ($2^{|\mathcal{I}|}$ in the worst case), a naive exploration of all itemsets is intractable while the mathematical properties of constraints can be taken into account for an efficient extraction. Algorithmic issues will be discussed in the Section IV.

t_{id}	transaction
t_1	$\{a, b, c\}$
t_2	$\{a, b, c\}$
t_3	$\{c\}$
t_4	$\{a, b, e\}$
t_5	$\{a, e\}$

Table II: Transaction database

Example. Consider a database where each transaction consists of the set of items bought by a single player in a single a game. Given five items $\mathcal{I} = \{a, b, c, d, e\}$, and the transaction database shown in the Table II we have: $supp_{\mathcal{D}}(\{a, b, c\}) = 2$, $supp_{\mathcal{D}}(\{a, b\}) = 3$, $freq_{\mathcal{D}}(\{a, b\}) = 0.6$ and $freq_{\mathcal{D}}(\{a, b, c\}) = 0.2$. If we set the minimal frequency threshold $\sigma = 0.3$, we have that $\{a, c\}$ is frequent while $\{a, b, c\}$ is not a frequent itemset.

B. Mining itemsets that distinguish a class label

Consider now that each transaction is provided with a single label, say positive or negative, for win or lose[1] as depicted in Table III. Patterns for which one of the labels is more prevalent give interesting and comprehensive hypotheses for that label. The main idea is to consider two transaction databases, made of positive and negative examples respectively and then to compute the support of an itemset in both databases separately. Intuitively, if their difference is high, the pattern may be a signature of one specific base. More formally, the mapping

$$class : \mathcal{D} \to \{+, -\}$$

associates to any transaction its class label (positive or negative). Then, we can define the positive and negative bases:

$$\mathcal{D}^+ = \{t \,|\, t \in \mathcal{D}, class(t) = +\} \text{ and } \mathcal{D}^- = \mathcal{D} \backslash \mathcal{D}^+$$

Thanks to this partitioning, we can compute a measure that expresses how an itemset discriminates a class label in favor of the other. Such an itemset is denominated an *emerging pattern* and the associated measure is called the *growth-rate* [17].

Definition 2 (Normalized growth-rate): Given databases $\mathcal{D}^+$ and $\mathcal{D}^-$, the normalized growth-rate of pattern $X \subseteq I$ is:

$$\phi(X) = \frac{|supp_{\mathcal{D}^+}(X)| - |supp_{\mathcal{D}^-}(X)|}{|supp_{\mathcal{D}^+}(X)| + |supp_{\mathcal{D}^-}(X)|}$$

where $supp_{\mathcal{D}^+}(X)$ (resp. $supp_{\mathcal{D}^-}(X)$) counts the number of transactions supporting X in the positive database (resp. negative). This means that if $\phi(X) = -1$ (resp. 1) then the itemset X appears only with a negative (resp. positive) label. On the other hand, if $\phi(X) = 0$ then X appears as much in the positive than in the negative database.

Example. We consider the same example as before, but we add as class attribute the outcome of the game, see Table III. The positive examples (resp. negative) correspond to a win (resp. loss). We have: $\phi(\{a\}) = (2-2)/(2+2) = 0$, $\phi(\{a, b\}) = (2-1)/(2+1) = 0.33$, $\phi(\{a, b, c\}) = (2-0)/(2+0) = 1$ and $\phi(\{e\}) = (0-2)/(0+2) = -1$. Consequently, choosing a, b and c can be interesting for a player as it discriminates victory and as it was played relatively often ($freq_{\mathcal{D}^+}(\{a, b, c\}) = 66.66\%$, $freq_{\mathcal{D}^-}(\{a, b, c\}) = 20\%$). In this scenario, we are interested in itemsets with *high* support and growth-rate.

t_{id}	$class(t_{id})$
t_1	$+$
t_2	$+$
t_3	$+$
t_4	$-$
t_5	$-$

Table III: Label of each transaction

[1] Multi-labelled data can also be handled directly, however we omit this model in this article for sake of simplicity.

Figure 2: Early game positions in *League of Legends*. Such positions totally differ in other contexts, see [3]

pid	Trajectory a	Description	Description	Outlier Score
1	$\langle 1,4,7,5,7,5,7\rangle$	$\{buy_X, buy_Y\}$	$\{ab_{A_1}, ab_{B_2}\}$	0.33
2	$\langle 1,2,3,5,3,5,3\rangle$	$\{buy_X, buy_Y\}$	$\{ab_{A_1}, ab_{B_2}\}$	0.33
3	$\langle 1,5,7,5,7,5\rangle$	$\{buy_X\}$	$\{ab_{A_1}, ab_{B_2}\}$	0.40
4	$\langle 1,2,3,5,3,6,3\rangle$	$\{buy_X, buy_Z\}$	$\{ab_{A_1}, ab_{C_2}\}$	0.66
5	$\langle 1,2,3,5,6,3\rangle$	$\{buy_Z\}$	$\{ab_{A_1}, ab_{C_2}\}$	0.80

Table IV: Contextualized trajectories: descriptions involves items bought (X, Y, Z) and abilities (A, B, C) taken at level 1 or 2.

IV. Method

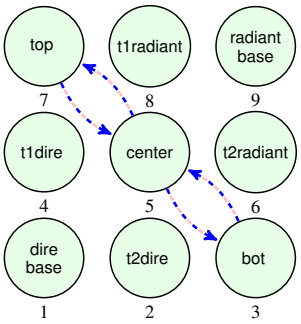
In the following, we describe a methodology for discovering descriptive patterns of deviant behaviors in four steps:

1) Contextualized Trajectory Generation
2) Reference Behavioral Model Construction
3) Trajectory Deviation Evaluation
4) Discriminant Pattern Mining

A. From traces to contextualized trajectories

As described in Section II-B, a trace for a hero is a sequence of events that describe its states and positions throughout the game, as well as other actions executed by the player controlling it. A trace also contains global information of the match such as its length, winning team, etc. However, when considering positioning it is rather hard to work with map coordinates, as those shown in the example of Table I, given the high number of points created for each hero on each game (a coordinate each 33 ms). Instead, we will define arbitrary points in the map which we will refer to as *points-of-interest* or POI. Series of coordinates are converted to POI sequences or *trajectories* by taking the nearest POI at each coordinate (ignoring POI repetitions), similarly to [18]. An example is shown in Figure 3 where the table contains 9 different POI with their corresponding coordinates and labels (arrows in the figure will be explained later).

Definition 3 (Player trace): Let $A = \{A_1, ..., A_n\}$ be a set attributes, where each attribute is either numerical or categorical (that is, takes either numbers or symbols as values). A record $r \in R$ is a n-tuple $r = (a_1, ..., a_n)$ with $a_i \in dom(A_i)$. The sequence $t = \langle r_1, ..., r_k\rangle$ with $r_i \in R$ is a player trace. A collection of player traces is denoted by $\mathcal{T}$. Global attributes are functions $f_i : \mathcal{T} \rightarrow Dom$ where Dom depends of the nature of the attribute, e.g., $outcome : \mathcal{T} \rightarrow \{win, lost\}$



poi_id	x,y	label
1	-7500,-7000	dire base
2	-600,-6000	t2dire
3	5000,-6000	bot
4	-6100,-850	t1dire
5	0,0	center
6	6200,-1600	t2adiant
7	-4700,6000	top
8	0,6000	t1radiant
9	7500,7000	radiant base

Figure 3: Simple expert model for DOTA2. Only edges with an important weight are displayed.

Player traces are considered in two axes: the way a hero moves in a set of point-of-interests (called the *trajectory*) and particular actions or properties that describe the player (called the *description*).

Definition 4 (Contextualized trajectory): Let $t \in \mathcal{T}$ be a player trace. The trajectory of t is a sequence of POI $trajectory(t) = \langle v_1, ..., v_n\rangle$ where $v_i \in V$ and V is a set of POI. The description of t is a set of items $description(t) \subseteq \mathcal{I}$, that are chosen/computed from the player trace. As such, a pair $(trajectory(t), description(t)), \forall t \in \mathcal{T}$ can be understood as a contextualized trajectory.

Example. Player traces in Table I have been transformed into contextualized trajectories in Table IV using the POI introduced in the Figure 3. For the sake of readability, descriptions are split in two columns: one with the objects purchased by the player, the other with the abilities acquired by the hero at specific experience levels.

B. Building a Reference Behavioral Model

The Reference Behavioral Model (or Reference Model) can be understood as the encoding of an expert opinion or more formally, as an expert model. Experts models are ideas, hypotheses or a priori knowledge of a given domain. In the most general case, an expert model is a function that returns a score for a given input to quantify an aspect of data (utility, outlier score, or a domain specific score). For example, the risk of soil erosion is assessed by a formula built by domain experts [20].

In the case of DOTA2 where each match is played in a single map containing well-known point of interests (POI) such as corners, tower, bases, among others, it is intuitive to express expert knowledge as a graph of POI transitions. Actually, in MOBA games it is well established that a hero's movement and positioning throughout a match is a very good indicator of the experience and skill of the player that controls it [3]. We will refer to this as the *mobility assumption*.

The *mobility assumption* allows fixing *allowed paths* in the POI graph that represent skilled (expert) player usual

transitions in the map. Obviously, the set of all *allowed paths* can be encoded as a Directed Graph which is not necessarily connected nor acyclic. Let us call this graph the Reference Behavioral Model.

Definition 5 (Reference Behavioral Model): A reference model is a graph $G = (V, E)$ with V a set of nodes (which represents POI) and $E \subseteq V \times V$ a set of edges (which represents *allowed* transitions between POI).

The reference model can be built manually by a game expert or automatically. In our case, we infer the reference model from a set of traces $\mathcal{T}_m$ that are chosen w.r.t. the data analysis goals. For example, one may select all traces of a specific hero during a season, or instead all those from a particular professional player. Then, we choose a set of POI (9 zones in our simple example in Figure 3) denoted as V. For each player trace $t \in \mathcal{T}_m$, we compute the trajectory $trajectory(t)$, i.e., the sequence of transitions between POIs (forbidding self loops). From the set of trajectories $\{trajectory(t), \forall t \in \mathcal{T}_m\}$, we compute the distribution of each direct POI transitions. For example, consider trajectory $\langle 5, 7, 5, 7, 5, 3, 5, 3, 5, 3, 2\rangle$, where transitions are distributed as follow: $(5, 7) : 2$, $(7, 5) : 2$, $(3, 5) : 2$, $(5, 3) : 3$ and $(3, 2) : 1$. Actually, these values are edge weights of the reference model G. Consequently, we can choose a *cut off* threshold to remove edges from the graph with low weights, e.g. consider a *cut-off* of 2, then we remove transition $(3, 2)$. This reference model corresponds to the graph represented with blue edges in Figure 3.

C. Measuring the deviation of a trajectory to the model

Given a Reference Model graph $G = (V, E)$, each trace t to be analyzed is confronted to it. This is done using a function that takes t and G, and outputs a quantification of the *outlierness* of t, denoted as μ. Outlierness of a trace is measured by calculating the proportion of transitions in $trajectory(t)$ that do not correspond to an edge in the Reference Model graph. We will use a $|V| \times |V|$ matrix representation of the Reference Model graph G denoted as M where $M(i, j) = 0$ if $(v_i, v_j) \in E$ and $M(i, j) = 1$ in the opposite case. A outlier scoring function for trace t, denoted by $\mu(t, M) \rightarrow [0, 1]$, returns 0 if the trajectory fully sticks to the Reference Model and 1 if it completely deviates from it.

Definition 6 (Outlier Score): Given a trace t such that $|trajectory(t)| > 1$, and a Reference Model matrix representation M, the outlier score is defined as:

$$\mu(t, M) = \frac{\sum_{i=0}^{i=|trajectory(t)|-1} M(t_i, t_{i+1})}{|trajectory(t)| - 1} \quad (1)$$

where $|.|$ counts the number of POI of the trajectory (its size) and $|trajectory(t)| > 1$ secure that the trajectory contains at least one transition.

Example. Consider the model $G = (V, E)$ given in Figure 3, and trajectories $t_1 = \langle 1, 4, 7, 5, 7, 5, 7\rangle$ and $t_2 = \langle 1, 2, 3, 5, 6, 3\rangle$. We can calculate their outlier scores as $\mu(t_1, M) = \frac{1+1+0+0+0+0}{7-1} = 2/6 = 0.33$ and $\mu(t_2, M) = \frac{1+1+0+1+1}{6-1} = 4/5 = 0.80$. Thus, the second sequence is more anomalous than the first sequence. At this step, we can compute the values for the outlier score for each trajectory in Table IV.

D. Describing Deviant Behaviors with Discriminant Patterns

The outlier scoring measure allows splitting the original set of traces $\mathcal{T}$ into two transaction databases, containing positive and negative traces (non deviant and deviant, respectively). However, when defining these databases we will only consider the *description* axis of traces, i.e. global information (such as winning or losing) and purchased items during the match for each hero. These elements will allow us building *explanations* of non-deviant and deviant behavior as we explain next.

Definition 7 (Positive and Negative Traces): Consider an outlier scoring measure μ, a set of player traces $\mathcal{T}$ and a threshold $\theta \in [0, 1]$ called *outlier threshold*, the positive and negative transaction databases are given by:

$$\mathcal{D}^+ = \{description(t) \mid t \in \mathcal{T}, \mu(t, M) \leq \theta\}$$
$$\mathcal{D}^- = \{description(t) \mid t \in \mathcal{T}, \mu(t, M) > \theta\}$$

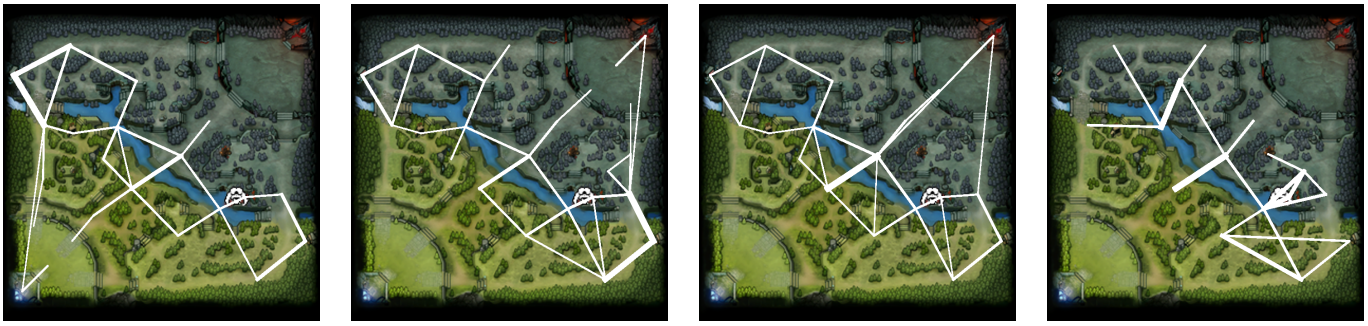
Once the positive and negative transaction databases are built, we can mine frequent itemsets as defined in Section III. Notice that we are interested in itemsets that are actually patterns of trace descriptions (a subset of $\mathcal{I}$) that frequently occurs in $\mathcal{T}$. Finally, to discriminate if a pattern is a hypothesis of non-deviant or deviant behaviors we will use the normalized growth-rate ϕ introduced in Section III-B.

Example. Let us consider traces in Table I and their trajectory representations in Table IV. In the following, we use the notation $d(t) = description(t), \forall t \in \mathcal{T}$. If we set $\theta = 0.5$ we have two databases of traces $\mathcal{D}^+ = \{d(t_1), d(t_2), d(t_3)\}$ and $\mathcal{D}^- = \{d(t_4), d(t_5)\}$. Using $min\_sup = 2$, we have the frequent patterns $X_1 = \{buy_X\}$, $X_2 = \{buy_Z\}$, $X_3 = \{buy_X, buy_Y\}$ with $supp(X_1) = 4$, $supp(X_2) = 2$, $supp(X_3) = 2$, resp. The normalized growth-rate for each pattern is:

$$\phi(\{buy_X\}) = (3 - 1)/(3 + 1) = 0.5$$
$$\phi(\{buy_Z\}) = (0 - 2)/(0 + 2) = -1$$
$$\phi(\{buy_X, buy_Y\}) = (2 - 0)/(2 + 0) = 1$$

As we can see, itemset $\{buy_X, buy_Y\}$ is a hypothesis of just positive or non-deviant traces. In other words, it is what we can consider a totally normal behavior. On the other hand, itemset $\{buy_Z\}$ describes only negative or deviant traces. We can consider it as a hypothesis of abnormal behaviors. Other examples are $\phi(\{ab_{A_1}, ab_{B_2}\}) = (2 - 0)/(2 - 0) = 1$ and $\phi(\{ab_{A_1}, ab_{C_2}\}) = (0 - 2)/(0 + 2) = -1$. We can say that players who deviate from the model bought object Z, took ability A at level 1 and ability C at level 2. Other players took the ability B at level 2 and bought object Y.

Algorithmic details. For mining frequent patterns, we use an efficient implementation of CHARM [35]. It extracts frequent closed itemsets. An itemset is closed if it has no superset with exactly the same support. Closed itemsets form a lossless condensed representation of frequent itemsets and that they maximize the growth-rate [28]. For each pattern, we also compute a $\mathcal{X}^2$ score (also maximized by closed itemsets) that allows one to measure how the distribution of the support of the itemset in the positive and negative bases is expected or not (introduced in [22]). Actually, in our experiments, we generally only consider the $\mathcal{X}^2$ score as statistically sound (even when we use the term *growth rate*).



(i) (ii) (iii) (iv)

Figure 4: Reference models for early game of: i) *Mirana* (radiant), (ii) *Mirana* (dire), (iii) *Pudge* (dire), (iv) *Invoker* (radiant)

V. Experimental study

This section reports an experimental study of the proposed approach. Firstly, we introduce the dataset used in our evaluation. Secondly, we show how different Reference Models can be built and to what extent. Thirdly, a quantitative study is presented that proves the feasibility and scalability of our approach. Finally, we provide a discussion on different scenarios supporting the discovery of interesting patterns and their application. All the experiments were carried out on an Intel Core i7 CPU 2.50 Ghz machine with 16 GB RAM. Code was written in JAVA using the CHARM implementation from the SPMF framework [21].

A. Dataset

Any action performed during a match is stored afterwards in a file (replay), allowing to re-watch the game at any time. We were kindly provided with a collection of 9.193 replays from the dotabank.com website. Different replay parsing tools are freely available. In our setting, we used *Skadistats Clarity 2* parser[2].

Only traces from non-anonymous players were considered (unknown *steam id*). For this reason we obtain a bit less than 10 traces per game in average, for a total of $90,366$ traces. We have a total of $77,112$ different players and thus, a skewed distribution of heroes played and $(playerID, hero)$ pairs (given in Figure 5 left and middle in the first row). These distributions shall influence the choice of scenarios for defining Reference Models. Heroes played the most ($\geq 2,000$ times) are *Mirana*, *Phantom Assassin*, *Invoker* and *Pudge*. On the other hand, the most frequent $(playerID, hero)$ pairs are: 135 times (*XXXX30*, *Invoker*), 56 times (*XXXX45*, *Lycan*), 55 times (*XXXX45*, *Furion*), 51 times (*XXXX06*, *Techies*)[3]. The hero with the highest number of traces is *Invoker* with $2,090$ in total, and 135 traces for a unique player. : this is why we choose to focus on these game traces in what follows.

B. Reference models

Because of the *mobility assumption* discussed in Section IV we will use positioning to determine our Reference Models. The Web site [4] presents a study which aggregates the positions of $10,000$ matches of *League Of Legends* into a single map. Three phases (early, middle and late game) can be easily recognized in the aggregated game, where players occupy different regions of the map. This study also discusses on different roles for heroes occupying different regions of the map, namely *carry* and *support* in the bottom lane, *mage* in the center, *tank/melee* in the middle lane, and *jungler* on the whole map. We focus on the early phase of the game as positioning is crucial at this step.

We constructed several Reference Models that we present in Figure 4. For each, we consider 33 well known POI in DOTA2 (bases, shops, center of the map, towers, ...). Models are built for some of the most played heroes in our replay collection. For example, *Invoker* was played $2,089$ times. After computing the POI trajectories, we obtain the graph with edges having a minimum (resp. maximum) weight of 1 (resp. $50,332$) and an average of 731 for dire and 867 for radiant. We use these thresholds to drop non important edges in the graph and present the model for *Invoker* (*radiant* team) in Figure 4 (iv). Note that the size of each edge is proportional to the weight. We follow the same process to present models for *Mirana* 4 (i) and (ii), and Pudge 4 (iii). It can be observed that hero *Mirana* focuses its moves in the top and bottom lanes (depending of her team). *Pudge* (in the *dire* team) mainly plays in the mid lane. Finally, *Invoker* (*radiant* team) is a *jungler* thus more present in the top and bottom jungles.

C. Quantitative experiments

Here we provide evidences of the computational feasibility of our approach. For this matter, we build the largest collections of traces that we could extract from the initial dataset and that were coherent with our approach. Firstly, the Reference Model was built by using the traces of any player using hero *Invoker*. We will denominate this set of traces as $\mathcal{T}_m$ where ($|\mathcal{T}_m| = 2,090$). Secondly, the set of traces to be analyzed was built from a unique player, namely the most active player in the collection using hero *Invoker*. We will denote this set as $\mathcal{T}$ ($|\mathcal{T}| = 135$). Finally, trace descriptions are built from a collection of 116 possible items to be purchased during a given match plus 68 different options for upgrade a hero's skills (build). Trajectories built from $\mathcal{T}$ contain between 16 and $1,205$ POI, with an average size of 476 POI per trajectory. The distribution of trajectory lengths is shown in Figure 5 (top right).

[2]https://github.com/skadistats/clarity

[3]In order to maintain players anonymity we obfuscate real game ID's maintaining just the last two digits. In this case, both codes finishing with 45 correspond to the same ID.

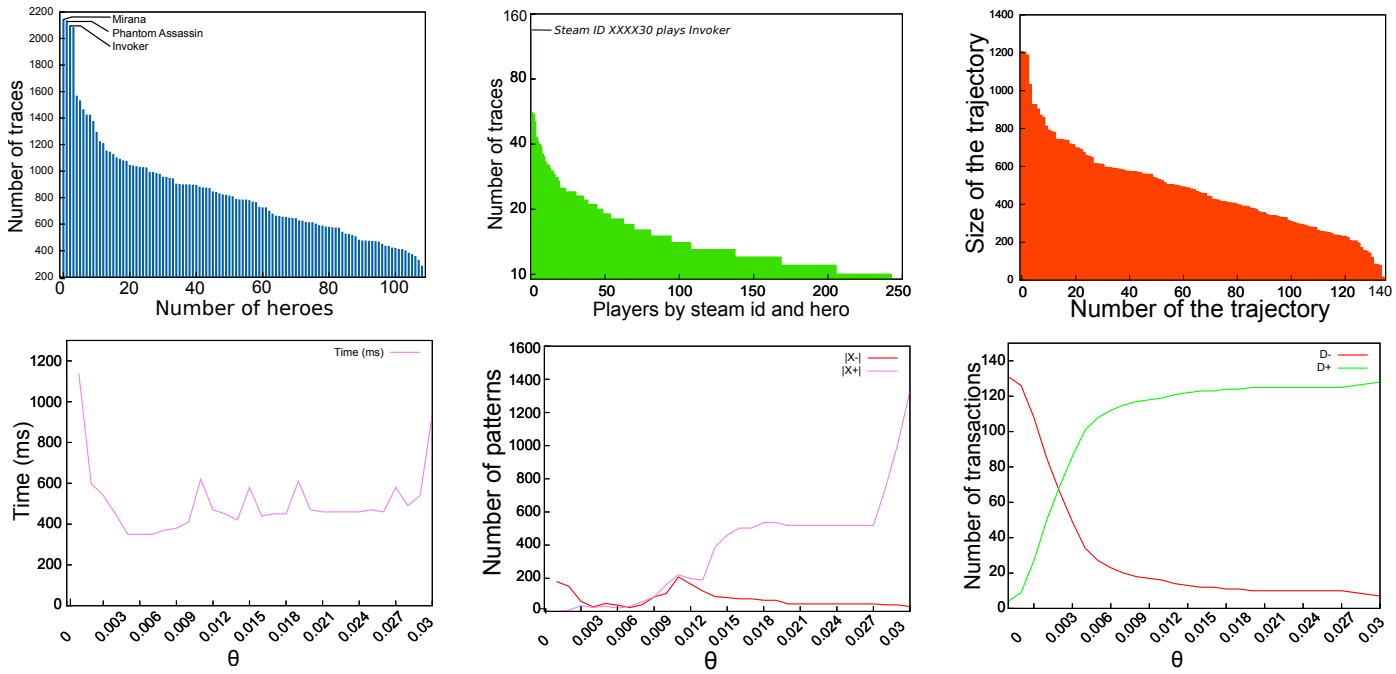


Figure 5: In the first row, several distributions on the DOTA2 dataset (traces by heroes, then by heroes/players, and sizes of behavior part). In the second row, we have several measures in terms of θ: time execution in milliseconds, number of patterns, and number of transactions.

To run the mining algorithm, we need to set several parameters: the minimum frequency σ (set to 1%), the graph weight cut off (set to the average of all weights in the graph) and the θ parameter that allows to split the transaction database into positive $\mathcal{T}^+$ and negative $\mathcal{T}^-$ examples. We report the number of transactions in both positive and negative databases, as well as the run time and number of patterns, for several values of θ. It follows that run times are very low, but the number of patterns can be sometimes too high to be analyzed by a Human. Our goal is to get few but accurate negative patterns: We choose $\theta = 0.008$. Negative patterns, denoted by X^- are those with a negative normalized discriminant measure. Figure 10 plots the resulting patterns, using a normalized $\mathcal{X}^2$ measure: Interestingly most of the patterns cover normal behaviours while a small proportion has a measure below -0.5. Note that we only reported pattern mining run times. Parsing each replay and processing them into an appropriate format can take more than one second per file. This is however done a single time, and thus we do not take it into account when reporting run times.

With the replay collection we possess, our methodology has no computational issues. Publishers that own massive replay collections can use parallelized versions of pattern mining algorithms in presence of scalability issues [26], [25].

D. Qualitative experiments

In DOTA2, each hero has a specific kind of game play. To win, one has to take choices, in a some order, which differs from one hero to another. Available objects for purchasing provide heroes with powers: some objects are better for some strategies than others. The same applies when choosing abilities. At each level of experience, a player has to choose between different upgrades for his hero. This choice influences the strategy of the player and efficiency of his hero. So called *build orders* are even shared and discussed on several community websites such as *Dotabuff.com*. Our purpose here is to provide patterns that inform the player about his choices and their capacity to discriminate the outcome of the game (first scenario) and his mobility behavior (second scenario).

1) Scenario 1 : Patterns that discriminate the game outcome: In this scenario, we label player traces with positive or negative classes depending on the $outcome(t)$ function. If $outcome(t) = \{win\}$ then $class(t) = \{+\}$, and if $outcome(t) = \{lost\}$ then $class(t) = \{-\}$. We seek to answer our first problem: what are the choices (purchased objects and hero upgrade options), that discriminate a victory or a defeat?

We consider the games traces of player *XXXX30* using hero *Invoker* (the most prevalent in our replay collection). In 135 player traces, 69 games resulted in defeat and 66 in a victory, which can be considered as a balanced overall outcome. Each of the 135 traces is described by the objects purchased ($\mathcal{I}$ is the set of all available objects). This leads to a database of 135 transactions on which we apply the CHARM algorithm with a minimal frequency threshold of $\sigma = 1\%$. 390 patterns are extracted in a negligible time. Figure 6 gives for each pattern its frequency in the whole database and growth-rate ϕ. Recall that the closer is the value of $\phi(X)$ to -1, the more discriminant for defeat is pattern X, and vice versa. We can observe patterns with extreme growth-rate values (1 and -1), that is, observed only in victorious or lost games. However, their frequency is very low, actually almost never observed. In contrast, the most balanced patterns appear more often (in 60%

of the games). Note that this observation was made in a similar scenario studying balance issues in *StarCraft 2* [8]. Table V presents five patterns that appear only in lost games which we compare with a list provided by *Dotabuff* of the most popular objects purchased by any given hero during the last year[4]. While patterns found through our approach contain objects *staff_of_wizardry*,*blade_mail*,*healing_salve* for hero *Invoker*, these are not very popular in the list of *Dotabuff*. We can infer fromt his that these five patterns are indeed not very common, probably because of their associated low winning chances.

We conducted a second experiment by taking into account the upgrades chosen at each level by player *XXXX30*. Thus, $\mathcal{I}$ is only composed of pairs $(hero\_level, ability\_level)$. Using the same parametrization than in the last experiment, we extracted 177 patterns from which we selected the pattern with the lowest growth-rate. which appears in only 4 traces. Again, we compare this pattern with the upgrade choices statistics collected by the Web site *Dotabuff* and shown in Figure 7. Each row denotes an upgrade of *Invoker* while columns denote the level of the hero. A cell thus counts the number of known matches in which a player chose to improve the correspondent upgrade a at the given level. Cells highlighted in green represent discovered patterns. They indicate that player *XXXX30* sticks quite well to common behavior in levels 4,7,8,17 and to a lesser extent, level 10. However, it seems that considering upgrade choices and purchased items may be not enough. Indeed, a player may follow well known build orders, but position his hero wrongly on the map. Moreover, match outcomes depend not only on the choices of a single player but also, on the team's. MOBA games are based on team cooperation and positions are very important on the map. In the next scenario, we use a Reference Model that captures positioning as a way to improve the quality of patterns.

#	X	$supp(X)$	$\phi(X)$
1	$\{tpscroll, force\_staff, blade\_mail\}$	3	-1.0
2	$\{tpscroll, staff\_of\_wizardry, blade\_mail\}$	3	-1.0
3	$\{tpscroll, healing\_salve, gloves, power\_treads\}$	3	-1.0
4	$\{boots, tpscroll, healing\_salve, blade\_mail\}$	3	-1.0
5	$\{tango, tpscroll, force\_staff, blade\_mail\}$	2	-1.0

Table V: Pattern discriminating defeat (Scenario 1)

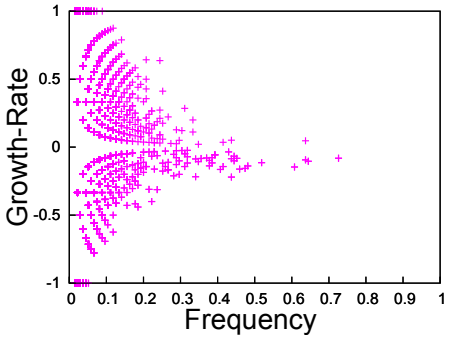


Figure 6: Pattern frequencies and growth-rates (Scenario 1)

[4]http://www.dotabuff.com/heroes/invoker/items?date=year

2) Scenario 2 : Patterns of traces that deviate/respect a reference model: In this scenario, we label each player trace with the help of the Reference Models illustrated in Figures 4. As we are studying games of the hero *Invoker*, we use its model for *dire* (resp. radiant) when labeling a trace played as *dire* (resp. *radiant*). Our goal is to describe sets of player traces that deviate the most from these models. The positive and negative transaction databases are build using $\theta = 0.008$ (as explained in Section V.C.). Like in Scenario 1, we build trace descriptions in terms of (i) purchased objects, (ii) chosen upgrades. In both cases, we have $|\mathcal{D}^-| = 27$ and $|\mathcal{D}^+| = 108$ while the model was computed on a set of $\mathcal{T}_m = 2,090$ traces.

With these settings, and for the case (i) only 39 patterns have a negative growth rate. Support and growth rate for all patterns is plotted in Figure 10. Discriminant patterns for deviant behaviors are shown in Figure 8. Objects $robe$, $sage\_mask$, $healing\_salve$ and $staff\_of\_wizardry$ are all rarely purchased items when using hero *Invoker*.

For case (ii), we extracted 91 patterns describing deviant behaviours. We illustrate some of them in Figure 9 which also depicts common upgrade statistics gathered by *Dotabuff*. In the figure we can observe that the player makes a mistake at level 9, by choosing the ability 3 instead of 2, again at level 12, by choosing upgrade 2 instead of 4, and again at level 14 and at level 16 where his chois is only shared by 4.4% of players. Thus, our method discriminates better the bad and infrequent moves.

VI. Related Work

The analysis of behavioral video game data is not new. However, for a long time it seems to have only provided a test bed for artificial intelligence techniques. For example, the real-time strategy game *StarCraft* has attracted much attention for the design of intelligent agents and even serves as a test bed for AI to compete through tournaments [27]. The problem of building an agent able to beat a Human is so hard that it is now an objective for both Facebook AI research and Google Deep Mind.

With the advent of massively multiplayer online-played games, the game industry got interested in analyzing these massive sets of historical data by means of visualization, machine learning and data mining techniques. This is one of the many facets of *video game analytics* aiming to enhance user experience and extending game lifetime [33]. There are several relevant tasks that demand massive data. Identifying imbalanced strategies in *StarCraft 2* thanks to pattern mining was recently studied [8]. It is also possible to predict who is playing thanks to keystrokes used by *Starcraft 2* players [34] and thus to recognize banned players with a new identity [11]. Without being exhaustive, we can also mention the tasks of detecting unexpected situations and bugs [33], cheaters [6], improving match making systems [32], [10], designing interactive player advice systems [15], understanding when to surrender in a MOBA [14], among others. Massive datasets can also be used in eSport analytics, e.g. [29], [24]. Our methodology is useful mainly for the task of understanding player behaviors, and addressed either to a player or to the game publisher. Several works considered the problem of player advising (recommendation) for MOBA and RTS games.

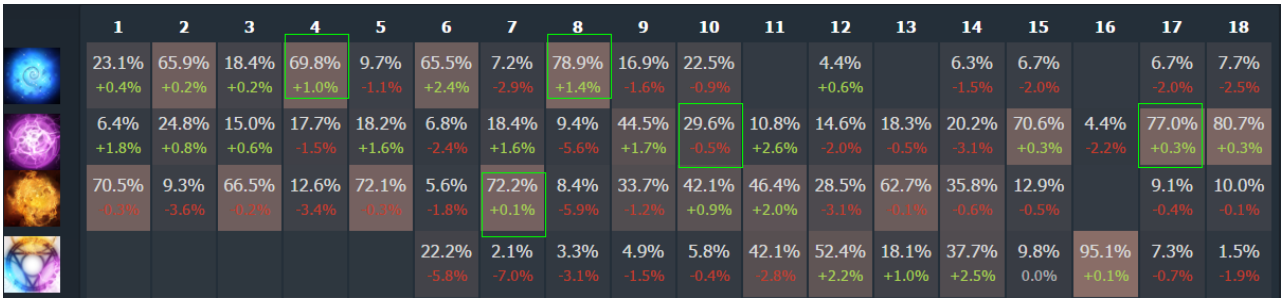


Figure 7: Comparing an itemset of skill choices with Dotabuff statistics (Scenario 1)

#	X	$supp_{\mathcal{D}}(X)$	$\phi(X)$
1	$\{tpscroll, phase\_boots, tango, robe\}$	3	-1.0
2	$\{tpscroll, phase\_boots, tango, sage\_mask\}$	3	-1.0
3	$\{tpscroll, boots, sage\_mask, force\_staff, healing\_salve\}$	5	-1.0
4	$\{tpscroll, phase\_boots, force\_staff, point\_booster, dust, scythe\_of\_vyse\}$	3	-1.0
5	$\{tpscroll, boots, sage\_mask, force\_staff, healing\_salve, staff\_of\_wizardry\}$	3	-1.0

Figure 8: Patterns discriminating deviant trajectories.

Silva et al. presented an approach to help novice players of *League of Legends* [16]. The system tracks the player actions and gives tips in real-time, e.g. when the hero's health is too low, his positioning is wrong, etc. They use domain knowledge from expert players to build a decision tree that is used in real-time. Chunha et al. purposes an advisory system to help players in a RTS game [15]. They formalized expert guides and also used them for helping the player to master the game more rapidly. In both cases, the systems rely on expert knowledge *a priori* and manually designed, which can be a tedious task. Our approach can use either an existing model or derive it automatically from selected game traces. It is however not designed to help the player in real time, but afterwards, when analyzing his games (or those of others).

Anomaly detection is a well-known task in data-mining (as surveyed in [12]). The task consists in detecting objects which strongly differ from others w.r.t. an outlier measure. There are a lot of applications, e.g. fraud detection in many domains. Contextualized anomaly detection [7] [30] [19] still considers a set of objects in which outliers are searched for, but also in which context the objects are outliers: a normal object may be an outlier in some subspace of the data. Our approach differs as we are neither looking for outliers or contextualized outliers, but contextual information (or hypotheses) that mostly characterize objects that deviate from a norm.

VII. Conclusion

With the recent developments of eSports, the video game industry faces new challenges for developing games attractive for both professional and amateur players. Gamers are in need of data analytic tools that highlight their strengths and weaknesses and help them during a long learning process. For that matter, discriminant pattern mining techniques provide intelligible explanations for several kinds of targets. We developed in this article a global methodology that enable to output strategic patterns explaining victory and explaining deviations from a reference behaviour that can be customized for various scenarios. The choice of a reference model, but also the different features to be encoded in the itemsets and the target given for each trace (game outcome, reference model, ...) makes it indeed customizable. Our choices in these article were mainly motivated by the available of data. It remains thus to further develop this first attempt, and to achieve a deeper experimental validation with several heroes, players, description encodings, but also to involve Human players in the loop. Another issue concerns the lack of data: it is hard to find a sufficient number of replays involving the same players and player/hero pairs.

Acknowledgments

This research has been partially funded by the French National Project FUI AAP 14 Tracaverre 2012-2016. The authors warmly thank Rob Jackson and the *Dotabank* website for providing us with a large collection of DOTA2 replays.

References

[1] David kim and the impossible balancing act of starcraft 2: Legacy of the void. http://www.pcgamesn.com/starcraft-ii/david-kim-and-the-impossible-balancing-act-of-starcraft-2-legacy-of-the-void. Accessed: 2016-07-29.

[2] Dota2, wikipedia. https://en.wikipedia.org/wiki/Dota_2. Accessed: 2016-06-06.

[3] Watch 10,000 league of legends games in 30 seconds. http://www.nytimes.com/interactive/2014/10/10/technology/league-of-legends-graphic.html?_r=0. Accessed: 2016-06-08.

[4] Worlds 2015 viewership. http://www.lolesports.com/en_US/articles/worlds-2015-viewership. Accessed: 2016-06-06.

[5] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *VLDB'94, Proceedings of 20th International Conference on Very Large Data Bases*, pages 487–499, 1994.

[6] Muhammad Aurangzeb Ahmad, Brian Keegan, Jaideep Srivastava, Dmitri Williams, and Noshir S. Contractor. Mining for gold farmers: Automatic detection of deviant players in mmogs. In *Proceedings of the 12th IEEE International Conference on Computational Science and Engineering, CSE 2009*, pages 340–345, 2009.

[7] Fabrizio Angiulli, Fabio Fassetti, and Luigi Palopoli. Detecting outlying properties of exceptional objects. *ACM Trans. Database Syst.*, 34(1):7:1–7:62, April 2009.

[8] G. Bosc, P. Tan, J. F. Boulicaut, C. Raissi, and M. Kaytoue. A pattern mining approach to study strategy balance in rts games. *IEEE Transactions on Computational Intelligence and AI in Games*, 2015.

[9] Jean-François Boulicaut and Baptiste Jeudy. Constraint-based data mining. In Oded Maimon and Lior Rokach, editors, *Data Mining and Knowledge Discovery Handbook, 2nd ed.*, pages 339–354. Springer, 2010.

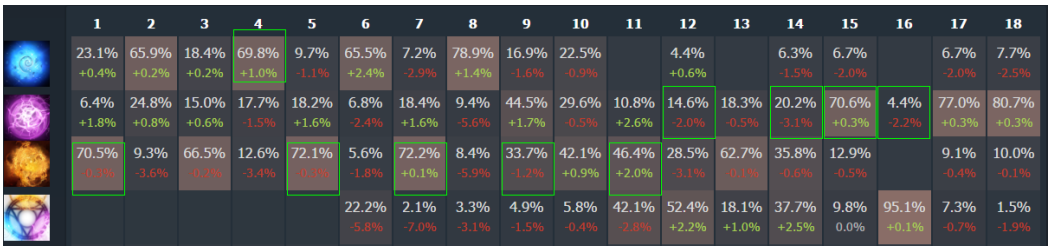


Figure 9: Comparing an itemset of skill choices with Dotabuff statistics (Scenario 2)

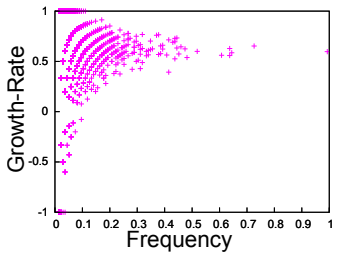


Figure 10: Pattern frequencies and growth-rates (Scenario 2)

[10] Alexandra Buchan and Jacqui Taylor. A qualitative exploration of factors affecting group cohesion and team play in multiplayer online battle arenas (mobas). *The Computer Games Journal*, pages 1–25, 2016.

[11] Olivier Cavadenti, Víctor Codocedo, Jean-François Boulicaut, and Mehdi Kaytoue. When cyberathletes conceal their game: Clustering confusion matrices to identify avatar aliases. In *IEEE International Conference on Data Science and Advanced Analytics (DSAA 2015)*, pages 1–10, 2015.

[12] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 41(3):15:1–15:58, July 2009.

[13] Gifford Cheung and Jeff Huang. Starcraft from the stands: understanding the game spectator. In *Proceedings of the International Conference on Human Factors in Computing Systems, CHI 2011*, pages 763–772, 2011.

[14] M. Claypool, J. Decelle, G. Hall, and L. O'Donnell. Surrender at 20? matchmaking in league of legends. In *IEEE Games Entertainment Media Conference (GEM)*, pages 1–4, 2015.

[15] Renato Luiz De Freitas Cunha, Marlos C. Machado, and Luiz Chaimowicz. Rtsmate: Towards an advice system for rts games. *Computers in Entertainment*, 12(1):1:1–1:20, February 2015.

[16] Victor do Nascimento Silva and Luiz Chaimowicz. A tutor agent for moba games. In *SBGames*, 2015.

[17] Guozhu Dong and Jinyan Li. Efficient mining of emerging patterns: Discovering trends and differences. In *SIGKDD'99*, pages 43–52, 1999.

[18] 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 *Games Media Entertainment (GEM), 2014 IEEE*, pages 1–8, Oct 2014.

[19] Lei Duan, Guanting Tang, Jian Pei, James Bailey, Akiko Campbell, and Changjie Tang. Mining outlying aspects on numeric data. *Data Mining and Knowledge Discovery*, 29(5):1116–1151, September 2015.

[20] F. Flouvat, J. Sanhes, C. Pasquier, N. Selmaoui, and J-F. Boulicaut. Improving pattern discovery relevancy by deriving constraints from expert models. In *ECAI'14*, pages 327–332, August 2014.

[21] Philippe Fournier-Viger, Antonio Gomariz, Ted Gueniche, Azadeh Soltani, Cheng-Wei Wu, and Vincent S. Tseng. SPMF: a java open-source pattern mining library. *Journal of Machine Learning Research*, 15(1):3389–3393, 2014.

[22] Tias Guns. Declarative pattern mining using constraint programming. *Constraints*, 20(4):492–493, 2015.

[23] Mehdi Kaytoue, Arlei Silva, Loïc Cerf, Wagner Meira Jr., and Chedy Raïssi. Watch me playing, i am a professional: a first study on video game live streaming. In *Proceedings of the 21st World Wide Web Conference, WWW 2012*, pages 1181–1188, 2012.

[24] Tobias Mahlmann Matthias Schubert, Anders Drachen. Esports analytics through encounter detection. In MIT Sloan, editor, *Proceedings of the MIT Sloan Sports Analytics Conference 2016*, 2016.

[25] S. Moens, E. Aksehirli, and B. Goethals. Frequent itemset mining for big data. In *Big Data, 2013 IEEE International Conference on*, pages 111–118, Oct 2013.

[26] Benjamin Negrevergne, Alexandre Termier, Marie-Christine Rousset, and Jean-FranÃ§ois Mehaut. ParaMiner: a Generic Pattern Mining Algorithm for Multi-Core Architectures. *Journal of Data Mining and Knowledge Discovery (DMKD)*, 2013.

[27] Santiago Ontañón, Gabriel Synnaeve, Alberto Uriarte, Florian Richoux, David Churchill, and Mike Preuss. A survey of real-time strategy game AI research and competition in starcraft. *IEEE Trans. Comput. Intellig. and AI in Games*, 5(4):293–311, 2013.

[28] Marc Plantevit and Bruno Crémilleux. Condensed representation of sequential patterns according to frequency-based measures. In *IDA'09*, pages 155–166, 2009.

[29] François Rioult, Jean-Philippe Metivier, Boris Helleu, Nicolas Scelles, and Christophe Durand. Mining Tracks of Competitive Video Games. In *AASRI Conference on Sports Engineering and Computer Science (SECS 2014)*, Londres, United Kingdom, 2014.

[30] Guanting Tang, James Bailey, Jian Pei, and Guozhu Dong. Mining multidimensional contextual outliers from categorical relational data. In *Proceedings of the 25th International Conference on Scientific and Statistical Database Management*, SSDBM, pages 43:1–43:4, New York, NY, USA, 2013. ACM.

[31] T. L. Taylor. *Raising the Stakes:E-Sports and the Professionalization of Computer Gaming*. MIT Press, 2012.

[32] Maxime Véron, Olivier Marin, and Sébastien Monnet. Matchmaking in multi-player on-line games: Studying user traces to improve the user experience. In *Proceedings of Network and Operating System Support on Digital Audio and Video Workshop*, NOSSDAV '14, pages 7:7–7:12, New York, NY, USA, 2014. ACM.

[33] Arthur Von Eschen. Machine learning and data mining in call of duty. In *European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD)*, LNCS 8724. Springer, 2014. an Industrial Invited Talk.

[34] Eddie Q. Yan, Jeff Huang, and Gifford K. Cheung. Masters of control: Behavioral patterns of simultaneous unit group manipulation in starcraft 2. In Bo Begole, Jinwoo Kim, Kori Inkpen, and Woontack Woo, editors, *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI 2015)*, pages 3711–3720. ACM, 2015.

[35] M.J. Zaki and C.-J. Hsiao. Efficient algorithms for mining closed itemsets and their lattice structure. *IEEE Transactions on Knowledge and Data Engineering*, 17(4):462–478, April 2005.