Mining the Lattice of Binary Classifiers for Identifying Duplicate Labels in Behavioral Data

Quentin Labernia, Victor Codocedo, Céline Robardet, Mehdi Kaytoue

Mining the Lattice of Binary Classifiers for Identifying Duplicate Labels in Behavioral Data

Quentin Labernia, Victor Codocedo, Céline Robardet, and Mehdi Kaytoue

Université de Lyon, CNRS, INSA-Lyon, LIRIS UMR5205, F-69621, France
Corresponding author: mehdi.kaytoue@insa-lyon.fr

Abstract. Analysis of behavioral data represents today a big issue, as so many domains generate huge quantity of activity and mobility traces. When traces are labeled by the user that generates it, models can be learned to accurately predict the user of an unknown trace. In online systems however, users may have several virtual identities, or duplicate labels. By ignoring them, the prediction accuracy drastically drops, as the set of all virtual identities of a single person is not known beforehand. In this article, we tackle this *duplicate labels identification problem*, and present an original approach that explores the lattice of binary classifiers. Each subset of labels is learned as the positive class against the others (the negative class), and constraints make possible to identify duplicate labels while pruning the search space. We experiment this original approach with data of the video game StarCraft 2 in the new context of Electronic Sports (eSport) with encouraging results.

Keywords: binary classification, label duplicate, data quality

1 Introduction

Sensors are nowadays part of our daily life, hidden in our cars, phones or watches and recording our position, speed, bio-signals, etc. Professional athletes may have position sensors in their shoes when playing soccer, or in their racket when playing tennis. Alone or combined together, these (mobile) devices generate rich behavioral data which, properly analyzed by means of data mining, machine learning and visualization techniques, can help answering several industrial challenges and inventing new services and applications for the common good.

In this article, we are interested in user identification techniques from behavioral data. Such methods are useful for security applications (fraud detection, targeted marketing, identity usurpation) and privacy preserving issues (e.g., for evaluating data anonymization techniques). There are indeed several domains for which it was established that the user who has generated a trace can be found through data analysis techniques: only a few points of interest in space and time uniquely identify a person [2]; typing patterns allow to recognize a person typing his password [7] or even playing a video game [9], etc.

However, especially on the Web, it often happens that a user has several identities (called *avatar aliases* in the remainder of this paper) and that the

mapping between users and aliases is not known beforehand. For example, one issue for targeted marketing applications is to identify that several Web cookies from different devices (tablet, smart phone, laptop, computer...) belong to the same individual [5]. In this work, we consider behavioral data from video games, as such data are extremely rich, freely available on the Web and without privacy preserving issues. Moreover, the video game industry is in crucial need of automatic methods to be able to detect cheaters, that is, users usurping an avatar [8]; as well as electronic sport structures seek to identify professional athletes hiding their tactics behind avatars when training on the Internet (more details are given in [1]).

It was shown in the context of online gaming by [9] that prediction models learned from particular typing patterns (keyboard usage while playing) can very accurately identify a player. Accuracy however strongly degrades in presence of avatar aliases: when individuals use several virtual identities the model hardly detects that two labels (or more) in the data describe the same user, that is, these two labels are avatar aliases. We refer in what follows to this problem as the *duplicate labels identification problem*: given a set of behavioral traces labeled by avatars, output groups of avatars that each denotes the same user.

Problem. Consider a set of users U and a set online identities L called avatars, the duplicate labels identification problem consists in discovering the mapping $f : U \rightarrow \wp(L)$ that corresponds to the set of identities assigned to each user. Note that $\wp(L)$ is the power set of L. The objective is to partition L into a set of label sets, each label block corresponding to an unknown yet unique user.

Recently, Cavadenti et al. presented an original approach to solve this problem [1]: it relies on mining the confusion matrix yielded by a supervised classifier, and exploiting the confusion the classifier has in presence of avatar aliases. Whereas this method has interesting results for identifying avatar aliases, it has a drawback we propose to address in this paper: it operates as a post-processing of a unique classification model and under exploits the power of classification algorithms by considering a static target value. Consequently, some classes, especially unbalanced ones, cannot be properly learned. Our intuition is that, when merged with their aliases, these classes should be properly learned.

Consequently, the approach we introduce takes advantage of the power of classification algorithms by recomputing the model for all target generalizations. We thus explore the lattice of binary classifiers, where each set of labels (duplicate candidate) is evaluated against all the other labels. For each new generated subset, the confusion matrix is compared to those of its subsets to tell us either (a) that the label subsets (i.e. positive examples) belongs to a same user, or (b) to prune the search space by stopping the enumeration of it supersets. To do so, we study the evolution of (i) the F1-measure and (ii) the distribution of the data objects in the positive/negative classes while generalizing label subsets.

2 Method

We propose an original method that considers the lattice of binary classifiers, where each element is a model learned from positive and negative examples that are respectively the instances of a subset of labels B and their complementary instances. This constitutes the search space of our problem and each binary classifier forms a potential group of avatar duplicate. We propose an efficient way to traverse the lattice of binary classifier to output the set of *duplicate label sets*.

2.1 The lattice of binary classifiers

Consider a set of traces T. Each trace is labeled by an avatar $label(t) = l$, with $t \in T, l \in L$. Consider now an arbitrary subset of labels $B \subseteq L$. Its corresponding binary classifier ρ_B is learned from positive and negative examples. The positive class is given by $B \subseteq L$ and the negative class by $\bar{B} = L \backslash B$. The data instances of the positive class are thus the set traces labeled by any $b \in B$, i.e. $\mathcal{I}_+(B) = \{t \in T \mid label(t) \in B\}$, while the instances of the negative class are the remaining ones, i.e. $\mathcal{I}_-(B) = T \setminus \mathcal{I}_+(B) = \{t \in T \mid label(t) \in \bar{B}\}$.

Reality \ Prediction		$+$	$-$
	C^{ρ_B}	$+$	$-$
	$+$	α_{++}	α_{+-}
	$-$	α_{-+}	α_{--}

Table 1. Confusion matrix of a binary classifier ρ_B.

Definition 1 Binary classifier and confusion matrix. *For any $B \subseteq L$, we define a classifier $\rho_B \colon T \longrightarrow \{+, -\}$. The confusion matrix C^{ρ_B} of this binary classifier is given in Table 1 where each score α_{ij}, with $i, j \in \{+, -\}$, counts the number of traces with class i classified as class j. Incidentally, it is easy to observe that α_{++} corresponds to true positives, α_{+-} to false negatives, α_{-+} to false positives and α_{--} to true negatives.*

Definition 2 Scores of a binary classifier. *Given a non-empty subset of labels $B \subseteq L$ and its binary classifier ρ_B. From the confusion matrix of the classification of traces, we compute two scores $\varphi_B \in [0, 1]$ and $p_B \in \mathbb{N}$ such that*

$$\varphi_B = \frac{2 \cdot \alpha_{++}}{(2 \cdot \alpha_{++}) + (\alpha_{+-}) + (\alpha_{-+})} \qquad p_B = (\alpha_{++}) + (\alpha_{+-}) + (\alpha_{-+})$$

Intuitively, φ_B corresponds to the F1-score or the harmonic mean of the precision and recall measures associated to the classification of traces. The score p_B counts the number of "similar traces" according to the model, i.e. traces that are well classified in the positive class as well as the confusion.

Definition 3 Lattice of binary classifiers. $\mathcal{L} = (\wp(L), \subseteq)$ *constitutes a Boolean lattice where each element B is associated to a classifier ρ_B.*

Our method relies on the study of the changes of the measures φ_B and p_B while enumerating the search space in a bottom up fashion (from singletons $B = \{l\}, l \in L$ towards $B = L$). The main idea is to find out maximal elements B (the most general) for which a set of constraints holds, such that one can assume that B is the set of avatar aliases of a single user $f(u) = B, u \in U$.

2.2 Constraining the set of binary classifiers

The general idea is to enumerate the lattice of binary classifiers and evaluate each element to assess if it represents a set of duplicate labels or not. As the number of elements is exponential w.r.t the number of labels, it is not acceptable to enumerate all of them. We introduce two constraints a classifier should respect so that it represents an actual set of duplicate labels. These constraints rely on the evolution of the $F1$-measure and the distribution of the data objects in the positive and negative classes within its downset. It implies an algorithm with a bottom-up enumeration (from $\emptyset$ to L) of the lattice, which is affordable as duplicate labels sets are rather small in most of the applications (we could not find an application where an individual has hundreds of different aliases).

The first constraint relies on the following intuition. If $E \subseteq L$ is set of duplicates, its binary classifier ρ_E should be more robust than any of its subsets. Rewriting $E = C \cup D$, if it exists a subset $C \subset E$ such that $\varphi_C \geq \varphi_E$, it means that merging together C and D must be avoided. More formally, a set $E \subseteq L$ is valid if it respects Constraint 1.

Constraint 1 *Consider a label set $E \subseteq L$ and its associated classifier ρ_E. E is a valid set of labels iff it respects the following constraint, $\forall C, D \subseteq E$, $E = C \cup D$, $\varphi_E \geqslant max(\varphi_C, \varphi_D)$.*

Note that φ_E is not monotone, but Constraint 1 is. However, this constraint alone is not sufficient. Indeed, we may have robust classifiers merged together which does not consider similar set of positive and negative examples, that is, better classifier but which does not merge duplicates labels. Still considering the sets $C, D \subseteq E$, we need another constraint to control that indeed the instances assigned to E are those assigned to C and D. To be more robust, rather than observing directly the set of instances, we show how the score p_E should be expressed in terms of p_C and p_D so that the classifier ρ_E is valid.

For all $B \subset L$, P_B is the set of traces well identified to be duplicates (true positives) along with the traces confused by the classifier ρ_B (false positives and false negatives). Two traces confused by the classifier can belong to the same duplicated labels, but sometimes they can belong to different labels (the classifier can make a mistake confusing two traces they are not as similar as it seems w.r.t. other traces). Thus, if the set $E = C \cup D$ is a set of duplicates, then we consider that we can reasonably write $P_E = (P_C \cup P_D) \cap \mathcal{E}$, where $\mathcal{E}$ is the set of traces that are confused by ρ_C or ρ_D but not by ρ_E. Intuitively, this property we defined considers that if the set E is a set of duplicates, the merging between two of its subsets C and D results in that P_E does not contain traces that are not in P_C or P_D, i.e., P_E is a subset of $(P_C \cup P_D)$. So, it gives that $|P_E| \leq |P_C \cup P_D|$, with $|P_E| = p_E$. The formula $|P_E| = |P_C| + |P_D| - |P_C \cap P_D|$ always holds and enables us to estimate the upper bound of the validity interval of p_E. Also, it is clear that the set P_E contains at least the elements within P_C or P_D: the lower bound of the validity interval is $|P_E| \geq \max(|P_C|, |P_D|)$.

Constraint 2 *We introduce* $\mu\colon \wp(P)^2 \longrightarrow \mathbb{N}$ *and* $\theta \in [0,1]$ *such that*

$$|P_C \cap P_D| = \mu(P_C, P_D) \cdot \theta \tag{1}$$

where θ *represents the overlapping factor between* P_C *and* P_D *given a arbitrary measure* μ*. We have then naturally the following constraint,* $\forall C, D \subset E$*,* $E = C \cup D$*,*

$$max(|P_C|, |P_D|) \leqslant |P_E| \leqslant |P_C| + |P_D| - \mu(P_C, P_D) \cdot \theta \tag{2}$$

$$\mu(P_C, P_D) \leqslant min(|P_C|, |P_D|) \tag{3}$$

We can choose for example $\mu(P_C, P_D) = min(|P_C|, |P_D|)$ *and* $\theta = min(\varphi_C, \varphi_D)$ *as a way to estimate its value.*

In Equation 1, rewriting the upper bounds allows us to control a minimal overlapping factor between the instances of C and D: when this overlapping factor is zero, it means that p_C and p_D do not have to overlap. On the flip side, the more the overlapping factor, the stronger the similarity constraint. In practice, it is required to set an increasing similarity constraint because experience has shown that the confusion of singleton classifier $\rho_{\{l\};l\in L}$ is less accurate than that of a classifier ρ_B with B a set of a higher cardinality. This is why we choose to express θ in function of φ.

In the end, we introduce the two mappings $C_1 : L \to \{true, false\}$ and $C_2 : L \to \{true, false\}$ telling if a subset of labels verifies respectively Constraint 1 and Constraint 2. A subset $B \subseteq L$ is valid iff $C_1(B) = true$ and $C_2(B) = true$.

2.3 Characterizing the result

Remembering that our goal is to discover the avatar set $f : U \to \wp(L)$ for any user $u \in U$. As we have no information about the users, our output shall be a set of label sets, each one belonging to a unique and unknown user. It means that we are looking at a partition of L. We construct our result as follows. Firstly, the set of all valid label sets is given by $\mathcal{V} = \{\ B \subseteq L \mid C_1(B) = true \wedge C_2(B) = true\}$.

The final result we are looking for is the set of maximal elements of $\mathcal{V}$ w.r.t. set inclusion: $\mathcal{R} = \{v \in \mathcal{V} |\ \nexists v' \in \mathcal{V}$ s.t. $v \subseteq v'\}$. $\mathcal{R}$ is an anti-chain of the lattice $(\wp(L), \subseteq)$, hence it is not necessarily a partition of the label set. $\mathcal{R}$ is here a tolerance relation, that is a set of sets that covers L but that can overlap (as opposed to a partition or equivalence relation where parts cannot overlap). We could as such enforce the fact that our result is a partition, e.g. by choosing some elements to remove, or adding constraints in the definition of the set $\mathcal{V}$. However, given our initial hypothesis and the choice of our constraints we should observe in practice that $\mathcal{R}$ is a partition as (i) only duplicates labels should be merged together (no intersecting parts), (ii) the singletons cannot be pruned by the constraints by definition (parts covering L). The only possible explanation for not having a partition is the case an avatar would be shared between several users: it shall be pruned early and not joined with another set respecting C_1 and C_2, we check this assumption in the experiments. Finally, note that it could be shown that $\mathcal{L} = ((\mathcal{V} \cup \emptyset \cup L), \subseteq)$ is a lattice, and that $\mathcal{R}$ is the anti-chain composed of all co-atoms.

2.4 Algorithm

The theoretical search space is the power set of labels $(\wp(L), \subseteq)$. We explore this lattice in a level-wise manner. Firstly, "*singleton*" classifiers $\rho_{\{l\}, l \in L}$ are generated. These singletons are pairwise combined to generate the next level, and so on. The new model is learned and tested to be valid or not. If the new classifier is valid, it will be used to generate the next level. If the classifier is not valid, the set is marked as irrelevant and none of its super sets shall be considered. The algorithm continues until there is no more possible merging. The worst-case complexity is $O(h \cdot 2^{|L|})$ where h depends on the classification method ρ used.

3 Experiments

This section reports an evaluation of our approach through both quantitative and qualitative experiments. As we study the video game StarCraft 2 and seek to identify groups of avatars belonging to the same player, we consider the same data of [1]. All experiments were performed on a 2,5 GHz Intel Core i7 with 8 GB main memory running OSX. The basic enumeration algorithm was coded in python. We used the Weka's implementations of several supervised classification methods to build the models $\rho_{B \subseteq L}$ ([4]).

3.1 Data and experimental settings

Replay collections. There are two collections of replays: $\mathcal{C}_1$ and $\mathcal{C}_2$. A replay is a game record which contains all actions made by the players, hence several behavioral traces each labeled by an avatar. The first collection is composed of the 955 games made by 171 expert players during the *2014 World Championship Series*. The rules of this tournament ensure us that there is no avatar aliases in this collection. We use this collection to build a ground truth, that is, inserting avatars aliases. The second collection is composed of $10,108$ one-versus-one games taken on a specialized website entailing $3,805$ players. We use this collection as real-world settings.

Features and classification models. We use the same features than two previous works to train the classifiers $\rho_{B \subseteq L}$. We briefly recall them and refer the interested reader to the work of [1] and [9]. The game allows the player to customize its usage of the keyboard in a limited way (change the function associated to keys 0 to 9). There are three ways to use a key given the current state of the game which implies 30 features counting the frequency of the different key usage by the player. A few other features were also added, such as the number of actions per minutes made by a player (up to 300 for expert players). When there is no avatar duplicates in the replay collection under study, it was indeed shown that these features allows to predict the avatar with an accuracy over 95%. Although our method is independent of the choice of a classification method, we report with several techniques (knn, J48, Multilayer Perceptron, Naive Bayes,

RandomForest and SMO) and their basic *Weka* implementations ([4]). In the end, we learn the model $\rho_{B,B\subseteq L}$ from a set of traces with positive (a set of avatars) and negative (the other avatars) examples and use a using 10-cross validation for building the confusion matrices. Then, the scores φ_B and p_B are computed. Note that a unique classification method is used for each full run of our algorithm (we do not "mix" classification models while enumerating the lattice).

Parameters. Concerning the avatar aliases identification problem, we consider three additional parameters also used the previous work of [1]. We consider only the τ first seconds of a game when computing the features as it was show to have an impact in the learning phase (the best being between 10 and 20 seconds). Second, there are labels with a very few instances which leads to bad average accuracy: we consider a trace in the dataset if its associated avatar has at least $\Theta \in \mathbb{N}$ examples, i.e. $\forall \ell \in L, |T_{\{\ell\}}| \geqslant \Theta$. It was previously shown that good predictions require $\Theta \geq 10$. Finally, a threshold $\Lambda \in [0;1]$ permits the selection of $R \in \mathcal{R}$ iff. $\varphi_R \geqslant \Lambda$. This cut on $\mathcal{R}$ is able to increase precision introduced now.

Ground-truth and evaluation. Given a set of labels, our method aims at finding the set of label sets $\mathcal{R}$ for which each part $R \in \mathcal{R}$ represents duplicate labels (avatar aliases of an unknown, yet unique user). As there exists no ground-truth (for privacy preserving issues the mapping between users and their avatars is not available), we build one. For that matter, we consider datasets built from the collection $\mathcal{C}_1$, where there is no duplicate labels. First, we choose the γ first labels that have the more instances. For each of such labels, we split their set of instances into several parts, each part being an avatar alias. In other words, we replace each of these γ labels by p new labels $(\ell_i)_{i\in[1;p]}$ and a family or *proportions* $(r_i)_{i\in[1;p]}$ such that $\forall i \in [1;p], |T_{\{\ell_i\}}| = \frac{r_i \cdot |T_{\{\ell\}}|}{\sum_{j\in[1;p]} r_j}$. We will use the following notation to explain a split: 1_1_2 means that each label l (with at least γ instances) is replaced by three labels: having respectively 25%, 25% and 50% of the instances of l (randomly distributed). This allow us to study balance issues.

To evaluate a result $\mathcal{R}$ w.r.t. the ground truth $\mathcal{G}$, we proceed as follows. The powerset of labels $\wp(L)$ is cut into positive and negative examples : $\mathcal{G}^+ = \{X \subseteq G, \forall G \in \mathcal{G}\}$ while $\mathcal{G}^- = \wp(G) \backslash \mathcal{G}^+$ and this is our ground truth. We operate similarly to partition the observed result : $\mathcal{R}^+ = \{X \subseteq R, \forall R \in \mathcal{R}\}$ while $\mathcal{R}^- = \wp(R) \backslash \mathcal{R}^+$. We can thus compare the ground truth w.r.t. the observed results : TP, FP and FN (resp. standing for true positives, false positives and false negative) can be defined as usual, as well as the classical evaluation metrics of precision, recall and F1-measure.

This evaluation roughly consists in comparing two partitions. However, $\mathcal{R}$ is not necessarily a partition but may be a tolerance. As explained before, this should not happen. Moreover, a null precision and recall penalize these cases in the experiments.

3.2 Experimental results

Parameter selection. Before giving our first results, we explain how the main parameters were chosen. We use the collection $\mathcal{C}_1$. The parameter γ is fixed to 10, $\Theta = 15$ and all classifiers were used (except SMO that has bad results). This experiment sets the value of parameter Λ as threshold over the elements of $\mathcal{R}$. This choice is based on the third quantile of false positive series, i.e., elements of $\mathcal{R}$ which are false positives. It ensures to drop 75% of these false positives. The true positive dropped elements rate can be shown on Figure 1 as a function of τ. The best result is for $\tau = 200$ matching with $\Lambda = 0.78$. Figure 2 illustrates the FP and TP distribution for this final setting. These two figures gives distributions that have been aggregated for all classifiers (without SMO).

Finally, four ways are explored to calculate θ as a function of φ_C, φ_D, $C, D \subseteq L$. These are: $\theta = 0$ (null), min, mean and max. The table on the right side shows aggregated results with parameters $\Gamma = 10$, $\Theta = 20$, $\tau = 200$ and all classifiers used. Although the results have a low mean/high variance (as aggregated results between good and bad classifiers), it clearly appears that $\theta = 0$ draws the best result. Some classification methods perform particularly well (we have indeed a 97%-precision and 61%-recall for the Naive Bayes classification algorithm with $\theta = 0$, while SMO is an outlier).

θ	Precision	Recall
null	0.76 ± 0.28	0.69 ± 0.28
min	0.50 ± 0.50	0.22 ± 0.28
mean	0.39 ± 0.49	0.17 ± 0.26
max	0.35 ± 0.48	0.16 ± 0.26

Run-time and memory analysis. Given the chosen parameters, we build several ground truth $\mathcal{G}$ with different proportions, some fully balanced other unbalanced. Recall that, e.g., (1_1_1_1) means that an original class was cut into 4 subclasses, each with the same amount of instances, while (1_4) means that a class was cut into two, with an unbalanced distribution of 20% vs. 80%. For all classification method we used, the number of generated nodes in the lattices of binary classifiers was less than a thousand which is insignificant w.r.t. the size of the theoretical search space 2^{171}. This means that our constraints allows very early pruning which makes the method possible in practice: Except for the method SMO, all run times were below 50 seconds.

Efficiency analysis. Still with the same parameters, we show some aspects on how efficient the method is. Figure 3 plots the precision and recall of our method when comparing the obtained results $\mathcal{R}$ with the ground truth $\mathcal{G}$. The main result is that the *Naive Bayes* implementation gives the best results, favors precision over recall, and is robust with unbalanced classes. Actually, as our method requires that the two constraints C_1 and C_2 are valid for any subset, our method favors precision in general. In an unreported experiment, we observed that the method favors recall if we set the restriction to the existence of only at least two different direct subsets that respect the two constraints. However, the goal in user identification is generally to favor precision.

Qualitative experiment. Until now, the goal of the experiments was to study how our method can retrieve a ground truth: traces of Collection $\mathcal{C}_1$ had no duplicate, we inserted some in a controlled way and observed how they can be

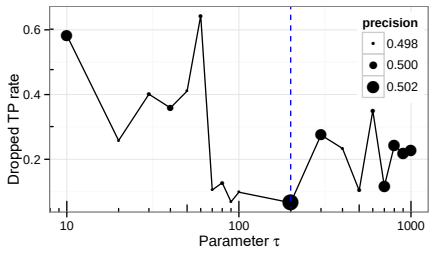

Fig. 1. % of dropped TP as a function of parameter τ when setting Λ as the third quantile of each FP series. The dashed line shows the best solution $\tau = 200$.

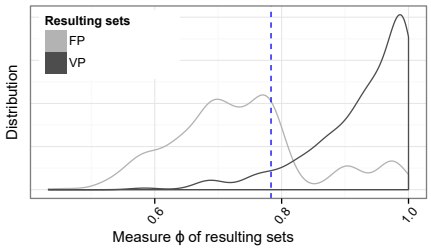

Fig. 2. TP and FP distribution for $\tau = 200$. The dashed line shows the third quantile of FP serie. This solution implies around only 6% of dropped TP.

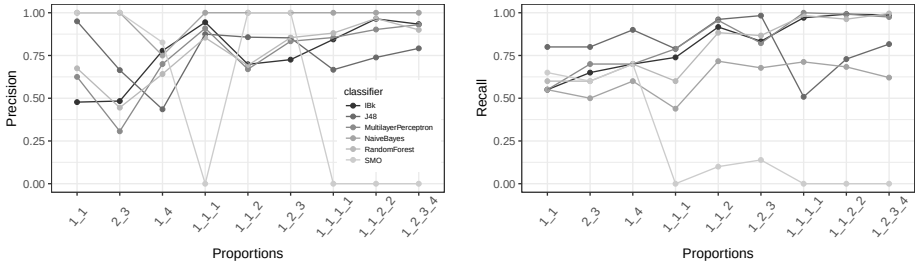

Fig. 3. Precision and recall with different ground truths (label distributions)

retrieved. In this last experiment, we run our method on the collection $\mathcal{C}_2$ which corresponds to real-world settings. We ran our algorithm with several parameters and report here only our first results. The settings were the following: we choose the Naive Bayes classification algorithm as it experimentally favors precision over recall, and behaves the better for imbalanced classes that we suspect to occur in $\mathcal{C}_2$. We set $\tau = 200$ and $\theta = mean(\varphi_C, \varphi_D)$ and $\Lambda = 0$ after several trials. We order the label sets of the result w.r.t. the robustness of their classifier, i.e. $\mathcal{R} = \langle R_1, ..., R_n \rangle$, where $\varphi_{R_i} \geq \varphi_{R_{i+1}}$, for $1 \leq i \leq n-1$. After a run of $1,017$ seconds, out of the $|L| = 58$ initial avatars labeling $|T| = 5,883$ traces, we find 7 aliases of size 2, i.e. $|\mathcal{R}| = 51$. For four of the found pairs, we have that the avatars share a same ID, so we are sure that it is the same user account: we omitted that information when building the dataset, and we kept only the avatar names. For example, we found the avatar names *EGStephanoRC*, a famous ex-professional player associated with the avatar name *lIlIlIlIlIII* (a name not recognizable on purpose). One pair of avatars share a same name (pro-player *LiquidHero*), but not the same ID: we can assume with high confidence that this is a true positive. Finally, two false positives, for which we cannot advance anything, but just assume that the same player is behind each of these pairs; these are actually the most interesting avatar aliases we are looking for.

4 Conclusion

In several online applications, a single user may have different virtual identities. When this mapping is not known, we face the duplicate label identification problem, whose resolution has applications in targeted marketing, online systems security, etc. Whereas this problem has similar goals as entity resolution techniques [3, 6], we treat it in a new way, taking into account the user behavior hidden in the data traces by building a model for each possible subset of label (in theory). Indeed, behavioral traces generated by the users can help building accurate prediction models that only confuse avatars of a same user. We proposed a method that takes advantage of this idea, by generating a binary classifier for each possible subset of labels. Using a bottom-up generation of the label sets, appropriate constraints ensure that we generate few accurate classifiers that each depict the same user. We experimented the implementation of our approach with behavioral data of a video game where players use several identities to play online. The results are encouraging although more experiments and comparisons remain to be done.

Acknowledgments. This work has been partially financed by the projects FUI AAP 14 Tracaverre 2012-2016, VELINNOV (ANR INOV 2012) and GRAISearch (FP7-PEOPLE-2013-IAPP).

References

1. Cavadenti, O., Codocedo, V., Boulicaut, J.F., Kaytoue, M.: When cyberathletes conceal their game : Clustering confusion matrices to identify avatar aliases. In: International Conference on Data Science and Advanced Analytics (DSAA) (2015)
2. De Montjoye, Y.A., Hidalgo, C.A., Verleysen, M., Blondel, V.D.: Unique in the crowd: The privacy bounds of human mobility. Nature Scientific reports 3(1376), 779–782 (2013)
3. Getoor, L., Machanavajjhala, A.: Entity resolution: Theory, practice & open challenges. PVLDB 5(12), 2018–2019 (2012)
4. Hall, M.A., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: an update. SIGKDD Explorations 11(1), 10–18 (2009), http://doi.acm.org/10.1145/1656274.1656278
5. ICDM Contest: Identify individual users across their digital devices. In: IEEE International Conference on data mining (2015)
6. Mugan, J., Chari, R., Hitt, L., McDermid, E., Sowell, M., Qu, Y., Coffman, T.: Entity resolution using inferred relationships and behavior. In: IEEE International Conference on Big Data. pp. 555–560 (2014)
7. Peacock, A., Ke, X., Wilkerson, M.: Typing patterns: A key to user identification. IEEE Security & Privacy 2(5), 40–47 (2004)
8. Von Eschen, A.: Machine learning and data mining in call of duty (invited talk). In: Eur. Conf. on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD) (2014)
9. Yan, E.Q., Huang, J., Cheung, G.K.: Masters of control: Behavioral patterns of simultaneous unit group manipulation in starcraft 2. In: 33rd Annual ACM Conf. on Human Factors in Computing Systems (CHI 2015). pp. 3711–3720. ACM (2015)