Alias Detection in Link Data Sets

Paul Hsiung[1] *and Andrew Moore and Daniel Neill and Jeff Schneider*
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
{hsiung+, awm, neill, schneide}@cs.cmu.edu

Keywords: Link Analysis, Information Extraction, Information Sharing and Collaboration, Fusion

Abstract

The problem of detecting aliases - multiple text string identifiers corresponding to the same entity - is increasingly important in the domains of biology, intelligence, marketing, and geoinformatics. Aliases arise from entities who are trying to hide their identities, from a person with multiple names, or from words which are unintentionally or even intentionally misspelled. While purely orthographic methods (e.g. string similarity) can help solve unintentional spelling cases, many types of alias (including those adopted with malicious intent) can fool these methods.

However, if an entity has a changed name in some context, several or all of the set of other entities with which it has relationships can remain stable. Thus, the local social network can be exploited by using the relationships as semantic information.

The proposed combined algorithm takes advantage of both orthographic and semantic information to detect aliases. By applying the best combination of both types of information, the combined algorithm outperforms the ones built solely on one type of information or the other. Empirical results on three real world data sets support this claim.

1. Introduction

The premise of a link data set is that one **entity** represents a unique individual whether it is an actual person, word, or even research paper. However, each entity can have many **names**. If two or more names map to an entity, they are called **aliases**. A link data set consists of a set of names and a set of **links**. Each link contains two or more names and represents an observed relation between the names. The definition of relation is general but can be specified on a particular data set. For example, a relationship can be any connection ranging from membership in the same terrorist cell to collaboration on a research paper. But for this paper, types of relationship between names in the links are not taken into account. The only property used is the statistical information that names appear together.

The way that the intelligence community gathers data is compatible with the way in which link data sets are constructed. Intelligence analysts collect articles (often written in foreign languages) and write down their subjective observations concerning the relations between names inside the articles For example, consider the following web-article [2]:

```
Wanted al-Qaeda chief Osama bin Laden and his
top aide, Ayman al-Zawahri, have moved out of
Pakistan and are believed to have crossed the
border back into Afghanistan.
```

A small number of links will summarize the information. For instance, the following link can describe a terrorist cell:

(*Osama bin Laden*, *Ayman al Zawahri*, *al Qaeda*)

Likewise, these links can describe location information:

(*Pakistan, Osama bin Laden*)
(*Afghanistan, Osama bin Laden*)

Link data sets create a way for algorithms to understand human-readable information, but they are susceptible to noise and often requires human interventions and subjective judgments. The use of link data was previously discussed in these papers: (Goldenberg *et al*. 2003; Kubica *et al*. 2003).

The problem of alias detection is very broad. In another variant of this problem, one name corresponds to many entities. For example the name Michael Jordan represents a statistician and a sports figure as well as many others who share that name. Various methods that address this problem are discussed in Neill (2002) and Jurafsky and Martin (2000).

This paper focuses on many names corresponding to a single entity. For example, *Osamabin Laden* is also known

[1] The first author is currently attending University of Southern California. This work was done while he was at CMU.

[2] http://uk.news.yahoo.com/040225/323/emvp1.html

as *Usama bin Laden*. These two strings are orthographically similar and, hence, are easy-to-spot aliases. But there are other more difficult aliases such as *The Prince* or *The Emir*. To detect these aliases, the local social network structure of these names must be exploited. The **friends** of *Osama* bin Laden are defined as all the names that have some relationship with *Osama bin Laden* (i.e. occur in the same link). To exploit the social network, friends of *Osama* bin Laden are compared with friends of *The Prince*, and some type of correspondence is computed between these two sets of friends.

This paper extends the very appealing idea introduced in Sarawagi and Bhamidipaty (2002) of using active learning to automatically tune up an alias detector. We consider orthographic information of the same type as those used by Sarawagi and Bhamidipaty (2002) but we also presents experiments with adding semantic information based on link data sets. We also investigate multiple classifiers and conclude that logistic regression, which actively attempts to discriminate between classes, is most effective.

2. Probabilistic Orthographic Model

The most natural measure of alias likelihood between two names is orthographic similarity. If two strings are very similar, they are likely to be versions of the same name. One of the most common measures is string edit distance: the minimum number of insertions, deletions, and substitutions required to transform one string into the other (see section 5.6 of Jurafsky and Martin (2000)).

There are many possible string edit distance functions that measure similarity. How do we select, or combine multiple orthographic measures? This will be discussed in Section 4. Four orthographic measures are described below, with other possibilities mentioned in Zobel and Dart (1995; 1996).

String Edit Distance (SED) The minimum number of insertions, deletions, and substitutions required to transform one string into the other.

Normalized String Edit Distance (NSED) This is computed by dividing *SED* with the max length of the two strings we are comparing.

$$NSED(s_1, s_2) = \frac{SED(s_1, s_2)}{max(length(s_1), length(s_2))}$$

Discretized String Edit Distance (DSED) DSED is NSED binarized by a threshold of 0.7. This threshold was selected by empirical observation.

Exponential String Edit Distance (ESED)
$ESED(s_1; s_2) = exp(SED(s_1; s_2))$

3. Probabilistic Semantic Model

The set of links in which names have appeared offers additional secondary evidence for this problem. For example, if two people have almost exactly the same set of friends (i.e. people with whom they have historically occurred) in the same historical proportions, then it is more likely that the names are aliases for one person. Such measures include:

Dot Product (DP) A name's friends list is represented as a vector of occurrences with other names it appears with. Dot Product is just the dot product of two vectors from two names. For example, in Table 1,

$$DP(Osama, ThePrince) = 10 * 2 + 2 * 8 = 36$$

Table 1: Hypothetical Example of Friends List

	Osama	The Prince
# Occurrences with AlQaeda	10	2
# Occurrences with CNN	2	8
# Occurrences with Music	0	50
# Occurrences with Islam	5	0

Normalized Dot Product (NDP) This is almost the same except each vector is normalized by dividing by its magnitude before taking the dot product. In Table 2,

$$DP(Osama, ThePrince) = \\ 0.588 * 0.033 + 0.118 * 0.133 = 0.0351$$

Table 2: Normalized Friends List

	Osama	The Prince
Normalized AlQaeda	0.588	0.033
Normalized CNN	0.118	0.133
Normalized Music	0.000	0.833
Normalized Islam	0.294	0.000

Common Friends (CF) is defined as number of friends that co-occur with both names. In above example, *AlQaeda* and *CNN* co-occur with *Osama* and *The Prince*, so the measure score is two.

KL Distance (KL) Normalized friends lists can be treated as probability vectors. *KL* can then be used to measure the similarity between these two vectors. Add-one smoothing (i.e. add one to each value of the vector before normalization) is applied to deal with cases where two entities do not co-occur in a link. The KL distance is given by:

$$\sum_i O_i log\left(\frac{O_i}{P_i}\right) + P_i log\left(\frac{P_i}{O_i}\right)$$

where P_i is the *i*th entry in probability vector of *The Prince* and O_i is the *i*th entry of *Osama*.

4. Combined Model

How should these measures be combined? Four approaches are considered:

1. **Manually pick the best single measure.** This "feature selection" approach does not work well because orthographic and semantic measures convey different information. This is confirmed in the results section, where the combined model outperforms those using only orthographic or only semantic measures.
2. **Hand designed formulas** This method is based on subjective judgments made by human beings. This might not produce the optimal combination. Worse still, the best combination is likely to change from domain to domain. Examples of this approach are Baroni *et al.* (2002), who used weighted sum, and Hernandez and Stolfo (1997), who manually created rules.
3. **Probabilistic model** Another approach is a full probabilistic generative model of links, names, and the string corruption process. Under the assumption that it is possible to model such a complex set of link phenomena and that it is computationally tractable to solve such models for more than a few hundred names, this would be a very promising approach. Marthi *et al.* (2003) gives a very detailed description of what such a generative model looks like in the related area of bibliometrics analysis. This area is interesting although the representational and computational challenges are severe.
4. **Supervised learning** This approach requires a small hand-selected set of positive and negative examples of whether a pair of names are aliases. These examples are used to build a classifier that tests if two names are aliases. This is the approach taken in this paper.

5. Alias Classifier

5.1 Training

In this combined model, the goal is to train a classifier that tests whether two names in a link data set are aliases. For the purpose of training the classifier, positive examples come from hand selected aliases. Because the pool of names is so large and the number of true aliases is small, with a high degree of safety, negative examples are picked by randomly selecting pairs of names among the link data set. For each pair of names, all measures in both the semantic and orthographic models are calculated and incorporated as **attributes** into the training set. An output attribute called *Alias?* labels each example as positive or negative. Each pair of names represents a row in the training set. For example, see Table 3.

Table 3: Hypothetical training set

		Orthographic Measures		Semantic Measures		
name1	name2	SED	...	DP	...	Alias?
Osama	The Prince	15	...	36		Yes
Usama	Usama	2	...	24		Yes
...	...	...	...	...	...	...
Bob	Sid	6	...	2		No
John	The Prince	10	...	5		No

Now the problem of alias detection is transferred to the more familiar world of straightforward classification. A series of cross-validations were performed on the training set using a suite of common classification algorithms including Decision Trees, KNN, Naive Bayes, SVM, and Logistic Regression. Logistic Regression has slightly better general performance, so that is used in subsequent experiments in this paper.

5.2 Prediction

For the task of prediction, a query name, that is also a name in the link data set, is picked - for instance, *Osama*. *Osama* is paired with all of the other names in the link data set. The attributes of all the pairs are then computed. The classifier predicts on all the pairs and ranks them by the class conditional scores. See Table 4.

Table 4: Hypothetical prediction data set

name1	name2	SED	...	DP	...	Classifier's Estimate $\hat{P}(alias \mid measures)$
Osama	The Prince	15	...	36		0.70102
Osama	Usama	2	...	24		0.69283
Osama	Bob	6	...	6	...	0.11451
Osama	Sid	6	...	4		0.02315
Osama	John	7	...	12		0.01204
...	...	...	...	...	...	...

6. Empirical Evaluation

In order to evaluate this algorithm, large link data sets filled with alias-rich information are needed. However, the types of people who use aliases are the ones that tend to hide from the general public. Suitable candidates thus include terrorists and spammers. The terrorist information can be obtained from newspaper articles and spam is always readily available.

Spam-based link data sets are alias-rich because spammers will often create aliases through intentional misspelling to confuse the spam filter. For example, instead of using *mortgage* in an email soliciting a loan, the spammers might use the word *m0rtg@ge* instead. While the spam filter might be confused, any human will recognize the similarity between *mortgage* and *m0rtg@ge*.

In a spam-based link data set, each spam email is represented as a separate link, with the names in that link being the word-tokens appearing in the subject header and the main body of the email. However this requires several steps of pre-processing on each email message:

1. Parse out all the HTML tags.
2. All the words are converted to tokens. All the tokens are unique, so multiple occurrences of a single word in an email will be treated as a single occurrence.

3. The tokens are filtered through a stop list of about 120 common words.
4. If a token does not appear in more than 2 emails, it is eliminated.

For example, given a spam email that looks like this

Subject:Mortgage rates as low as 2.95%

Ref<suyzvigcffl>ina<swwvvcobadtbo>nce to<shecpgkgffa>day to as low as 2.<sppyjukbywvbqc>95% Sa<scqzxytdcua>ve thou <sdzkltzcyry>sa<sefaioubryxkpl>nds of dol<scarqdscpvibyw>l< sklhxmxbvdr>ars or b<skaavzibaenix>uy the <br> ho<solbbdcq oxpdxcr>me of yo<svesxhobppoy>ur<sxjsfyvhhejoldl>eams!<br>

The final processed link might look like this:

(*mortgage*, *rates*, *low*, *refinance*, *today*, *save*, *thousands*, *dollars*, *home*, *dreams*)

6.1 Link Data Sets

This section contains the description of the three link data sets used to evaluate the algorithm. See Table 5 for size information.

Terrorists This link data set is manually extracted from a set of public web pages and news stories (often written/hosted by various governments and news organizations) related to terrorism. The names mentioned in the articles are linked subjectively upon reading the information. This data was used in Kubica *et al.* (2003).

HsiungSpam is a collection of spam emails from the author's mailbox.

ArchiveSpam is a collection of spam emails from the website *www.spamarchive.org*.

Table 5: Data sets and their size

Data set	Links	Names
Terrorists	5581	4088
HsiungSpam	373	2452
ArchiveSpam	5601	8451

6.2 Alias Selection

To gather positive training examples for learning, all the aliases for a particular entity are manually collected. Then alias pairs are generated by exhaustively matching up all aliases that belong to that entity. Each alias pair corresponds to a positive example in the training set. The following describes how the aliases are collected. See table 6 for size information.

Terrorists We collected the aliases from an open source website of the top 20 most wanted terrorists. So we have 20 entities, each having two to 14 aliases each. The entity with two aliases only generates one alias pair where as the entity with 14 aliases generates $\binom{14}{2}$ or 91 pairs of aliases. In this training set, there are 919 possible alias pairs (positive training examples).

HsiungSpam These aliases are manually generated. Below is a subset. Note that each line represents a single entity.

1. add added increase plus
2. pill pills drugs
3. viagra v1a*ra v1agra

ArchiveSpam These aliases are also manually created. Below is a subset:

1. brilliant smart intelligent
2. exercise exercises exercising
3. small little tiny mini micro

Table 6: Data sets and alias ground truth

Data set	Source	Ground Truth Entities	Alias Pairs
Terrorists	Open Source Website	20	919
HsiungSpam	Hand Labeled	21	89
ArchiveSpam	Hand Labeled	10	47

7. Empirical Results

7.1 ROC Curve Analysis

The combined classifier is tested alongside one that is built strictly from orthographic attributes and another one from semantic attributes.

K-fold cross-validation is used to evaluate all three classifiers. For each “fold”, an entity with known aliases and all related alias pairs are removed from the training set. Then each classifier is trained on the remaining training set. Since the classifiers test whether two names are aliases, a query is performed with one name of the removed entity against all possible names in the link data set (this is the prediction set). From the sorted classifier scores of the prediction set, the ranks of the correct aliases (other names of the removed entity) are identified, and a ROC curve is produced. For the next “fold,” another query on another name of the same removed entity is performed, another ROC curve is produced, and this is repeated until all the names of the entity are exhausted. When that happens, the next entity with known aliases is used.

All ROC curves are represented by first normalizing all the axes and then averaging all the curves. A good measure of performance for each classifier is the area under the averaged ROC curve (AUC).

Three average ROC curves are produced. The dash-dot curve is based on the classifier that only has attributes from the probabilistic semantic model. The dotted curve is based on the orthographic model, and the solid line curve is the combined model.

Terrorists Since the links are produced manually (although subjectively), there is relatively little noise. Therefore, the semantic classifier performed very well. The orthographic classifier performs poorly. However, the combined classifier is able to take advantage of the extra information given by the orthographic measures and outperform the semantic classifier. See Figure 1.

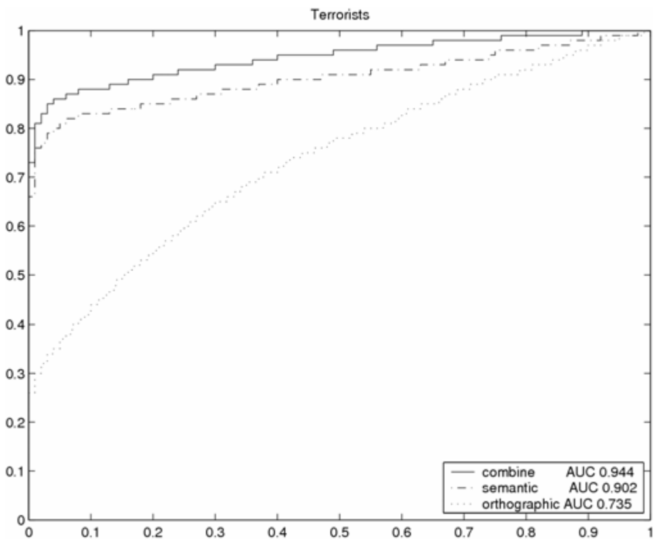

Figure 1: Average ROC for *Terrorists*

HsiungSpam Both spam data sets contain noise due to the nature of spam. In the presence of noise, the semantic classifier did not do as well. Meanwhile, the combined classifier still takes advantage of both models and produces a significantly better AUC than both. See Figure 2.

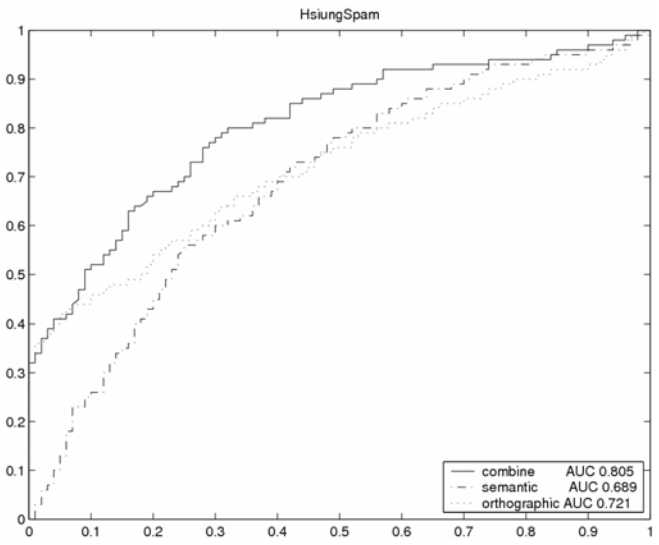

Figure 2: Average ROC for HsiungSpam

ArchiveSpam Again the combined classifier was able to obtain the highest AUC. This is almost the same as HsiungSpam, except the semantic classifier was able to outperform the combined classifier on the latter part of the curve. Nevertheless, the combined classifier performed better at the first part of the curve, which is more important than the latter part, for many applications. See Figure 3.

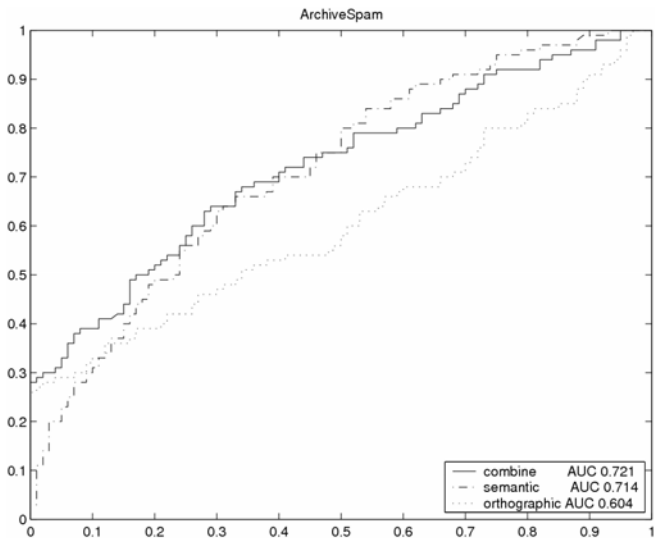

Figure 3: Average ROC for ArchiveSpam

In all three combined ROC curves, a spike occurs at the initial part of the graph. This means that in most cases, the combined algorithm is likely to rank a true alias for the query name near the top.

7.2 Training Set Degradation

To get an idea of how well the training set performs under a data shortage, training set rows are randomly removed to see how the AUC for K-fold test on *HsiungSpam* is affected. See Table 7.

Table 7: Training Set Degradation

Percentage of Original	Combine AUC	Semantic AUC	Orthographic AUC
100	0.805	0.689	0.721
75	0.803	0.705	0.721
50	0.796	0.698	0.718
25	0.777	0.682	0.709

Surprisingly, the performance degrades smoothly when the training set degrades. This is especially true in the orthographic case, where a large training set is not needed to train the classifier.

7.3 Attributes Importance

To get an informal idea of how well each attribute performs, one or two attributes are removed, and then the AUC score is recorded. See Table 8.

Table 8: AUC Degradation with Attribute Elimination

Attributes Deleted	AUC Degradation
String Edit Distance	No degradation
Normalized SED	-0.01
Exponential SED	No degradation
Discretized SED	-0.058
Normalized SED and Discretized SED	-0.083
Dot Product	-0.001
Normalized Dot Product	-0.026
Common Friends	-0.002
KL Distance	No degradation

On the orthographic side, the most important attribute is probably *Discretized String Edit Distance* and *Normalized String Edit Distance*. On the semantic side, the most important is probably *Normalized Dot Product*.

8. Related Work

Very similar to this paper, Baroni *et al*. (2002) has discussed and implemented an unsupervised algorithm that detects aliases (morphologically related words) in a text corpus using both orthographic and semantic information. On the orthographic side, they used string edit distance and on the semantic side, they used mutual information. However, to combine the two, they arbitrarily choose a function of the orthographic and semantic scores (weighting the two by hand). This is in contrast to using a learning method, which involves a much larger pool of similarity measures and which leaves the combination task to the classifier (see Section 4).

Pasula *et al*. (2002) showed a very promising probabilistic approach to resolving multiple citations of the same paper. They built a Bayesian network to represent each citation. Their solution was a well tailored domain-specific system (as it relied heavily on relationships which were specific to publications) as opposed to the more general, self tuning, system described in this paper.

Bilenko and Mooney (2003) addressed the issue of duplicated records in databases. Their approach involved training and building a classifier and is very similar to this paper. The difference is that this approach handled and exploited the semantic meaning behind the entities, where as, they were limited to orthographic similarities in their domain.

As discussed in the introduction, Sarawagi and Bhamidipaty (2002) is the most relevant related work and this paper can be viewed as its extension. However the key difference is our application towards semantic information behind link data sets.

References

M. Baroni, J. Matiasek, and H. Trost 2002. Unsupervised discovery of morphologically related words based on orthographic and semantic similarity. *Proceedings of the Workshop on Morphological and Phonological Learning of ACL/SIGPHON-2002*.

M. Bilenko and R. J. Mooney 2003. On evaluation and training-set construction for duplicate detection. *KDD-2003 Workshop on Data Cleaning, Record Linkage, and Object Consolidation*. August 2003.

A. Goldenberg, J. Kubica, P. Komarek, A. Moore, and J. Schneider 2003. A comparison of statistical and machine learning algorithms on the task of link completion. *KDD Workshop on Link Analysis for Detecting Complex Behavior*. August 2003.

M. Hernandez and S. Stolfo 1997. Real-world data is dirty: Data cleansing and the merge/purge problem. In *Journal of Data Mining and Knowledge Discovery*.

D. Jurafsky and J. H. Martin 2000. *An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice-Hall.

J. Kubica, A. Moore, D. Cohn, and J. Schneider 2003. cgraph: A fast graph-based method for link analysis and queries. *Proceedings of the 2003 IJCAI Text-Mining & Link-Analysis Workshop*. August 2003.

B. Marthi, B. Milch, and S. Russell 2003. First-order probabilistic models for information extraction. *IJCAI 2003 Workshop on Learning Statistical Models from Relational Data*.

D. B. Neill 2002. *Fully Automatic Word Sense Induction by Semantic Clustering*. M.Phil Thesis. Cambridge University.

H. Pasula, B. Marthi, B. Milch, S. Russell, and I. Shpitser 2002. Identity uncertainty and citation matching. *Advances in Neural Information Processing* (NIPS). 2002.

S. Sarawagi, A. Bhamidipaty. Interactive Deduplication using Active Learning. In *Proceedings of ACM SIGKDD-2002*, Edmonton, Alberta, 2002.

J. Zobel and P. W. Dart 1995. Finding approximate matches in large lexicons. *Software -- Practice and Experience*, 25(3):331-345.

J. Zobel and P. W. Dart 1996. Phonetic string matching: Lessons from information retrieval. *Proceedings of the 19th International Conference on Research and Development in Information Retrieval*, Zurich, Switzerland, 1996, pp. 166-172. H.-P. Frei, D. Harman, P. Schäble, and R. Wilkinson (eds). ACM Press.