Multi-Domain Alias Matching Using Machine Learning

Michael Ashcroft*, Fredrik Johansson†, Lisa Kaati*† and Amendra Shrestha*
*Dep. of Inf. Technology
Uppsala University, Uppsala, Sweden
Email: firstname.lastname@it.uu.se
†Swedish Defence Research Agency (FOI), Stockholm, Sweden
Email: firstname.lastname@foi.se

***Abstract*—We describe a methodology for linking aliases belonging to the same individual based on a user's writing style (stylometric features extracted from the user generated content) and her time patterns (time-based features extracted from the publishing times of the user generated content). While most previous research on social media identity linkage relies on matching usernames, our methodology can also be used for users who actively try to choose dissimilar usernames when creating their aliases. In our experiments on a discussion forum dataset and a Twitter dataset, we evaluate the performance of three different classifiers. We use the best classifier (AdaBoost) to evaluate how well it works on different datasets using different features. Experiments show that combining stylometric and time-based features yield good results on our synthetic datasets and a small-scale evaluation on real-world blog data confirm these results, yielding a precision over 95%. The use of emotion-related and Twitter-related features yield no significant impact on the results.**

I. Introduction

In today's society, social media services are used more than ever. User generated content available in social media platforms contain massive amounts of information. Aggregation of such information can bring value for various types of actors. Large scale analysis of social media data can e.g. be used for purposes such as sentiment analysis, election forecasting, and selling ads, but also for more security related applications such as discovering potential violent extremists who use social media for spreading propaganda.

A common problem to all these applications is that people may use several social media services, or even several user accounts within the same social media service (e.g., one individual may have several aliases on a single discussion forum or be the author of several blogs). If so, it is in general better to build a user profile based on information from all accounts belonging to the individual rather than to treat them individually. If a person express an interest in sports in one site and clothes in another, more focused ads can be targeted to the individual than if the accounts were assumed to belong to two different individuals. In the same manner, if a person asks for advice on how to create improvised explosive devices using one social media service and express hatred against some specific religion in another, information about these two accounts belonging to the same individual may be very important when security services assess the threat posed by the holder(s) of the accounts. For this kind of reasons, it becomes important to be able to detect and fuse aliases or accounts belonging to the same individual.

Several types of features can be used when linking aliases within a social media service or across services. All services are not alike, which make some features useful for some services and not for others. However, generally speaking, most social media services use the concepts of usernames and postings, where postings in general consist of some user generated content as well as a publishing time. Additionally, many sites use various types of user profiles or user attributes in which the users can fill in information such as e-mail, gender, age, location, interests, etc. Obviously, similar information can also be present in the actual postings, but then in a more unstructured way.

In this paper we focus on using the actual user generated content to detect and link users with multiple aliases. This is accomplished by extracting stylometric-based, emotion-based, and time-based features from the postings of a user. The reason for using these sets of features instead of others such as usernames is that our suggested features can be exploited also in cases where individuals actively try to hide that multiple aliases belong to the same individual (e.g., by choosing very dissimilar usernames). The focus in our conducted experiments is on alias matching within a single social media platform. However, we test the proposed methodology on both a discussion forum, Twitter, and blogs to understand how well the the method works for intra-platform alias matching on several different social media services. We also make some initial testing on inter-platform linkage.

The main contributions of this paper is i) the formulation of the alias matching problem as a binary classfication problem, ii) systematic evaluation of various classification algorithms on well-controlled datasets, and iii) an evaluation of the proposed methodology on blog data, consisting of authors who own multiple blogs.

A. Outline

The rest of this paper is structured as follows. In Section II we present previous work that is related to alias matching. Section III describes some motivation to why alias matching is an important problem and provides a definition of the

problem. The experimental setup, including what features we have used and what datasets we have conducted our experiments on, is described in Section IV. We have done three different experiments on large-scale synthetic datasets: the first evaluates the performance of three different classifiers (AdaBoost, SVM, and Naive Bayes), the second evaluates how the best classifier performs on the different datasets and with different feature sets, while we in the third experiment evaluate the performance of cross-classification (training on one dataset, testing on another) and combined classification (training and testing on a mixture of two data sources). The results of our main experiments are presented in Section V. In Section VI we present an evaluation on blog data from bloggers authoring several blogs, giving an indication of how well the method will work in real applications. In Section VII we further reason about how well our trained models can classify users with multiple aliases in real applications. Finally, Section VIII contains a discussion on the obtained results, some conclusions, and directions for future work.

II. Related work

In [1], aliases are linked to each other by analyzing the content of their social media postings, or more specifically, the vocabulary used by the various aliases. Their initial experiments suggest that the use of general words as features is more useful for alias matching than other features such as function words and punctuation (stylometric features related to what we propose in this work). However, although the suggested method seems to work well for the used synthetic datasets where an individual sticks to a single topic, it performs much worse when being applied to more heterogenous writings. This shows that context-dependent words are not suitable for being used as features in real-world scenarios where individuals can write about different topics on different occasions or using different social media accounts. Hence, this also highlights that alias matching algorithms which seem to work well in synthetically generated test environments are not necessarily performing well "in the wild". This makes it extra important to check the assumptions used when creating the training and test sets, as well as striving to also test proposed methods under as realistic circumstances as possible.

In [2], it is argued that accounts across different communities can be linked using usernames as features. It is also mentioned that the uniqueness of e-mail addresses can serve as a universal identifier across communities, but since they are less "available" than usernames, the latter is focused upon. The authors propose a method to identify corresponding usernames across communities by extracting a base username from a base community, generating candidate usernames by adding and removing prefixes and suffixes from the base username, and to search for those candidate usernames on the target community using the Google search engine. The proposed method is evaluated, resulting in an average accuracy around 66%.

Along the same lines as in [2], the feasibility of using usernames to link multiple profiles belonging to the same individual is investigated in [3]. While the approach suggested in [2] assumes that identical usernames found in different online services belong to the same individual, Perito et al. [3] develop a model for estimating that two usernames (identical or not) from two separate social media services belong to the same individual. A central component in their approach is a Markov chain model for estimating the probability of encountering a certain username. The Markov chain model is trained on approximately 10 million usernames gathered from Google and eBay. The estimated probability is then used to estimate how unique the common part of two usernames we test for linkage is. If the common part turns out to be relatively common, the two usernames are quite likely to belong to various individuals even though they are identical or very similar, while it is more likely that they belong to the same individual if the shared part of the usernames is very uncommon.

In [4], the supervised method MOBIUS for finding mappings among identities of individuals across social media sites is suggested. It builds upon a large set of features extracted from the usernames (including the ones suggested in [3], [2]). Many classification techniques are evaluated in their experiments, but a logistic regression classifier is shown to perform best (with an accuracy of 93.8%). This classifier performs much better than the methods suggested in [3], [2] as well as a number of baselines (such as exact username matching and substring matching).

In [5], the framework HYDRA is proposed for enabling large-scale social identity linkage across social media platforms. HYDRA consists of two main components: one for measuring the heterogeneous behavior similarity between users and one for leveraging users' core social network structures. These components are combined using multiobjective optimization. In the behavior similarity calculation, many features are taken into account including a comparison of user profile images, various textual attributes from the users' profiles, and a rudimentary modeling of writing style. The suggested method is evaluated on impressively large datasets (several million Chinese users with accounts on several social networks obtained from a third-party data provider). The results show that HYDRA outperforms the other approaches, which is unsurprising since the other approaches take fewer sets of features into account.

Various unsupervised distance-based alias matching algorithms have previosly been proposed, e.g. in [6]. Moreover, many studies on supervised learning algorithms for the problem of authorship attribution exist, see e.g., [7], [9]. The main differences among previous approaches and the approach we propose in this paper are the formulation of the alias matching problem as a binary classification problem, the kind of features used for the linkage of profiles, and that we focus on intra-platform linkage (i.e., alias matching within a single social media platform) while the methods described above all are developed for inter-platform linkage. We are not aware of any previous studies which have combined stylometric features with time-based and emotion-based features as in this work.

III. Alias matching in social media

There are many reasons for why someone would like to use multiple aliases. A long list of reasons can be thought of, such as users having forgotten their username or password, or being banned by a moderator [6]. However, privacy aspects and reasons such as avoiding law enforcement agencies to link illegal activities to a physical identity are also common. For users who simply have created an extra alias due to a forgotten password or who use different usernames on different social media services, it is seldom of interest to actively try to "hide" that multiple aliases belong to the same individual. Hence, in these cases, alias matching methods based on finding similarities in usernames often work very well. On the other hand, if a user has created an extra alias for privacy reasons or for performing internet hate crimes, it is less likely that an approach based on matching of usernames will work. The features we are using in our proposed methodology are therefore not based on usernames, but are rather based on stylometry, emotions, and time which are not as easily altered. Note, though, that nothing prohibits combining the features suggested with others such as usernames in cases where this is appropriate.

A. The alias matching problem

The problem of alias detection can be defined as given two feature vectors $\mathbf{a_1}$ and $\mathbf{a_2}$ which are extracted from two different aliases, we would like to determine whether they belong to the same individual $\mathcal{I}$, i.e., we would like to learn an identification function $f(\mathbf{a_1}, \mathbf{a_2})$ where:

$$f(\mathbf{a_1}, \mathbf{a_2}) = \begin{cases} 1 \text{ if } a_1 \text{ and } a_2 \text{ belong to the same } \mathcal{I}, \\ 0 \text{ otherwise.} \end{cases} \quad (1)$$

A feature vector is an n-dimensional vector of numerical features that represents an object (in this case an alias). For the alias matching problem, we can distinguish between two main cases: 1) The aliases a_1 and a_2 are used on the same social media site (intra-platform alias matching), or 2) the aliases stem from different social media sites (cross-platform/inter-platform user linkage).

B. Proposed methodology

In order to learn the identification function defined above using our proposed methodology, we first extract all postings for a large number of users. Next, we randomly distribute a user's posts among a small set of sub-users. For each sub-user, we create a feature vector (which features that are used are explained in more detail in Section IV-A) and a user ID, where the user IDs are unique so that all sub-users created from a user share the same user ID, but is different among all sub-users who were not created from the same user.

Next step in the suggested methodology is to calculate the pairwise absolute difference of all the resulting feature vectors. After taking the the pairwise difference, the user ID is now specifying whether sub-users were taken from the same user (0, altered to 1), or not ($\neq$ 0, altered to 0 in order to create a binary classification problem). By using this transformation, we alter the problem from an unsupervised learning task of finding similar sub-user vectors (previously used in [6]), to a classification problem of classifying matching and unmatching pairwise-difference vectors. There are many alternative classifiers which can be learned from such training data, but in Section V we describe how we have trained three different classifiers using the suggested approach.

For cases where we are interested in detecting multiple aliases belonging to the same individual within a set of aliases of interest, we can simply calculate the pairwise differences of each pair of feature vectors and use the resulting vector as input to the classifier. For pairs where the classifier give a 1 as output, this is an indication of that both user accounts belong to the same author. If we instead have a problem where we would like to find out if a certain alias belong to a certain individual (i.e., a more traditional authorship attribution problem), we can construct a feature vector from user generated content known to belong to the individual of interest and calculate the pairwise difference to features extracted from the postings from the anonymous user account and see whether they match or not.

IV. Experimental setup

We have conducted a series of experiments in order to find out how well our proposed methodology works when it comes to discovering the use of multiple aliases.

A. Features

In the experiments we have evaluated the impact of using various feature sets. The first feature set which has been tested consists of the stylometric features described in [6] and is in the following referred to as feature set S. This feature set has previously been shown to yield good results for distance-based alias matching. In addition to these "standard" features, we have in this paper experimented with two additional feature sets. The first of these is the time-based features introduced in [7] (in the following referred to as feature set T). We have also experimented with the new feature set described in Table I. As can be seen, these features are intended to capture various kinds of emotions and Twitter-specific content. The reason for incorporating these features was that we expected traditional stylometric features to yield worse performance for tweets and that we wanted to add features which better could capture such content. In the following, we use the notation E when referring to this feature set. In the experiments, we are sometimes combining several sets of features into a single combined feature vector. In such circumstances we use notation such as S+T+E, which should be interpreted as the feature vector obtained when combining stylometric features with time-based features and the new emotion-related feature set.

B. Datasets

Two data sources have been used to generate our datasets. Firstly, we have extracted posts from the ICWSM forum dataset, containing data from the Irish forum site

TABLE I
THE EXTENDED FEATURE SET FOR EMOTIONS AND TWEET-SPECIFIC CONTENT

Category	Description	Count
Emotion words	Relative frequency of various sentiments words	108
Smilies	Relative frequency of various smilies :') :-) ;-) :P :D :X ¡3 :) ;) :@ :* :— :$ %)	14
Hashtag	Relative frequency of hashtags	1
User Mention	Relative frequency of user mentions	1
URLs	Relative frequency of URLs	1

https://www.boards.ie/. From this data we have selected the top-1000 posters from year 2007 and extracted all their posts. The reasons for selecting these users is that we wanted to have as large data material as possible. In next step, we have splitted each user u_i into five[1] "sub-users" $u_{i1}, u_{i2}, \ldots, u_{i5}$, where each of the user's posts have been randomly assigned one of the sub-users. From this set of 5000 sub-users, we have (for each combination of feature sets) extracted the corresponding feature vector and their user ID. We have then for each pair of sub-users taken the absolute difference among their feature vectors, resulting in 12,497,500 cases with the user id feature now specifying whether sub-users were from the same user (10,000 cases) or not (12,487,500 cases). Out of these cases, all matching instances have been used and divided among the training and test sets in a 9:1 training to test ratio. Similarly, we have randomly selected equally many non-matching cases and divided them among the training and test sets using the same ratio, ending up with a training set consisting of 18,000 cases (half unmatched and half matched) and a test set consisting of 2,000 cases (half unmatched and half matched). The dataset obtained after this procedure is in the following referred to as DB-All. Additionally, we have created a dataset of equal size in the same manner but where we have limited the amount of posts to 60 randomly selected posts for each user, used for testing the impact of the amount of data on the alias matching performance. This dataset is in the following referred to as DB-60.

We have also constructed two datasets from user accounts on Twitter in the same way as the discussion forum datasets have been created. The tweets used for creating those datasets were collected from Twitter during 2012. Since the crawled tweets contained multiple languages, we used the Google language-detection API [10] to detect the tweets' language and to select only English tweets. From this dataset we have randomly selected 1000 users who have posted at least 60 tweets, resulting in a set of 1000 users where the tweet count range from 140 (minimum) to 564 (maximum), with a majority of users posting in between 140 and 200 tweets. For each tweet written by a user we check whether it is a tweet or a retweet. If it is a retweet we extract the publishing time for using it when building our feature vector, but discard the text content in order to avoid building a feature vector from text which has been written by someone else in the first place. If it is a normal tweet, we count the number of URLs, mentions, hashtags, and smileys as described for the extended feature set, whereupon those are removed from the original tweet before extracting the stylometric features. Based on this data, we have created two datasets: TW-All and TW-60.

[1]The number of "sub-users" has been arbitrarily selected, following the approach in [7].

C. Classifiers

We examined the performance of AdaBoost, SVM, and naive Bayes (NB) classifiers. Good overviews of all three techniques are available in [11]. AdaBoost classifiers were created using the ada R package [12], and for the SVM and NB classifiers we used the e1071 R package [13].[2] For the SVM, a radial kernel was used. All other parameters were set to default values for the SVM and NB classifiers.

For AdaBoost, we used classification trees as base classifiers, wherein the feature space was divided in regions by recursive partitioning. We set the maximum number of iterations to 400 in the first experiment and 200 in the other experiments (due to the amount of time needed for running the experiments). The default parameters were used for the rest of the AdaBoost parameter settings.

D. Evaluation

The results from the comparisons among classifiers and various feature sets on the datasets of interest have been reported using confusion matrices in which we present the number of true positives, false negatives, true negatives, and false positives as illustrated in Table II. To compare the obtained results, we have used a standard statistical significance test (the proportional test as described in [14]) in which the null hypothesis has been that the two classifiers or feature vectors that are compared have performed equally good. In order to determine if the obtained differences are statistically significant, we have calculated p-values (e.g., the probability of observing an effect given that the null hypothesis is true) and evaluated whether the null hypothesis should be rejected for various values of the significance level α.

TABLE II
CONFUSION MATRIX

	Predicted class	
Actual class	**True Neg. (TN)**	**False Pos. (FP)**
	False Neg. (FN)	**True Pos. (TP)**

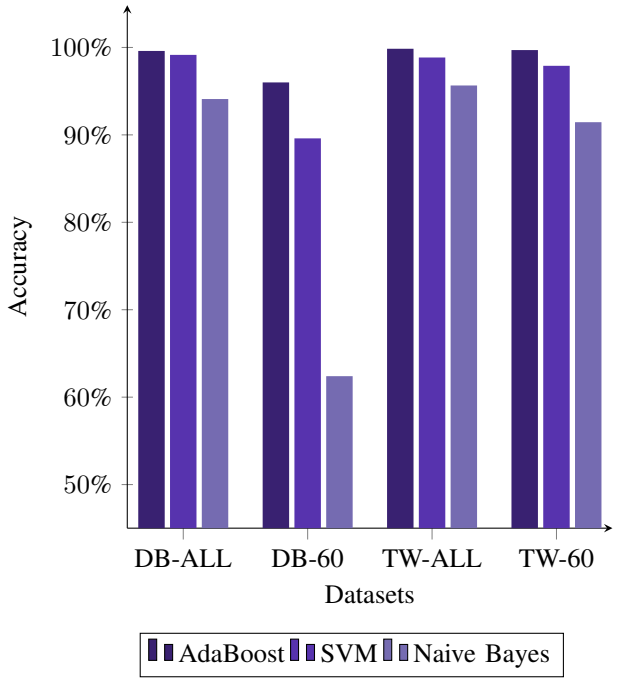

Fig. 1. Accuracy of the performance of the different classifiers on the different datasets using features S+T+E.

V. Experimental results

A. Experiment 1: Performance of Classifiers

In our first experiment, we have made use of all the suggested features (i.e.,S+T+E) and tested how well the three classifiers work on the four different datasets (DB-All, DB-60, TW-All, and TW-60). When running this experiment we ended up with the results reported in Table III. Figure 1 shows the corresponding accuracy results.

TABLE III
A comparison of the performance of the different classifiers on the different datasets when using the full set of features.

	DB-All		DB-60		TW-All		TW-60	
AB	994	6	899	101	998	2	995	5
	2	998	87	913	1	999	1	999
SVM	985	15	878	122	979	21	965	35
	2	998	86	914	2	998	7	993
NB	968	32	890	110	969	31	964	36
	86	914	642	358	56	944	135	865

Analyzing the obtained statistical significance test results, we find that AdaBoost and SVM perform significantly better than the NB classifier on the discussion board datasets. On DB-60l, the AdaBoost is also significantly better than the SVM classifier, while there is no significant difference between the two on DB-All. For the TW-All data, there is no statistically significant difference between AdaBoost and the SVM classifier either. For the TW-60 data, there is on the other hand a significant difference between AdaBoost and SVM also on the lowest significance level of $\alpha = 0.005$. These results suggest that AdaBoost and SVM both perform significantly better than the NB classifier and that the overall classification error of the AdaBoost classifier is lower than for the SVM. For these reasons, we are in the following experiments selecting to use the AdaBoost classifier.

[2]Both packages are freely available from http://CRAN.R-project.org/

B. Experiment 2: Performance on Different Datasets and Features

In our second experiment, we have compared the performance of the best classifier (i.e., AdaBoost) over all datasets using different feature vectors. The results of this experiment are summarized in the confusion matrices in Table IV and, for a subset of the feature combinations, in Figure 2.

When studying the confusion matrices for the AdaBoost classifier, we can see that the rates of false positives and false negatives overall are low enough to suggest that this kind of models performs well enough to be used in real-life applications (except for the DB-60 dataset). With this said, the expected performance of the model on data with other matched-unmatched ratios is discussed further
in Section VII.

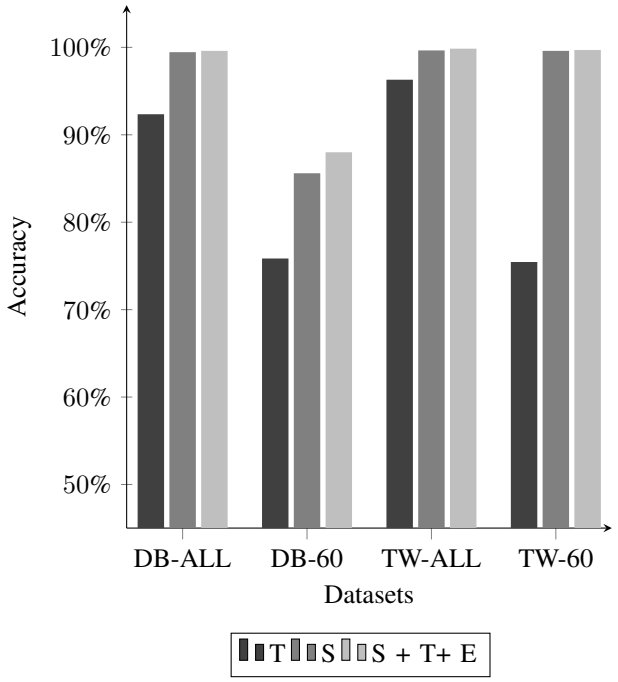

Fig. 2. Performance of AdaBoost on different datasets and features.

The findings from the experiment are not completely consistent among the various datasets. However, we can clearly see that time-based features alone are outperformed by the alternative feature vectors. For the DB-All and DB-60 data, we can see that it is beneficial to combine stylometric and time-based features. What is more unclear is whether the feature

TABLE IV
A COMPARISON OF THE PERFORMANCE OF ADABOOST ON A NUMBER OF DIFFERENT FEATURE VECTORS AND DATASETS.

	DB-All	**DB-60**	**TW-All**	**TW-60**
T	927 73 80 920	705 295 188 812	960 40 34 966	741 259 232 768
S	993 7 4 996	872 128 160 840	995 5 2 998	994 6 2 998
S + T	998 2 1 999	879 121 150 850	995 5 1 999	994 6 0 1000
S + E	992 8 3 997	868 132 159 841	995 5 5 995	994 6 3 997
S + E +T	994 6 2 998	883 117 123 877	998 2 1 999	995 5 1 999

set containing emotions should be used in combination with those, or if it is better to avoid adding it to the feature vector. On a significance level of 0.1 there is some reason to exclude it for DB-All, while there on the same significance level is some reason to incorporate the emotion features for DB-60. For the Twitter data we can see no reason to exclude time or the new emotion features, but there are on the other hand no significant difference between using any other combination of feature vectors (except for using the time-based features on their own which is clearly not beneficial).

Based on these results we argue that stylometric and time-based features give good results overall and should be used in combination. The value of also adding the emotion-based features is not clear, but since it adds very little extra computational effort to include these features, we have decided to use the S+E+T feature vector in the remaining experiment. Looking at the confusion matrices for this combination of features, we can see that the number of false positives and false negatives are low, except for the DB-60. Interestingly, the difference in performance between TW-All and TW-60 becomes quite small for all features (except for when using only time).

C. Experiment 3: Cross-classification

In the next experiment we have examined how well a classifier trained on data from one social media platform performs when evaluated on another (i.e., cross classification/inter-platform linkage). This is of interest for cases such as where we would like to determine the (unknown) author of a number of tweets given the forum posts written by a set of known set of candidate authors. In order to check this we created data where we trained on discussion forum data / Twitter, and tested on the other. As in the other experiments we used 18,000 training and 2,000 test cases, equally split between classes. We also investigated whether classifiers trained on a mixture of data from both the DB-All and TW-All datasets could classify such a mixture. The results are presented in Table V. As can be seen, cross-classification performed very poorly while the combined classification performed excellently. In fact, the combined classification performance was not significantly worse than either of the best models trained and tested on a single dataset. AdaBoost's use of boosting is likely involved in this, as it permits different base classifiers in the ensemble to concentrate on learning to classify different subsets of the training data. The good results of the combined classification indicate that a direction for future work is to examine whether training classifiers on data drawn from large numbers of varied data sources permits generalization to new data sources.

TABLE V
THE RESULTS OF THE CROSS-CLASSIFICATION EXPERIMENT.

Training data	**Test data**	**Results**
DB-All	TW-All	46 951 2 998
TW-All	DB-All	96 904 0 1000
Combined	Combined	997 3 1 999

VI. EVALUATION ON BLOG DATA FROM AUTHORS WITH MULTIPLE BLOGS

Although we have been able to reach very promising results in our experiments, it has to be highlighted that results obtained on synthetically generated data are not necessarily transferable as is to more realistic circumstances.

As a first attempt to evaluate the method on non-synthetic data we have downloaded publicly available Google blogs[3] for 4500 bloggers, out of which some authors have authored several blogs. Most of the blogs were written about personal interests, news, fashion and photography. On these blogs, we used a combination of the Google language-detection API [10] and manual verification for language detection of the blogs in order to only include blogs written in English (or in some cases, English mixed with other languages). Hence, completely non-English blogs were removed. Moreover, the blogs were filtered so that only those containing 60 blog posts or more were included in our experiment. For blog posts in English we extracted stylometric-based and time-based features, while for mixed-language blog posts we extracted only time-based features for the non-English posts. After the blog filtering process, feature vectors were created for the remaining 1800 blogs. In total there were 1414 distinct bloggers out of which 260 had written at least two separate blogs.

In the next step, we labeled the data using the assumptions that blogs from the same Google account were written by the same author, and that blogs from different Google accounts were written by different authors. Based on the results obtained in the first set of experiments we made use of an AdaBoost classifier with stylometric, time-based, and emotion-based features. Initial models were trained on 4000 examples, with 663 matched pairs of blogs. The rest of the training data examples were randomly selected unmatched pairs of blogs. The trained classifier was tested on 1000 examples, with 150 matched pairs of blogs and the others randomly selected unmatched pairs of

[3]https://www.blogger.com/

blogs. All matched pairs of blogs were either in the training or test data. The obtained results on the test set are summarized in Table VI. This corresponds to a precision of 0.966, a recall of 0.567, and an accuracy of 0.929. As argued further in Section VII, the precision will in a real world application be the most important of these metrics.

TABLE VI
CONFUSION MATRIX FOR THE ADABOOST CLASSIFIER ON THE BLOG TEST DATA

	Predicted class	
Actual class	**847 (TN)**	**3 (FP)**
	65 (FN)	**85 (TP)**

VII. DISCUSSION

The most important difference between alias matching in our controlled experiments using synthetic data and alias matching in the wild is that while we in our experiments have used well-balanced datasets for training and testing, this ratio will in some real-world applications be highly unbalanced. For example, if we on a real-world discussion forum would like to find out whether there are any users that have made use of multiple aliases we would have to take the absolute difference of feature vectors among all pairs of user accounts present on the forum. Among these pairs, it would most probably be a much higher proportion of unmatching pairs compared to matching pairs, since most people will use just a single username/account within a given forum. On the other hand, if our algorithm would be used in a cyber crime investigation and be presented with a series of threatening messages originating from a suspect's IP address, the prior probability of a match can be expected to be close to the matching/unmatching-ratio in our presented experiments. So what effect can this rate be expected to have on the classification performance?

The discussion board dataset (DB-All) has a 1:1 ratio of classes, meaning an equal number of matched and unmatched pairs. This matched the ratio found in the data we used to train the model. However, the larger dataset that was used to obtain the dataset for the experiments was composed of almost 13 million cases, of which only 10,000 were matched cases. The confusion matrix we would expect when applying the DB-All/S+T AdaBoost model to this complete data is a multiple of (figures chosen to make the smallest field 1):

828002	1666
1	998

This would yield the following statistics:

Accuracy:	.998
Sensitivity:	.999
Specificity:	.998
Precision:	.375

These numbers illustrate that an increase in the ratio between unmatched and matched pairs leads to a significant drop in precision. The precision that can be expected when alias matching is done on actual social media platforms will be lower or higher depending on the ratio of matching to non-matching accounts present. If we can expect only one matching account per n, the ratio of unmatched to matched accounts, r, will be $\binom{n}{2}-1 : 1$. Table VII shows expected precision rates for how AdaBoost models built from the different datasets would perform for different values of n, while Table VIII shows the expected performance for different values of r.

The results are calculated based on the assumption that the sensitivity and specificity performance of the model remains constant while the matched:unmatched ratio present in the data is varied. When calculating the confusion matrix for the complete dataset, this is reasonable, since the matched and unmatched pairs in the test data were selected randomly from matched and unmatched pairs of the complete dataset.

To understand the importance of how these results would impact a real application we need to examine some potential situations where an alias matching model could be used. We have identified three main types of cases where alias matching can be used:

1) A human expert brings in cases that she suspects are users with multiple aliases. The model is used to confirm or reject the human expert's suspicions.
2) The model is used to classify accounts that it suspects are multiple alias accounts, and the human expert is used for confirmation.
3) The model classifies accounts that it suspects are alias accounts with such accuracy that we do not require human confirmation.

In case (1) the human expert alters the distribution of cases brought to the model. If her judgment is such that she is correct 1 in 20 times, and in so doing she does not (significantly) affect the sensitivity and specificity performance of the model, we would expect the precision of the DB-All (S+T) and TW-All (S+T+E) models on this presented data to be that of the row associated with $r = .95$ which is .963. Note that this requirement that the sensitivity and specificity performance of the model not be affected is important, and ideally applications in such a scenario should attempt to provide evidence that this assumption is met.

In case (2), the viability of such a procedure relies on providing a manageable number of potential true positives for the human expert to analyze. So if the natural ratio of unmatched to matched accounts is $100 : 1$, for the DB-All (S+T) and TW-All (S+T+E) models we would expect the human expert to be faced approximately 10 false positives for every true positive (row corresponding to $r = .99$) in the cases presented to her for analysis.

Case (3) requires either orders of magnitude improvement in (already impressive) specificity or a low ratio of unmatched to matched cases. We see, for instance, that the DB-All (S+T) and TW-All (S+T+E) models falls below the common confidence threshold of .95 somewhere between $n = 5$ and $n = 10$.

TABLE VII
EXPECTED PRECISION RATES FOR DIFFERENT MODELS FOR DIFFERENT VALUES OF n.

n	r	DB-All (S + T)	DB-60 (S+T+E)	TW-All (S+T+E)	TW-60 (S+T+E)
5	.909	.980	.474	.980	.952
10	.978	.917	.167	.917	.816
20	.995	.724	.045	.724	.513
50	.9992	.290	.007	.290	.140
100	.9998	.092	.002	.092	.038
1000	.999998	.001	.00002	.001	.0004
10000	.99999998	.00001	.0000002	.00001	.000004

TABLE VIII
EXPECTED PRECISION RATES FOR DIFFERENT MODELS FOR DIFFERENT VALUES OF r.

r	DB-All (S + T)	DB-60 (S+T+E)	TW-All (S+T+E)	TW-60 (S+T+E)
.9	.983	.501	.982	.957
.95	.963	.322	.963	.913
.98	.911	.156	.911	.803
.99	.834	.084	.834	.669
.999	.333	.009	.333	.167

VIII. CONCLUSIONS AND FUTURE WORK

We have presented the problem of linking aliases belonging to the same physical individual. Previous approaches to the problem have mainly focused on using usernames for linking aliases, but we have instead focused on stylometric, time-based, and emotion-based features since such features are more likely to be useful also for situations where people more actively would like to hide that they make use of several profiles. We present a transformation of the multi-class author attribution problem which allows for a machine learning approach to learn the difference among matching and unmatching feature vectors.

We have in our alias matching experiments shown that AdaBoost classifiers perform significantly better than Naive Bayes classifiers, and in most cases also significantly better than Support Vector Machines. Stylometric and time-based features work very well when detecting multiple aliases, and the combination of stylometric and time-based features is even more powerful. We have not found any significant reasons for excluding the suggested emotion-based features, but for most tested datasets there is not a significant advantage in including it either.

Our experiments on a small set of Google blogs containing accounts authoring both single and multiple blogs confirm that the proposed method can be used also for real-world linkage of user accounts, given that precision rather than recall is of top priority.

ACKNOWLEDGMENTS

This research has been financially supported by Security Link and the Swedish Armed Forces Research and Development Programme.

REFERENCES

[1] J. Novak, P. Raghavan, and A. Tomkins, "Anti-aliasing on the web," in *Proceedings of the 13th international conference on World Wide Web*. New York, NY, USA: ACM, 2004, pp. 30–39.

[2] R. Zafarani and H. Liu, "Connecting corresponding identities across communities." in *ICWSM*, E. Adar, M. Hurst, T. Finin, N. S. Glance, N. Nicolov, and B. L. Tseng, Eds. The AAAI Press, 2009.

[3] D. Perito, C. Castelluccia, M. Kaafar, and P. Manils, "How unique and traceable are usernames?" *Privacy Enhancing Technologies*, pp. 1–17, 2011.

[4] R. Zafarani and H. Liu, "Connecting users across social media sites: A behavioral-modeling approach," in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '13. ACM, 2013, pp. 41–49.

[5] S. Liu, S. Wang, F. Zhu, J. Zhang, and R. Krishnan, "Hydra: Large-scale social identity linkage via heterogeneous behavior modeling," in *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '14, 2014, pp. 51–62.

[6] F. Johansson, L. Kaati, and A. Shrestha, "Detecting multiple aliases in social media," in *Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2013)*. ACM, 2013, pp. 1004–1011.

[7] ——, "Time profiles for identifying users in online environments," in *Proc. 1st Joint Intelligence and Security Informatics Conference, IEEE Computer Society*, 2014, pp. 83–90.

[8] A. Abbasi and H. Chen, "Writeprints: A stylometric approach to identity-level identification and similarity detection in cyberspace," *ACM Trans. Inf. Syst.*, vol. 26, no. 2, pp. 7:1–7:29, Apr. 2008.

[9] P. Juola, "Authorship attribution," *Found. Trends Inf. Retr.*, vol. 1, no. 3, pp. 233–334, 2006.

[10] N. Shuyo, "Language detection library for java," 2014. [Online]. Available: http://code.google.com/p/language-detection/

[11] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, ser. Springer Series in Statistics. New York, NY, USA: Springer New York Inc., 2001.

[12] M. Culp, K. Johnson, and G. Michailides, "ada: An r package for stochastic boosting," *Journal of Statistical Software*, vol. 17, no. 2, pp. 1–27, 2007.

[13] D. Meyer and T. U. Wien, "Support vector machines. the interface to libsvm in package e1071. online-documentation of the package e1071 for r," 2001.

[14] T. G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," *Neural Comput.*, vol. 10, no. 7, pp. 1895–1923, Oct. 1998.