Article

First Steps towards Data-driven Adversarial Deduplication

Jose N. Paredes [1,†], Gerardo I. Simari [1,2,†,*] Maria Vanina Martinez [3,†] and Marcelo A. Falappa [1,†]

[1] Department of Computer Science and Engineering, Universidad Nacional del Sur (UNS) and Institute for Computer Science and Engineering (CONICET–UNS); {jose.paredes,gis,mfalappa}@cs.uns.edu.ar
[2] Arizona State University; gsimari@asu.edu
[3] Department of Computer Science, Universidad de Buenos Aires (UBA) and Institute for Computer Science (CONICET–UBA); mvmartinez@dc.uba.ar
* Correspondence: gis@cs.uns.edu.ar; Tel.: +54-291-459-5101 ext. 2628
† These authors contributed equally to this work.

Abstract: In traditional databases, the entity resolution problem (which is also known as deduplication), refers to the task of mapping multiple manifestations of virtual objects to its corresponding real-world entity. When addressing this problem, in both theory and practice, it is widely assumed that such sets of virtual object appear as the result of clerical errors, transliterations, missing or updated attributes, abbreviations, and so forth. In this paper, we address this problem under the assumption that this situation is caused by malicious actors operating in domains in which they do not wish to be identified, such as hacker forums and markets in which the participants are motivated to remain semi-anonymous (though they wish to keep their true identities secret, they find it useful for customers to identify their products and services). We are therefore in the presence of a different, even more challenging problem that we refer to as adversarial deduplication. In this paper, we study this problem via examples that arise from real-world data on malicious hacker forums and markets arising from collaborations with a cyber threat intelligence company focusing on understanding this kind of behavior. We argue that it is very difficult—if not impossible—to find ground truth data on which to build solutions to this problem, and develop a set of preliminary experiments based on training machine learning classifiers that leverage text analysis to detect potential cases of duplicate entities. Our results are encouraging as a first step towards building tools that human analysts can use to enhance their capabilities towards fighting cyber threats.

Keywords: Adversarial Deduplication; Machine Learning Classifiers; Cyber Threat Intelligence

1. Introduction and Motivation

The classical problem of *entity resolution*—or *deduplication*—in databases seeks to address situations in which seemingly distinct records are stored that actually refer to the same entity (object, person, place, etc.) in the real world. Typically, the goal is to identify and merge such records [1,2]; see Section 4 for a discussion of related work.

The characteristic that is overwhelmingly shared among these traditional approaches is that they assume that the existence of multiple records for the same real entity is the product of *involuntary* situations like simple typos during data entry procedures, ambiguity in attribute values such as transliterations and abbreviations, and inconsistency and incompleteness due to overspecification and underspecification (two addresses for the same person, or address completely missing), respectively, or evolving values such as address changes. In this paper, we are interested in situations in which these assumptions simply cannot be made because there are actors who may purposefully be taking actions towards hiding their identity behind multiple profiles. Take, for instance, the setting of malicious hacker forums on the dark/deep Web, in which participants seek to buy and sell different kinds of goods such

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2018 doi:10.20944/preprints201806.0425.v1

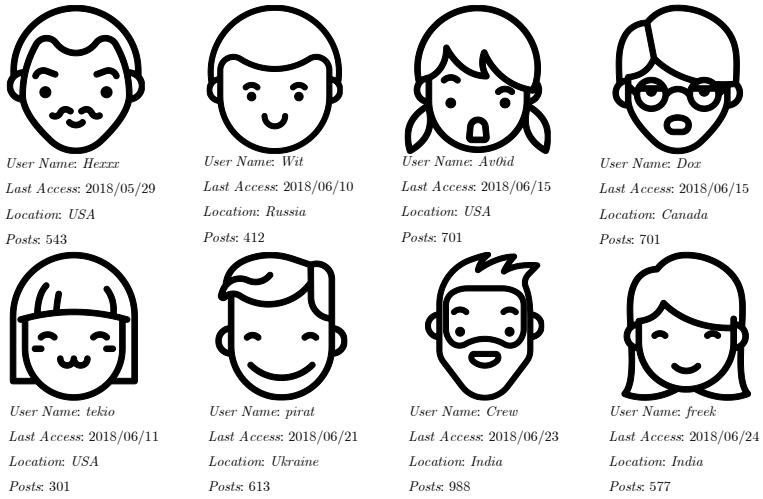

Figure 1. In real-world applications such as Dark Web forums and marketplaces, it may not always be clear who is behind user profiles; in particular, there may be two or more profiles that correspond to the same person.

as malware, passwords, credit card numbers, and other illicit materials. There is an interesting dynamic that arises among the participants in these forums and marketplaces: though of course they wish to remain anonymous–especially from government agents who may be watching—they on the other hand also wish to maintain their reputation within the community, and must therefore remain identifiable. The same actor typically operates using different profiles, but keeping certain characteristics constant; perhaps most importantly, they also leave *involuntary traces* behind that can be analyzed and leveraged by deduplication tools. We refer to this as the *adversarial deduplication/entity resolution problem*. Figure 1 illustrates this situation via a simple visualization—consider the problem of trying to determine clues that point to the conclusion that a given pair of faces might correspond to the same real-world user (or perhaps to the opposite conclusion).

The following is a simplified example of the kind of information that we can obtain from dark web forums and marketplaces.

Example 1.1. Consider the database schemas shown for the tables in Figure 2, where we have a table for forums, topics, and posts. In this paper, we focus on the latter since it is the main source of material that can be used towards identifying potential duplicates. ∎

This information is based on the system developed in [3] and CYR3CON[1] for cyber threat intelligence, which scrapes data from various social platforms, especially in the dark net and deep net. They collect and store information from hacker forum discussions and marketplaces offering products and services that focus on malicious hacking, such as sales of malware/exploits (including CVE numbers, which are identifiers given by the National Vulnerability Database [4,5]) and hacker forums (discussions regarding services and threats). The crawling and parsing of these sites yields time-varying data, since the system returns periodically to the same sites.

The rest of this paper is organized as follows: Section 2 presents our method to train machine learning classifiers to recognize posts made by users based on text analysis techniques, Section 3 presents our empirical evaluation consisting of two main experiments (a first phase consisting of a broad evaluation of different classifiers and hyperparameter settings, and a second phase analyzing the

[1] https://cyr3con.ai/Home

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2018 doi:10.20944/preprints201806.0425.v1

postId	postContent	postedDate	scrapedDate	forumsId	recordedDate	userId	topicId
11e9d297a29899f6a69c3da05391acaa	[u'', the only good way to work in computer security straight is to be a "black hat" then at some point (arrested a few to many times, you get a wife/family) you decide you can't do it any more. the other way is to go into sysadmin and at some point, once you have experience going into nothing but security. , u'', i see way to many people about trying to sell themselves as pen-testers and security consultants and know fuck all, don't be one of those people., u'']	11/10/2008	1/22/2017	56	11/10/2008	352820	481581
da1a4e396af0e1b87ba0d3a81b1e6277	your only recourse is using a second 3.60 or updated unit, or ps3 to download the games, then transfer them to your vita or qcma. and that only works for games you purchased, it isn't a gateway to piracy.	6/3/2017	6/5/2017	134	6/3/2017	75065	821237
92ac4e1e3cdb886080c6af2a528673eb	and i don^t particularly agree with touting this method either. why use wm_vsh_menu to call wmm functions to prepare the cfw when psnpatch does a fine job...?it^s not explained here but i assume it^s to run jb folder games to go online. however there is a good reason for psnpatch to remove cobra hooks stopping many jb folder games from working. bypassing this feature is not a smart idea for most users. i don^t recommend to do it. cfw users should always stick to iso & the psnpatch method. as to the title given the fact that the only users who require jb folders are cheaters & modders i wouldn^t actually hold my breath when telling them they will "never" get banned if they follow these steps... assuming that these steps will prevent a ban is not the same as knowing they will....	3/19/2017	6/23/2017	93	3/19/2017	1807	946623

forumsId	boardsName
56	null
134	vitahacks
93	thread locked

topicId	topicName
481581	[u'white hat hacker carrer']
821237	bought psn games on 3.63question (self.vitahacks)
946623	thread locked

Figure 2. Snippets of the *hackingPosts*, *forums*, and *topics* tables from our dataset.

effectiveness of the best two in finding pairs of entities), and Sections 4 and 5 discuss related work and conclusions, respectively.

2. Deduplication Leveraging Text-based Features

We now present a proposal for applying machine learning techniques towards solving adversarial deduplication problem; note that the general approach is not novel since it has been applied in several other problems such as malware identification and attribution, among others (cf. Section 4 for a discussion); however, this is to the best of our knowledge the first such proposal for this problem. Essentially, we wish to develop a *lightweight* method by which posts written by users can be automatically analyzed and *deduplication hypotheses* can be generated so that human analysts can step in to provide a more in-depth analysis. The workflow can be summarized in the following steps:

- Procure information from online discussions in forums and marketplaces; this is an ongoing effort that is generally carried out in a semi-automatic manner by specialists [6].
- Prepare the data by performing several cleaning processes (see below).
- Train one or more machine learning classifiers to recognize posts written by each user. For this step, we assume that posts made under different user names correspond to different users—we come back to this assumption when analyzing the results yielded in the testing phase.
- Apply the classifiers to *pairs* of new posts by users X and Y; if either X's classifier states that a post written by Y was written by X, or vice versa, then we generate a *deduplication hypothesis*.
- The set of deduplication hypotheses are sent to human analysts for further treatment.

Note that if we have no further information about the authors of posts, for n users we would have to analyze $\binom{n}{2}$ pairs to see if they actually correspond to the same user; for 50 users, this amounts to $1,225$, and for 100 we have $4,950$, which are already intractable numbers—clearly, scaling such a brute force analysis to larger numbers of users is impossible (cf. Figure 3). The general goal of our method is therefore to greatly reduce the set of pairs that humans must actually look at.

Analyzing text via n-grams. In order to extract basic elements from text, one common tool is the use of *n-gram*, which are can be defined in different ways; here, we adopt the commonly used definition of n-gram as a sequence of n characters. The advantage of using n-grams instead of directly analyzing a text is that typos, spelling variations, and other kinds of differences yield sets of n-grams that are closely related.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2018 doi:10.20944/preprints201806.0425.v1

Number of users n	Number of possible pairs $\binom{n}{2}$
50	1,225
100	4,950
150	11,175
200	19,900
500	124,750
1,000	499,500
2,000	1,999,000
5,000	12,497,500
10,000	49,995,000

Figure 3. Numbers of unordered pairs of users needed to be inspected by brute force analysis.

Example 2.1. Consider the word *software* and some common misspellings/intentional variations used by online communities: *sofware*, *softwarez*, and *sophwarez*. If we consider the 2-grams and 3-grams for each of these terms, we arrive at the following sets:

	2-grams	3-grams
software	so, of, ft, tw, wa, ar, re	sof, oft, ftw, twa, war, are
sofware	so, of, fw, wa, ar, re	sof, ofw, fwa, war, are
softwarez	so, of, ft, tw, wa, ar, re, ez	sof, oft, ftw, twa, war, are, rez
sophwarez	so, op, ph, hw, wa, ar, re, ez	sop, oph, phw, hwa, war, are, rez

Clearly, even the two most different variations (*software* and *sophwarez*) still share several n-grams. The value of n is a parameter to be tuned—a range within [3,7] has been found to work well in this kind of analysis [3]. ∎

Our working hypothesis is therefore that properly trained machine learning classifiers can detect unintentional traces left behind in the writing of people who are trying to hide behind multiple profiles in online forums and markets. In the next section we present the design and results of a set of experiments carried out as a preliminary attempt towards proving this hypothesis.

3. Empirical Evaluation

We adopted the following basic setup for the evaluation of our approach with real-world data, which was carried out in two main experiments (see below):

- *Dataset*: *posts* table, which contains 89,766 posts *users* table, which contains 128 users.
- *Data cleaning and preparation*: We removed HTML tags from posts using the *BeautifulSoup* tool[2], removed URLs, extra spaces, and strings that contained a combination of letters and numbers. Finally, we discarded posts that either contain less than 140 characters or any of the following strings "quote from:", "quote:", "wrote:", "originally posted by", "re:", or "begin pgp message". This yielded 40,453 clean posts corresponding to 54 users.
- *Feature generation*: We used the well-known TF-IDF (term frequency-inverse document frequency) technique to produce vectors of features based on n-grams, which essentially consists of assigning weights to features in such a way that they increase proportionally to the number of times it occurs in a document, and also takes into account the number of times the feature occurs in the whole corpus.
- *Classifiers*: Different standard machine learning approaches implemented with a standard Python library[3]:

[2] https://www.crummy.com/software/BeautifulSoup/
[3] http://scikit-learn.org/stable/

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2018 doi:10.20944/preprints201806.0425.v1

Table 1. Complete description of hyperparameter settings for all classifier instances

ID	Classifier Type	*max_df*	*n-grams*
SVC1	SVM–lin. kernel	0.02	[3,3]
SVC2	SVM–lin. kernel	0.03	[3,3]
SVC3	SVM–lin. kernel	0.02	[4,4]
SVC4	SVM–lin. kernel	0.01	[4,4]
SVC5	SVM–lin. kernel	0.03	[5,5]
SVC6	SVM–lin. kernel	0.03	[3,4]
SVC7	SVM–lin. kernel	0.06	[3,4]
SVC8	SVM–rbf kernel	0.01	[3,3]
SVC9	SVM–rbf kernel	0.05	[3,3]
SVC10	SVM–rbf kernel	0.03	[3,3]
SVC11	SVM–rbf kernel	0.05	[4,4]
SVC12	SVM–rbf kernel	0.08	[4,4]
SVC13	SVM–rbf kernel	0.05	[5,5]
SVC14	SVM–rbf kernel	0.03	[5,5]
DT1	Decision Tree	0.05	[3,3]
DT2	Decision Tree	0.05	[3,3]
DT3	Decision Tree	0.05	[3,3]
DT4	Decision Tree	0.02	[3,3]
DT5	Decision Tree	0.03	[3,3]
DT6	Decision Tree	0.03	[3,3]
DT7	Decision Tree	0.009	[3,3]
DT8	Decision Tree	0.03	[4,4]
DT9	Decision Tree	0.03	[4,4]
DT11	Decision Tree	0.02	[4,4]
DT12	Decision Tree	0.05	[4,4]
DT13	Decision Tree	0.01	[4,4]
DT14	Decision Tree	0.05	[5,5]
DT15	Decision Tree	0.03	[5,5]
DT16	Decision Tree	0.02	[5,5]
DT17	Decision Tree	0.01	[5,5]
DT18	Decision Tree	0.01	[5,5]

ID	Classifier Type	*max_df*	*n-grams*
MNB1	Multinom. BN	0.02	[3,3]
MNB2	Multinom. BN	0.01	[3,3]
MNB3	Multinom. BN	0.008	[3,3]
MNB4	Multinom. BN	0.007	[3,3]
MNB5	Multinom. BN	0.005	[3,3]
MNB6	Multinom. BN	0.005	[3,3]
MNB7	Multinom. BN	0.005	[4,4]
MNB8	Multinom. BN	0.005	[4,4]
MNB9	Multinom. BN	0.003	[5,5]
LR1	Log. Regression	0.03	[3,3]
LR2	Log. Regression	0.02	[3,3]
LR3	Log. Regression	0.03	[4,4]
LR4	Log. Regression	0.01	[4,4]
LR5	Log. Regression	0.03	[5,5]
LR6	Log. Regression	0.01	[5,5]
RF1	Random Forest	0.03	[3,3]
RF2	Random Forest	0.02	[3,3]
RF3	Random Forest	0.03	[4,4]
RF4	Random Forest	0.02	[4,4]
RF5	Random Forest	0.03	[5,5]
RF6	Random Forest	0.02	[5,5]

  - Decision Trees
  - Logistic Regression
  - Multinomial Bayesian Networks
  - Random Forests
  - Support Vector Machines

- *Hyperparameters*: We explored different values of two main hyperparameters: *max_df* and *n_gram range*. The former is a bound on the frequency with which a feature occurs in a post (essentially, as frequency increases the information content of a feature becomes lower), while the latter determines the length of the substrings into which the text is split. This yielded a set of 52 classifier instances (cf. Table 1).

  For some instances, we also applied a bound on the number of features taken into account. by the classifier(*max_features*):

  - DT2 is DT1 with *max_features* = 2,500
  - DT3 is DT1 with *max_features* = 2,000
  - DT9 is DT8 with *max_features* = 3,000
  - DT5 has *max_features* = 3,000
  - DT6 has *max_features* = 2,000
  - DT18 is DT17 with *max_features* = 5,000

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2018 doi:10.20944/preprints201806.0425.v1

	352792	20307	117723	43315	161133	282143	353596	352809	13585	146319
C_{352792}	**19**	0	0	1	1	0	0	0	0	1
C_{20307}	7	**17**	6	5	9	9	2	3	2	9
C_{117723}	5	5	**19**	6	4	3	4	6	1	9
C_{43315}	9	2	1	**17**	2	0	2	6	1	8
C_{161133}	5	4	4	7	**17**	2	5	0	0	13
C_{282143}	4	9	9	8	2	**14**	4	4	0	9
C_{353596}	12	7	11	16	8	3	**17**	8	6	14
C_{352809}	3	6	4	8	4	3	8	**10**	0	9
C_{13585}	1	2	9	4	4	7	9	2	**17**	9
C_{146319}	12	5	7	10	10	4	9	5	1	**17**

Figure 4. Example of a confusion matrix for Experiment 1; columns correspond to users and rows to classifiers. Each cell $M(i, j)$ contains the number of posts (out of 20) for which classifier C_i answered *yes* for a post actually authored by user u_j. The diagonal is highlighted in boldface.

- MNB6 is MNB5 with *max_features* = 3,000

This set of classifier instances was generated by means of manual exploration of the hyperparameters, looking for the combinations that had the most potential to yield high values for precision and recall.

3.1. Experiment 1: Broad evaluation of different classifiers and hyperparameter settings

The goal of the first set of experiments was to find the best-performing classifier instances among the 52 shown in Table 1; towards this end, we conducted three trials of the following set of steps:

- Choose 10 users at random.
- *Training phase*: For each user u_i, train a classifier C_i using between 200 and 500 sample posts written by them (positive examples, actual number of posts depended on availability) and the same number of posts written under other screen names (*presumed* negative examples).
- *Testing phase*: For each user, take 20 *test posts* that were not used to train any of the classifiers and query each resulting classifier. This yields a confusion matrix M where each row corresponds to a user and each column to a classifier; $M(i, j)$ contains the number of posts classified by classifier C_j as corresponding to user u_i. Note that a perfect confusion matrix in this case should have a value of 20 along the diagonal, and values of zero in all other cells. Figure 4 shows an actual instance of such a matrix.
- In order to evaluate the performance of each classifier, we make use of the following notions:
  - *True positives (tp)*: A test post written by user u_i is classified correctly by classifier C_i; the number of true positives for C_i is found at position $M(i, i)$.
  - *False positives (fp)*: A test post written by user u_i is classified incorrectly by a classifier $C_j \neq C_i$; the number of false positives for C_i can be found by calculating $\sum_{j \neq i} M(i, j)$.
  - *True negatives (tn)*: A test post written by user u_i is classified correctly by a classifier $C_j \neq C_i$; the number of true negatives for each classifier is simply $(10 - 1) \cdot 20 - \sum_{i \neq j} M(i, j) = 180 - \sum_{i \neq j} M(i, j)$.
  - *False negatives (fn)*: A test post written by user u_i is classified incorrectly by classifier C_i; the number of false negatives for C_i is calculated as $20 - M(i, i)$.

Based on the confusion matrix M and the above calculations, we can then derive:

$$precision(C_i) = \frac{tp(C_i)}{tp(C_i) + fp(C_i)}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2018 doi:10.20944/preprints201806.0425.v1

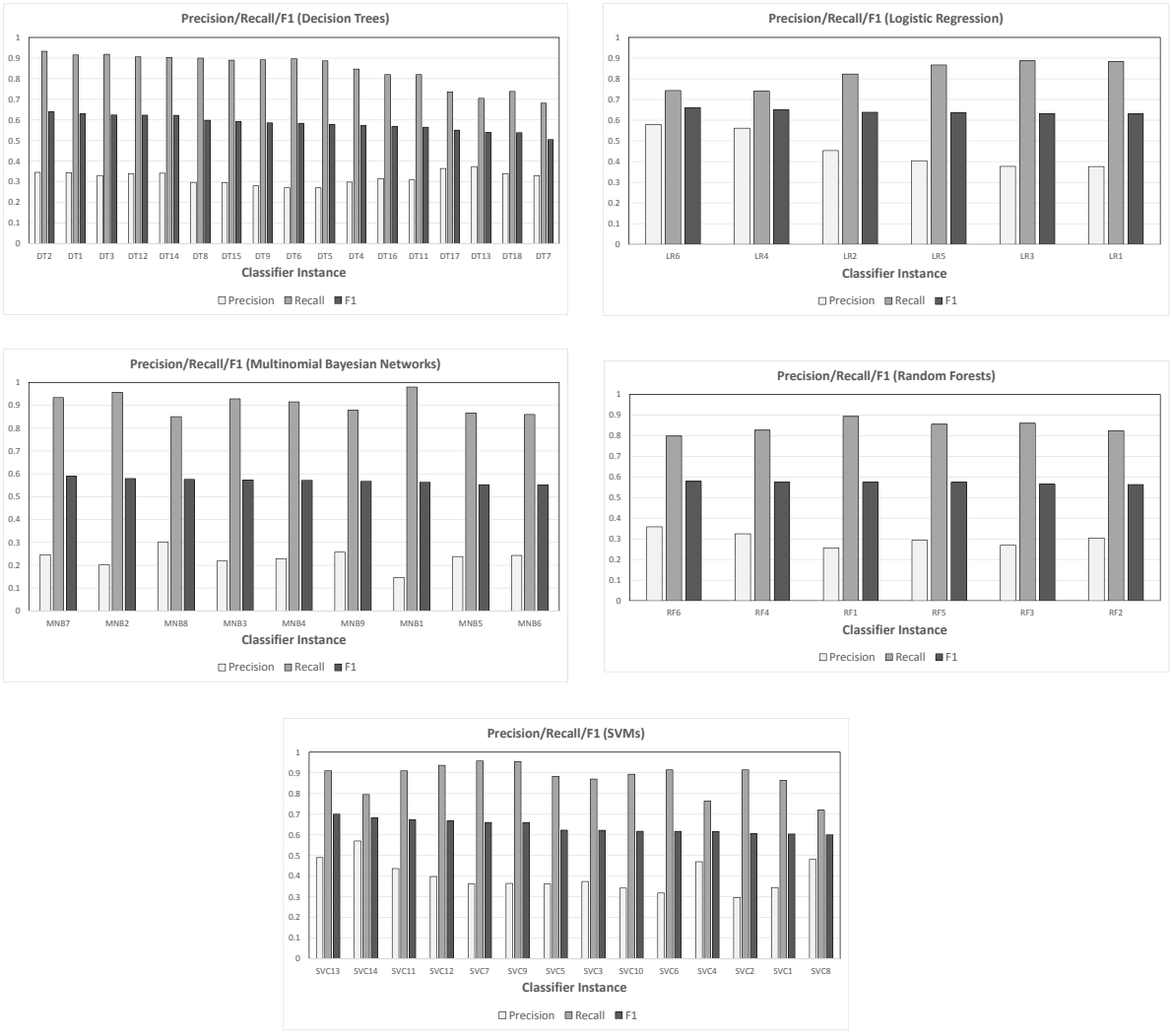

Figure 5. Precision, recall and F1 values for all classifiers evaluated in Experiment 1; in each graph, classifier instances are sorted in descending order of F1 measure.

and

$$recall(C_i) = \frac{tp(C_i)}{tp(C_i) + fn(C_i)},$$

which are standard metrics used to evaluate classifier performance; finally the harmonic mean of these two values, known as the F1 measure, is typically used as a good way to compare the performance of a set of classifiers.

Finally, we take the average of the three runs to obtain the final results, which we report next.

3.2. Results for Experiment 1

The results of this set of experiments are shown in Figure 5. Each graph shows the values obtained for precision, recall, and F1 measure, sorted by the latter; the typical tradeoff between precision and recall can be observed in each graph—though all instances have quite high values for recall, the classifiers that do best with respect to this measure do so at a high cost in precision (cf. MNB2, which boasts a recall of 0.96 but only 0.2 in precision. Figure 6 groups together the two best performers for each type of classifier.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2018 doi:10.20944/preprints201806.0425.v1

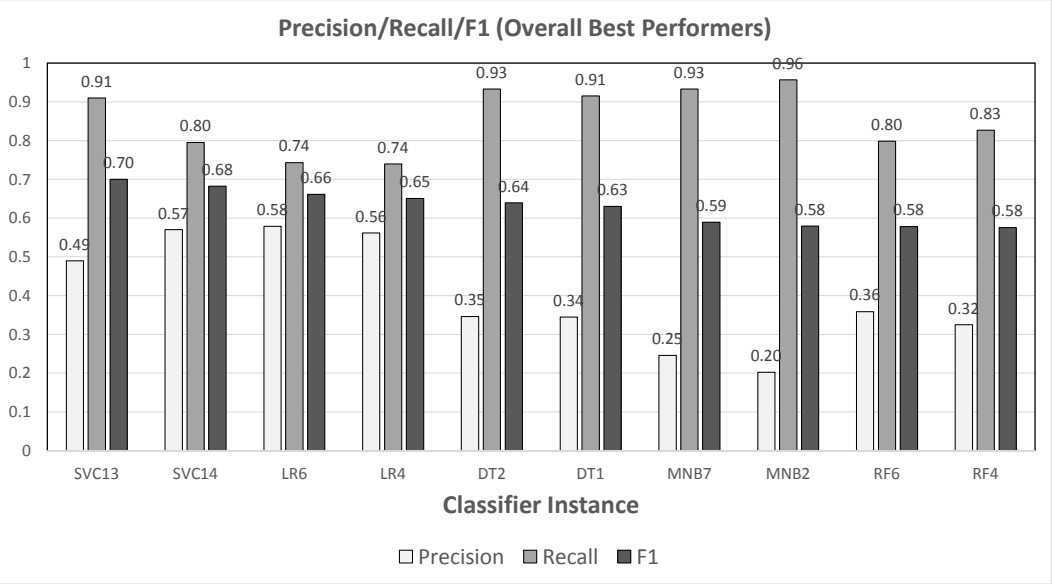

Figure 6. Best two performers from each chart in Figure 5, sorted by F1 measure.

It should be noted that choosing users uniformly at random to attribute authorship would succeed with probability 0.1 in this setting; therefore, the classifiers with the best precisions in Figure 6, which are the first four, are 4.9 to 5.8 times better than this baseline.

3.3. Experiment 2: Seeding known duplicates

We now select two classifiers from the first set of experiments and evaluate their capability to find duplicates. The choice was made among those shown in Figure 6 mostly by their performance with respect to F1 measure, which would yield SVC13 and SVC14; however, even though SVC14 performed slightly better than LR6, we chose the latter in order to favor diversity in classifiers.

Since, as discussed earlier, in this domain it is very difficult—or impossible—to obtain ground truth data, we engineer pairs of duplicates by simply dividing k users' posts into two sets and training independent classifiers with this data. Now, the definition of true/false positive and true/false negative depends on the type of pair being considered:

- For a pair of users that is known (or, rather, assumed) to be different:
  - *True negative*: classifier says no
  - *False positive*: classifier says yes

  True positives and false negatives are impossible in this case.
- For a pair of users that is known to correspond to a duplicate:
  - *True positive*: classifier says yes
  - *False negative*: classifier says no

  True negatives and false positives are impossible in this case.

Now, consider the task of choosing p pairs of users (u_i, uj) at random and presenting 20 test posts for each user to the classifier corresponding to the opposite user (so, u_i's posts to C_j, and vice versa). For $k = 2$, given that we have $54 + 2 = 56$ users (the original plus the duplicates), we have

$$\binom{54+2=56}{2} = 1,540$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2018 doi:10.20944/preprints201806.0425.v1

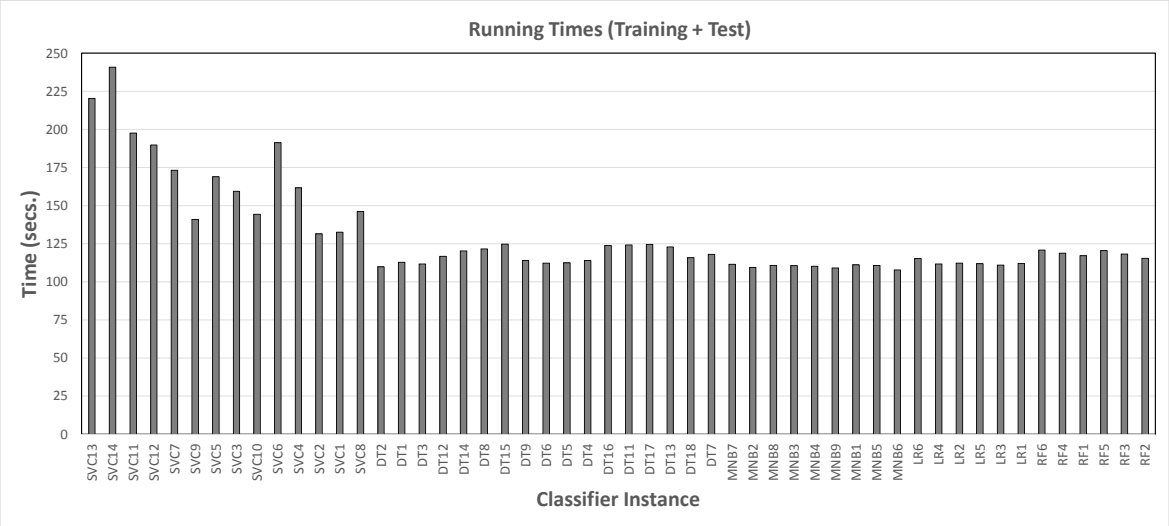

Figure 7. Running time taken to train and test each classifier instance from Experiment 1, sorted by their F1 measure in the charts from Figure 5.

possible pairs. In order to simulate a reasonable percentage of duplicates in the number of pairs tested, we conducted four runs varying $p \in \{20, 40, 60, 80\}$ and calculated the resulting precision, recall, and F1 values—the results are presented in Figure 8.

3.4. Results for Experiment 2

The first thing that is evident when analyzing the results is that both classifiers' performance with respect to precision is quite lower than before. This can be explained by the sharp increase in the number of users with respect to the previous experiment (from 10 to 56)—there are now many more opportunities for the classifiers to yield false positives. It should be noted, as before, that a classifier working by randomly answering *yes* one out of n times (for n users) would only have precision $1/56 \approx 0.0178$; therefore, the values of 0.125 (SVC13) and 0.267 (LR6) when $p = 20$ are 7.022 and 15 times higher, respectively. Furthermore, precision becomes lower as p increases, reaching 0.034 (SVC13) and 0.095 (LR6) for $p = 80$ (still 1.91 and 5.337 times better than chance, respectively). On the other hand, recall remains quite good for both classifiers, irrespective of the value of p, and is consistently better for SVC13 compared with LR6 (which has higher precision).

Finally, let's consider the last two steps of the workflow presented in Section 2, which involve the generation of *deduplication hypotheses*. Recall that what we are calling false positives in these experiments can actually be manifestations of unknown duplicates—pairs of user names that actually correspond to the same person. In order to deal with the relatively high incidence of such false positives that we saw above when we computed precision, we could set a threshold value t consisting of the number of times that posts presented to a different classifier trigger a positive response before we issue a deduplication hypothesis. For the same setting used in this experiment, we computed the number of deduplication hypotheses generated by each classifier (recall that the total number of unordered pairs is $1,540$ in this case):

- For SVC13, with $t = 10$, a total of $1,468$ hypotheses were generated (95.32% of the total), while for $t = 15$ the number was $1,411$ (91.62%).
- For LR6, with $t = 10$, 380 hypotheses were generated (24.67%), while for $t = 15$ the total was reduced to 111 (7.2%).

Two conclusions can be drawn from these results: first, the lower precision yielded by SVC13 in both Experiments 1 and 2 actually have farther-reaching causes than one might initially suspect by looking

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 26 June 2018 doi:10.20944/preprints201806.0425.v1

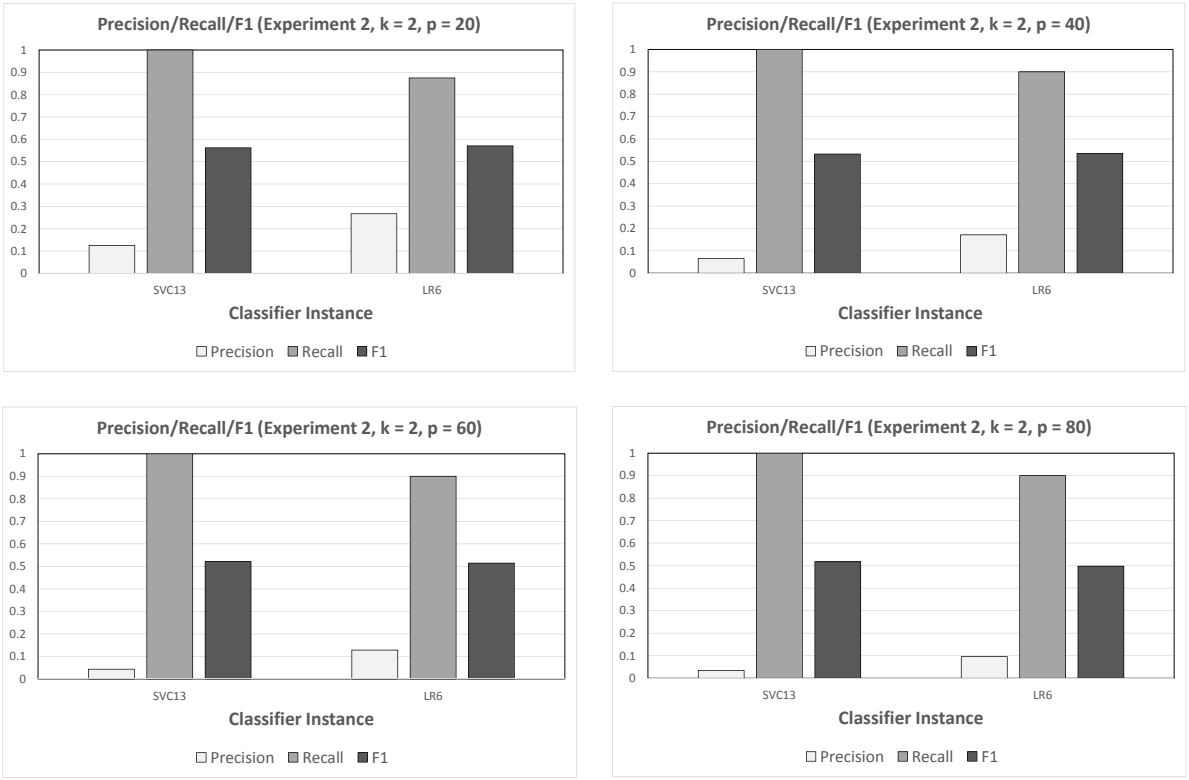

Figure 8. Evaluation of performance for the SVC13 and LR6 classifiers over 54 users plus 2 duplicates engineered by splitting two of the original users, varying the number of pairs p sampled at random (guaranteed to contain the duplicate pairs).

at the final quality measures, since clearly the false positive rate is quite high (as evidenced by the slight change in number of hypotheses when increasing the value of t); second, even though the performance of LR6 is also low in Experiment 2, it is nonetheless capable of reducing the number of pairs to inspect by about 75% for $t = 10$ (i.e., at least half of the test posts), and over 92% for $t = 15$ (i.e., at least three quarters of the test posts).

Example 3.1. The following posts are a few examples from a pair of users that, according to LR6, could have been written by the same person:

- Posts by user 353596:

  "lol i have that pasted to the front door of my officebut if we really want to get technical here...these are all "cheap" translations of binary. i will type up a full explanation from home tonight. i also dug up and old piece of software that i have that is wonderful for quick lookups. i will post that as well."

  "well "%path%" is the variable name. "path" is a term. it is always more important to learns terms and concepts rather than syntax that is specific to an environment.but yeah"

  "didn^t mean to burst your bubble about the script. i also saw some of the samples but none of them conviced me that the user thought they were really talking to a human being. most were prolly playing along like i do there are some msn ones also... anyone has visited they didn^t sign up for the forums. but that^s ok they can lurk for a while :whatsup:"
- Posts by user 352820:

> *"there are a few newish tools for but the issues with bt in the first place was poor implementation on behalf of the manufacturers, most of which were fixed."*
>
> *"well if talking about power friendly then a hd in lan enclosure would be best. you could build something that would consume less power but it would be more expensive, unless you are looking for and upwards."*
>
> *"the only good way to work in computer security straight is to be a "black hat" then at some point (arrested a few to many times, you get a wife/family) you decide you can't do it any more. the other way is to go into sysadmin and at some point, once you have experience going into nothing but security. , u", i see way to many people about trying to sell themselves as pen-testers and security consultants and know fuck all, don't be one of those people."*

Having identified this pair of users as a deduplication hypothesis, human expert analysts can then take a closer look into their posts, activities, and other data in order to confirm or reject it. ∎

4. Related work

Though there have been several approaches in the general areas of databases and security informatics that tackle problems similar to adversarial deduplication, there are important differences.

Most of the research carried out towards solving deduplication/entity resolution problems has been carried out in the databases community; traditional approaches involve leveraging pairwise similarity over entity attributes [7], but other promising proposals are based on so-called collective entity resolution [8], which exploits additional relational information in the data since references to different entities may co-occur. An example of such an approach is called *iterative blocking* [9], which is based on iteratively grouping together matching records in blocks; this allows the use of the information processed so far to inform further decisions. Other approaches take a similar techniques to the ones adopted here, in which machine learning classifiers are learned from examples and later used to determine whether or not an arbitrary pair of records are duplicates of each other based on a wide variety of features [8,10]. Recently, [11] has defined a declarative framework for entity resolution based on *matching dependencies* (MDs). This formalism, which was first introduced in [12,13], consists of declarative rules that generalize entity resolution tasks, eliminating duplicates via a matching process. These rules state that certain attribute values in relational tuples, under certain similarity conditions over possibly other attribute values in those tuples, must be made the same. The original semantics for MDs, defined in [13], was redefined and extended in [14].

As mentioned in the introduction, most of the approaches developed in the literature operate under the assumption that duplications are the effect of clerical errors in data entry, inherent ambiguity in names or other attributes, inconsistent abbreviations and formatting, etc., rather than the fact that the objects (users in this case) are explicitly trying to obfuscate their identities [8–10]. As a result of this, and the nature of the information that can be collected from the type of source we are looking at (forums and other sites in the Dark Web), the data set does not contain the relational information and structure needed for these proposals to be applied.

To the best of our knowledge, the work that is most related to ours is that of [15]; in that paper, the authors focus on the related problem of *authorship matching*, which seeks to validate whether two accounts having the *same username* on multiple Dark Web forums belong to the same person or not through writing style analysis (stylometry) and SVMs. An N two-way classification model, where N is the number of authors being tested, was trained with a set of ten active users (the ones with at least 400 posts and around 6000 words), divided into two parts, each of which contains half the posts of each user. In a validation phase, the best parameters for the model where found, in order to test it against the forum post of another set of accounts in a different Dark Web forum having the same username as the ones used in the validation phase. Using different types of similarity functions to evaluate the performance, their approach yields around 80% accuracy. The authorship matching problem is based on the assumption that there is a tendency in users to adopt similar usernames f or their accounts in

different sites. Though the results are promising, this assumption is quite strong and is not always valid in adversarial settings where users try to obfuscate their identities. In such scenarios, looking for equal or syntactically similar users names does not seem to be reasonable.

Stylometry is, historically, one of the more widely used methods for authorship analysis; it studies the personal characteristics of individuals' writing style—a survey of recent approaches to the authorship problem can be found in [16]. For instance, in [17], the authors apply stylometry techniques towards attributing authorship of instant messages, making use of features such as frequency distributions for characters, words, emoticons, function words, short words, abbreviations, and punctuation, average word length and average words per sentence, whether or not the message contains a greeting/farewell, and spelling/grammatical errors. It is possible that extending our approach with a deeper analysis of features such as these for specific forums and marketplaces will help us improve the accuracy of our methods. One of the problems with the application of such technique(s) to the kind of data we seek to analyze is the length of the texts, which are considerably short and therefore accuracy of identification (classification) suffers. Not surprisingly, even just a few short sentences can carry a great deal of information that a human analyst can use for identification, but that information goes beyond the style of the writing. Additional techniques that are capable of exploiting specific domain information are needed; the work of [18] suggests that contextual analysis and tolerance to uncertainty are especially needed for the identification of individuals in social media forensics.

Finally, the same kind of analysis based on applying machine learning techniques like classifiers and clustering algorithms has been successfully adopted in other problems related to cyber security, such as identification of product offerings in malicious hacker markets [19], at-risk system identification [20] and exploit prediction [21]. Though these are difficult problems, their advantage over adversarial deduplication is the availability of some kind of ground truth that can be used to evaluate and tune proposed solutions.

5. Conclusions and Future Work

In this paper we tackle the so-called *adversarial deduplication* problem, which seeks to identify pairs of users who are actively trying to hide their identity by creating multiple profiles. We argue that this problem is fundamentally different from the closely related to the traditional entity resolution or deduplication problem in databases since this problem is assumed to arise as a consequence of unintentional errors. We focus on the cyber-security setting of malicious hacker forums and marketplaces on the dark web, where such intentional obfuscation is the norm.

As a first step towards developing tools to address this problem, we proposed the use of machine learning classifiers trained to identify text-based features, and designed a set of experiments to evaluate their effectiveness. Our preliminary results are promising in that reasonably high precision and recall can be obtained in an initial training and evaluation phase with few users; in a second phase with over 50 users, the precision becomes much lower due to the incidence of false positives—however, since the overall goal of the approach is to create deduplication hypotheses that are then passed on to human analysts for further review, an additional threshold parameter can be applied to manage the number generated hypotheses.

Future work includes carrying out further experiments with other datasets, and identifying/learning other features (potentially quite complex) to incorporate them to the task aiming to capture the context in which these post are issued, such as topics in which users post, co-occurrence with other users, abnormality in learned behaviors, etc.

Author Contributions: Conceptualization, Jose Paredes, Gerardo Simari, Maria Vanina Martinez and Marcelo Falappa; Data curation, Jose Paredes; Formal analysis, Gerardo Simari and Maria Vanina Martinez; Funding acquisition, Marcelo Falappa; Methodology, Jose Paredes, Gerardo Simari and Maria Vanina Martinez; Project administration, Marcelo Falappa; Resources, Marcelo Falappa; Software, Jose Paredes; Supervision, Gerardo Simari and Marcelo Falappa; Validation, Gerardo Simari and Maria Vanina Martinez; Writing – original draft, Gerardo Simari and Maria Vanina Martinez; Writing – review & editing, Gerardo Simari, Maria Vanina Martinez and Marcelo Falappa.

Funding: This research was funded by Universidad Nacional del Sur (UNS) and CONICET, Argentina, by the U.S. Department of the Navy, Office of Naval Research, grant N00014-15-1-2742, and by the EU H2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement 690974 for the project "MIREL".

Acknowledgments: We are grateful to CYR3CON (https://cyr3con.ai/) for providing access to the dataset used in our experiments. Images in Figure 1 designed by Freepik from Flaticon (http://www.freepik.com)—used with permission.

Conflicts of Interest: The funding sponsors had no role in the design of the study, in the collection, analyses, or interpretation of data, in the writing of the manuscript, and in the decision to publish the results.

References

1. Elmagarmid, A.K.; Ipeirotis, P.G.; Verykios, V.S. Duplicate Record Detection: A Survey. *Proc. of IEEE TKDE* **2007**, *19*, 1–16.
2. Bleiholder, J.; Naumann, F. Data Fusion. *ACM Comput. Surv.* **2009**, *41*, 1–41.
3. Nunes, E.; Diab, A.; Gunn, A.T.; Marin, E.; Mishra, V.; Paliath, V.; Robertson, J.; Shakarian, J.; Thart, A.; Shakarian, P. Darknet and deepnet mining for proactive cybersecurity threat intelligence. Proc. ISI, 2016, pp. 7–12.
4. NIST. National Vulnerability Database. https://nvd.nist.gov/, 2018.
5. CVE. Common Vulnerabilities and Exposures: The Standard for Information Security Vulnerability Names. http://cve.mitre.org/, 2018.
6. Shakarian, J.; Gunn, A.T.; Shakarian, P. Exploring Malicious Hacker Forums. In *Cyber Deception, Building the Scientific Foundation*; 2016; pp. 261–284.
7. Getoor, L.; Machanavajjhala, A. Entity Resolution: Theory, Practice and Open Challenges. *PVLDB* **2012**, *5*, 2018–2019.
8. Bhattacharya, I.; Getoor, L. Collective Entity Resolution in Relational Data. *ACM Trans. Knowl. Discov. Data* **2007**, *1*.
9. Whang, S.E.; Menestrina, D.; Koutrika, G.; Theobald, M.; Garcia-Molina, H. Entity Resolution with Iterative Blocking. SIGMOD 2009. Stanford, 2009.
10. Bhattacharya, I.; Getoor, L. Query-time entity resolution. *Journal of Artificial Intelligence Research* **2007**, *30*, 621–657.
11. Bahmani, Z.; Bertossi, L.E.; Vasiloglou, N. ERBlox: Combining matching dependencies with machine learning for entity resolution. *Int. J. Approx. Reasoning* **2017**, *83*, 118–141.
12. Fan, W. Dependencies Revisited for Improving Data Quality. Proc. of ACM PODS, 2008, pp. 159–170.
13. Fan, W.; Jia, X.; Li, J.; Ma, S. Reasoning About Record Matching Rules. *Proc. VLDB Endow.* **2009**, *2*, 407–418.
14. Bertossi, L.E.; Kolahi, S.; Lakshmanan, L.V.S. Data Cleaning and Query Answering with Matching Dependencies and Matching Functions. *Theory Comput. Syst.* **2013**, *52*, 441–482.
15. Ho, T.N.; Ng, W.K. Application of Stylometry to DarkWeb Forum User Identification. In *Information and Communications Security*; Springer, 2016; pp. 173–183.
16. Swain, S.; Mishra, G.; Sindhu, C. Recent approaches on authorship attribution techniques: An overview. 2017 International conference of Electronics, Communication and Aerospace Technology (ICECA), 2017, Vol. 1, pp. 557–566.
17. Orebaugh, A.; Allnutt, J. Classification of instant messaging communications for forensics analysis. *The International Journal of Forensic Computer Science* **2009**, *1*, 22–28.
18. Rocha, A.; Scheirer, W.J.; Forstall, C.W.; Cavalcante, T.; Theophilo, A.; Shen, B.; Carvalho, A.R.B.; Stamatatos, E. Authorship Attribution for Social Media Forensics. *IEEE Transactions on Information Forensics and Security* **2017**, *12*, 5–33.
19. Marin, E.; Diab, A.; Shakarian, P. Product offerings in malicious hacker markets. Proc. ISI, 2016, pp. 187–189.
20. Nunes, E.; Shakarian, P.; Simari, G.I. At-risk system identification via analysis of discussions on the darkweb. Proc. eCrime, 2018, pp. 1–12.
21. Tavabi, N.; Goyal, P.; Almukaynizi, M.; Shakarian, P.; Lerman, K. DarkEmbed: Exploit Prediction With Neural Language Models. Proc. AAAI, 2018.