Alias Detection in Link Data Sets

Paul Hsiung

CMU-RI-TR-04-22

March 2004

Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

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. This report investigates the extent to which probabilistic methods can help.

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.

Thesis Supervisor: Dr. Andrew W. Moore
A. Nico Habermann Professor

Acknowledgments

First and foremost, I would like to thank God for comfort through every affliction and His unseen hand through endless trials.

The work presented in this report is made possible from contributions of many individuals. More than any others, my interactions with my adviser, Dr. Andrew Moore, have shaped the way I think as a scientist. I appreciate his guidance, wisdom, endless energy, and kindness over the years. I am greatly indebted to him for teaching me machine learning and for opportunities to pursue research at Carnegie Mellon University.

I would like to thank Dr. Jeff Schneider for his contributions regarding the combined model and his feedback, and Daniel Neill for the invaluable ideas, suggestions, and inputs throughout the project. I want to extend my gratitude to other members of the committee who provided advice and guidance: Dr. Henry Schneiderman and James Bagnell.

I am grateful to Megan Tzeng for proofreading and hours of lost sleep. Last but not least, I would like to thank my parents whose support and encouragement I cannot do without.

Contents

Chapter 1

Introduction

The premise of a link data set is that one **entity** represents an unique individual whether it be an actual person, a word, or even a research paper. However, each entity can have many **names**. If two or more names map to one 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 or probabilistic relation between the names. The definition of relation is very 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.

The motivation for link data sets arises from the way information is presented in media such as news articles and emails. Link data sets offer a concise, general purpose, information-filled, computable form of data as opposed to a raw newspaper article. For example, consider the following web-article [1]:

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

Rather than representing this article by the entire free text, a small number of links will summarize the information. For instance, the following link can describe a terrorist cell:

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

$$(Osama\ bin\ Laden, Ayman\ al\ Zawahri, al\ Qaeda)$$

Likewise, the following links can describe location information of people:

$$(Pakistan, Osama\ bin\ Laden)$$

$$(Afghanistan, Osama\ bin\ Laden)$$

For this report, the 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 and the number of times they fall into the same link.

On the positive side, link data sets create a way for algorithms to understand human-readable information, but on the negative side, link data is susceptible to noise and often requires human overhead and subjective judgment. The use of link data is previously discussed in these papers: [2] [6].

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. In this case, aliases can occur based on misspelled names, cross-lingual transliterations (*Usama* and *Osama*), or even different titles (*Usama* might be called *The Emir*).

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 both a statistician and a sports figure as well as many others who share that name. Various methods that address this problem are discussed in Appendix A.

This report focuses on many names corresponding to a single entity. For example, *Osama bin Laden* is also known as *Usama bin Laden*. These two strings are orthographically similar. They are obviously 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 measure is computed between these two sets of friends.

This report restricts the problem to testing whether a pair of names are aliases based on orthographic information and/or semantic information. All the following models and measures are based on analysis of pairs of names in a link data set.

Chapter 2

Probabilistic Models

2.1 Probabilistic Orthographic Model

The most natural measure of likelihood that two names are aliases is orthographic similarity. If two names are very similar, they are likely to be two versions of the same name. One of the most common measures of similarity 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 [5]).

There are many possible string edit distance functions that measure similarity. The central issue is either: how to choose the best one of these measures, or, how to effectively combine multiple orthographic measures? This will be discussed in Chapter 2.3.

A suite of potential orthographic measures is described below:

String Edit Distance (SED) Again, this is 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 maximum length of the two string that are being compared.

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

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

Table 2.1: Hypothetical Example of Friends Lists

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

Exponential String Edit Distance (ESED)

$$ESED(s_1, s_2) = exp(SED(s_1, s_2))$$

There are many other possible orthographic measures mentioned in [11] and [12].

2.2 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 who have historically occurred in links) in the same historical proportions, then it is more likely (though by no means certain) that the two people are, in fact, aliases for one person. Again, the issue of how to combine the measures is discussed in Chapter 2.3.

A suite of potential semantic measures is described below:

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 2.1,

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

Normalized Dot Product (NDP) This is almost the same as *Dot Product* except each vector from each name is normalized by dividing by its magnitude before taking the dot product. In Table 2.2,

$$NDP(Osama, ThePrince) =$$

$$0.588 * 0.033 + 0.118 * 0.133 = 0.0351$$

Table 2.2: Hypothetical Example of Normalized Friends Lists

	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 1 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:

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

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

2.3 Combined Model

The previous sections have reviewed and introduced a number of measures of similarity between a pair of names that could help identify aliases. The main issue addressed in this section is how they should be combined. Four approaches to combination are considered:

1. **Manually pick the best single measure.** This "feature selection" approach does not work well because orthographic and semantic measures convey different but useful information. This claim is backed up 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. On the one hand, the model might be humanly understandable, but on the other hand, 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 [1], who used weighted sum, and [4], who manually created rules.

3. **Probabilistic model** Another approach to this problem is a full, probabilistic, generative model of links, names, and the string corruption process. Under the assumptions 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 is a very promising approach. [7] gives a very detailed description of what such a generative model looks like in the related area of bibliometrics analysis. Future work in this area is likely to continue and to be productive though both the representational and the computational challenges will be severe.

4. **Semi-supervised learning** This approach requires a human to provide a relatively small set of both 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 report.

Chapter 3

Alias classifier

3.1 Training

In this combined semi-supervised 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, the user gathers a collection of name pairs that includes positive and negative examples. Positive examples come from human-selected aliases. Because the pool of names is so large and the number of true aliases is small, the user can, with a high degree of safety, pick negative examples by randomly selecting pairs of names among the link data set with a very low chance of picking a true pair of aliases. For each pair of names, all of the measures in both the semantic and orthographic models are calculated and added as **attributes** into the training set. An attribute called *Alias?* is created to label each example as positive or negative. Each pair of names with all its attributes represents a row in the training set. For example, see Table 3.1.

Having framed the training set this way, the problem of alias detection is transferred to the more familiar world of straightforward classification. A series of cross-validation 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 it is used in subsequent experiments in this report. However, it is possible, in follow-up work, to investigate a wider range of classifiers. The space of possible classifiers is somewhat constrained by the requirement that they must rank order their predictions, so they need some measure of conditional class probability in addition to a straight classification.

Table 3.1: Hypothetical training set

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

Table 3.2: Hypothetical prediction data set

name1	name2	SED	...	DP	...	Classifier's Estimate $\hat{P}(alias\|measures)$
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
...	...	...	...	...	...	...

3.2 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 for all the created pairs is then computed. The classifier predicts on all the pairs and ranks them by the class conditional scores from the classifier. For an example prediction data set, see Table 3.2.

Chapter 4

Empirical Evaluation

In order to evaluate the algorithms, large link data sets filled with alias-rich information are needed. Because of the large amount of data needed, information that can be gathered on a daily basis such as newspaper articles and emails are used. However, the type of people who use aliases are the ones that tend to hide from the general public. Bearing this in mind, two obvious candidates for the link data sets are terrorists and spammers. The terrorist information can be obtained from newspaper articles, and of course, spam is readily available in anyone's email system.

Spam-based link data sets are alias-rich because spammers often try to create aliases through intentional misspelling to confuse the spam filter. For example, instead of using *mortgage* in a email soliciting a loan, the spammers might use the word *m0rtg@ge* instead. While some spam filters might be confused, any human being 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 with white spaces or new lines being the boundary. 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. Some spammers like to add noisy and nonsensical words. To counter this, if a token does not appear in more than two 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<solbbdcqoxpdxcr>me of yo<svesxhobppoy>ur
dr<sxjsfyvhhejoldl>eams!<br>
```

The final processed link might look like this (given that all these tokens appeared in more than two other emails):

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

4.1 Link Data Sets

This section contains the description of the three link data sets used to evaluate the algorithms. See Table 4.1 for the size information of each data set.

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 set is also used in [6].

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*. The setup is the same as *HsiungSpam*.

Table 4.1: Data sets and their size

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

Table 4.2: Data sets and alias ground truth

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

4.2 Alias Selection

To gather positive training examples for the semi-supervised learning, all the aliases for a particular entity are collected by a human or some other external source. 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 from the three link data sets (also see Table 4.2):

Terrorists The aliases are collected from the FBI website of the top 20 most wanted terrorists. Therefore, there are 20 entities, with each having two to 14 aliases each. The entity with two aliases only generates one alias pair whereas 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.

- add added increase plus
- big huge large massive
- pill pills drugs

- viagra v1a*ra v1agra
- fr—ee free freee

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

- brilliant smart intelligent
- exercise exercises exercising
- small little tiny mini micro

Chapter 5

Empirical Results

5.1 ROC Curve Analysis

ROC curves and area under the ROC curves(AUC) are used to represent the results. For more information on these tools, see [3].

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

K-fold cross-validation is used to evaluate all three classifiers. For each "fold," an entity 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 of each other, 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 classifier scores of all possible names, the scores are sorted, the ranks of the correct aliases (other names of the removed entity) are created, 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 run out of names. When that happens, the next entity is used.

All ROC curves are represented by first normalizing all the axes from zero to one and then averaging them together. The legends of these charts contain the AUC.

Three average ROC curves are produced. For Figures 5.1 to 5.3, the dash-dot curve represents 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 performs 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 5.1.

HsiungSpam Both spam data sets contain noise due to the nature of spam. In the presence of noise, the semantic classifier does not perform as well. Meanwhile, the combined classifier still takes advantage of both models and produces a significantly better AUC than either individually. See Figure 5.2.

ArchiveSpam Again the combined classifier is able to obtain the highest AUC. This is almost the same as *HsiungSpam*, except the semantic classifier is able to outperform the combined classifier on the latter part of the curve. Nevertheless, the combined classifier performs better at the first part of the curve, which is more important than the latter part, for many applications. See Figure 5.3.

In all three combined average 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.

5.2 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 5.1 for results.

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

5.3 Run time

To see how the algorithm scales, links are randomly removed from *HsiungSpam*. Figure 5.4 shows the length of each run given the reduced number of links. As one can see from the graph, the algorithm is linear with respect to the number of links.

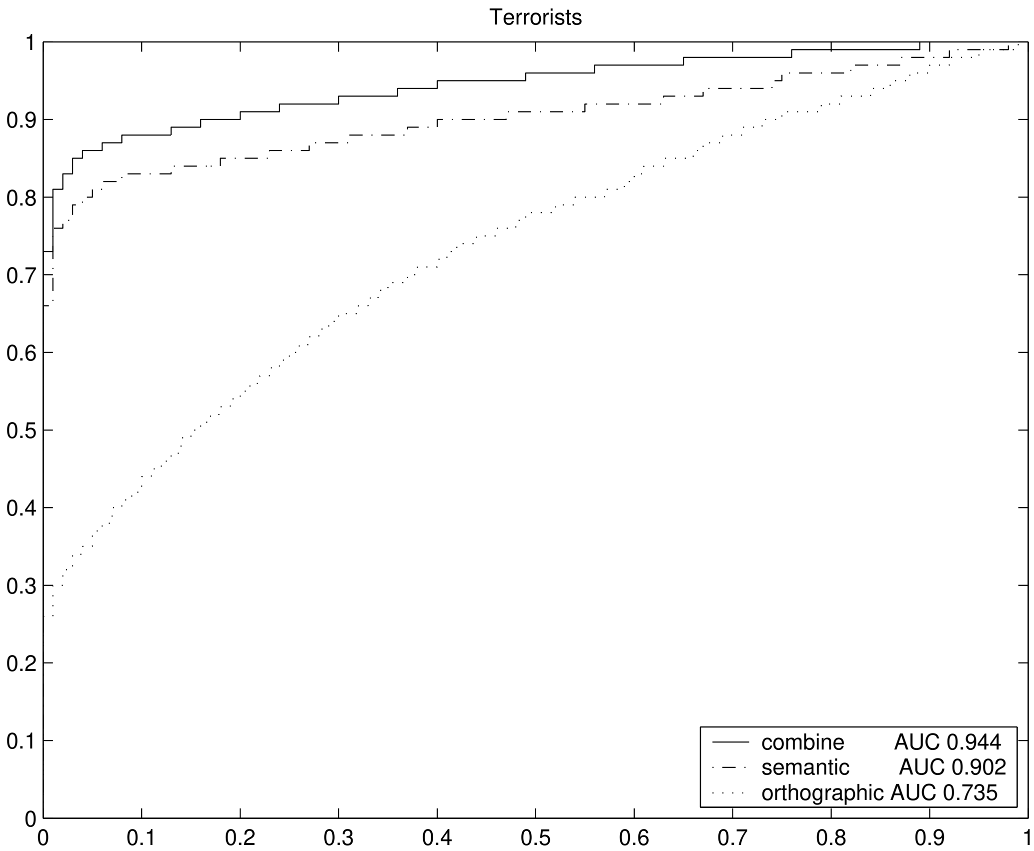

Figure 5.1: Average ROC for *Terrorists*

Table 5.1: 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

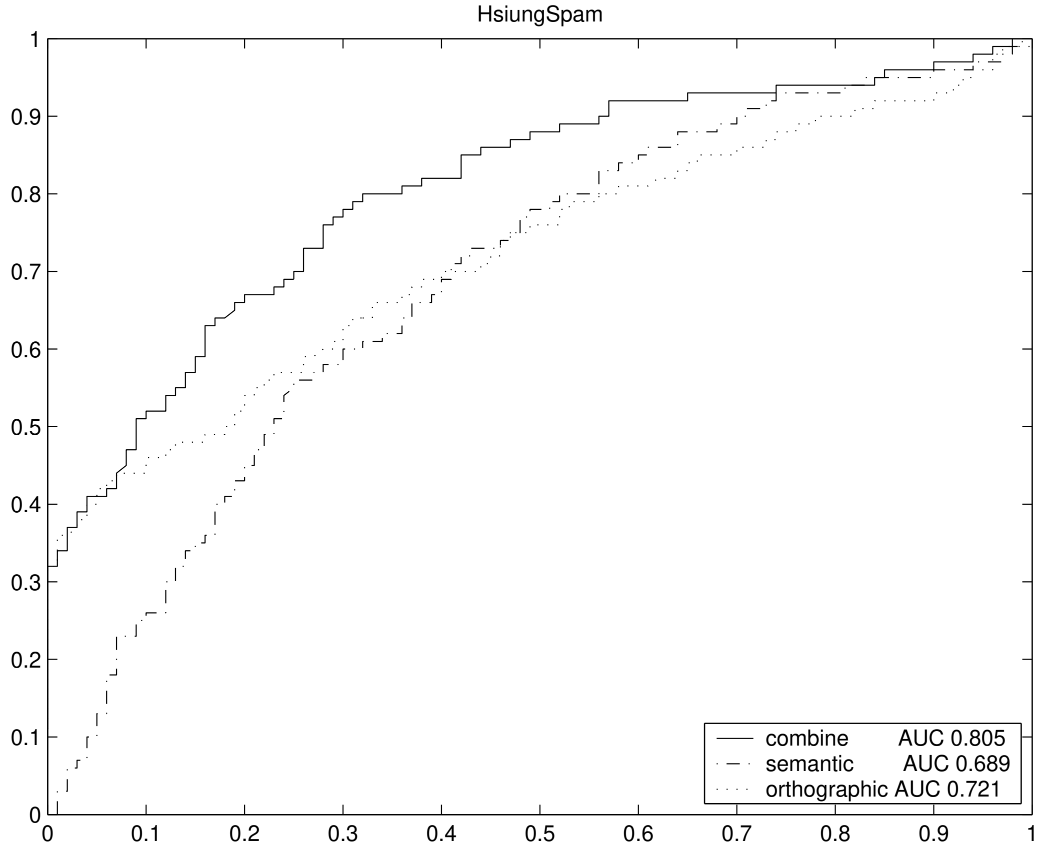

Figure 5.2: Average ROC for *HsiungSpam*

This is consistent with the expectations for computational complexity where the dominant costs for finding the aliases of the query name E are iterating over all the other names in the link data set and computing the suite of similarity measures with the other names. Each computation of similarity measures for a pair of names takes $O(F)$ where F is the typical number of friends of a name, and where computations are arranged to take advantage of the sparseness of the link graph. Thus computing all the measures between E and all other names is $O(N\ F)$, where N is the number of names in the link data. At the end, the names are sorted by alias probability so the overall cost is

$$O(N\ F + N\ log(N))$$

but only the former term is dominant in practice.

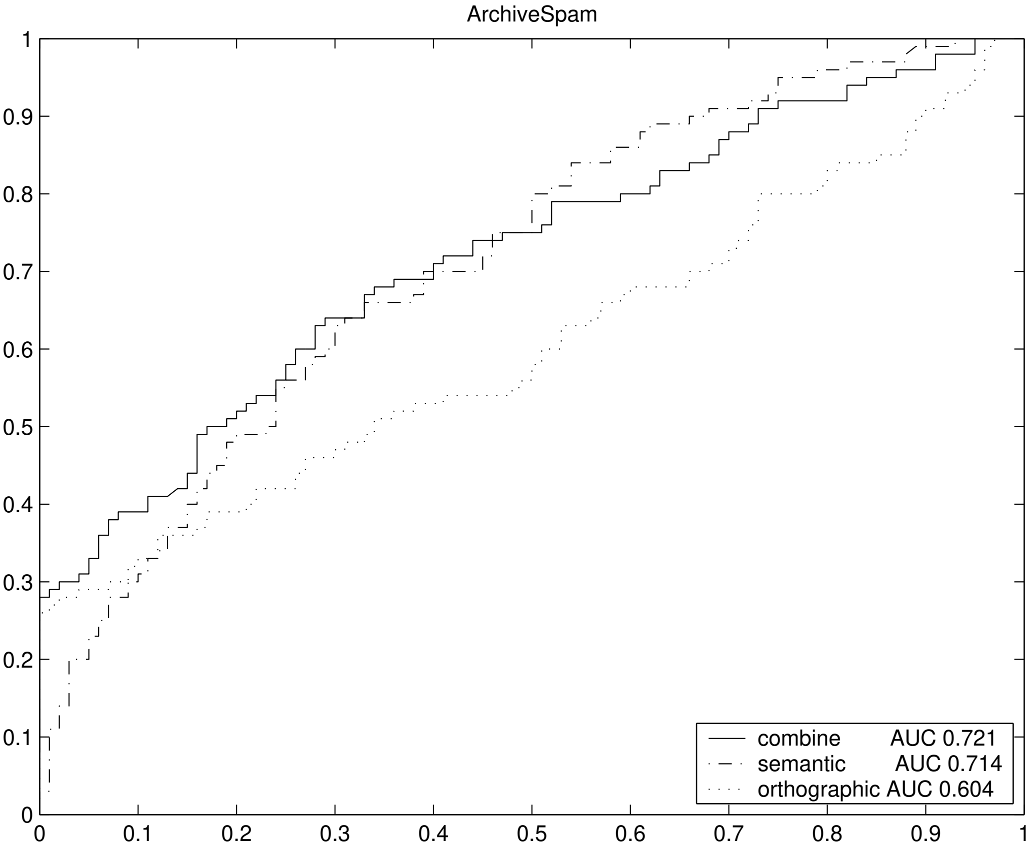

Figure 5.3: Average ROC for *ArchiveSpam*

5.4 Attribute Importance

To get an general idea of how well each of the measures performs, one or two attributes from the model are removed, and then the AUC score is recorded. See Table 5.2.

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

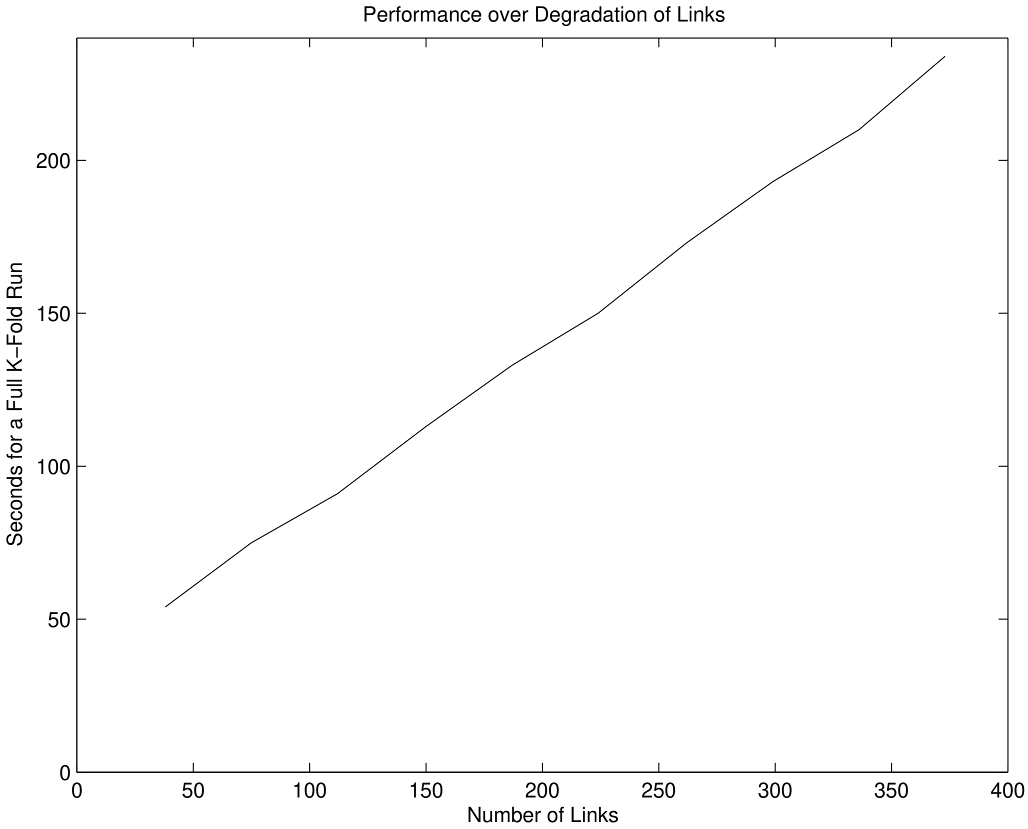

Figure 5.4: Scalability

Table 5.2: AUC Degradation with Attribute Elimination

Attribute(s) Deleted	AUC Degradation
String Edit Distance (SED)	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

Chapter 6

Related Work

This report is very similar to [1], who 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 use string edit distance, and on the semantic side, they use mutual information. However, to combine the two, they arbitrarily choose a function of the orthographic and semantic scores (weighing the two by hand). This is in contrast to this report's semi-supervised learning model, which involves a much larger pool of similarity measures and which leaves the combination task to the classifier (see Chapter 2.3).

[4] addresses the problem of detecting repeated records in databases, which might have resulted from spelling mistakes or wrong digits for ID numbers. Their solution involves scanning through the database in a particular, user-defined, sorted order and detecting repeated records based upon manually predefined rules, such as:

```
Given two records, r1 and r2.
IF the last name of r1 equals the last name of r2,
       AND the first names differ slightly,
       AND the address of r1 equals the address of r2
THEN
       r1 is equivalent to r2.
```

Differ slightly means that they compare strings orthographically through string edit distance. Their rule set is implemented with 60 lines of C code. The key difference between [4] and this report is that this classifier automatically learns decision policy to determine whether two names are aliases of each other. This learning can be application-dependent.

[10] shows a very promising probabilistic approach to resolving multiple citations of the same paper. They build a Bayesian network to represent each citation. Their solution is a well tailored domain-specific system (as it relies heavily on relationships that are specific to publications) as opposed to the more general, self tuning system described in this report.

Bibliography

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

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

[3] J. A. Hanley and B. J. McNeil. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143(1):29–36, April 1982.

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

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

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

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

[8] D. B. Neill. Fully Automatic Word Sense Induction by Semantic Clustering. M.Phil Thesis, Cambridge University, 2002.

[9] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm, 2001.

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

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

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

Appendix A

Name Disambiguation

Many different people have the same name. In a normal setting, people often use their background knowledge to disambiguate different people from a common name. A transactional database can contain thousands of common names, however, making it infeasible to manually obtain background information for every common name.

Nonetheless, more information can be obtained from the context surrounding the common name. Suppose the context is link data set and friends of each name is known, partitions can be formed from among the friends of the common name. The assumption is that if different people use a common name, there will be distinct partitions among friends of that name.

For example, the name Michael Jordan is used by many different people. Some will know Michael as a basketball star, and some will know him as a statistics professor. Supposing the background information (basketball star, statistics professor) is not given, disambiguation still can be done by knowing about the friends of the name. Michael Jordan, as a basketball star, will be closely associated with Scottie Pippen, the Washington Wizards, and other basketball entities. However Michael Jordan the statistics professor will be closely associated with names of graduate students, postdocs such as David Cohn, and University of California at Berkeley. Just by observing Michael Jordan's friends, there will be a clear partition between the friends of the basketball star versus the friends of the professor.

A.1 Approach

In all the following approaches, the inputs are a query name and a symmetric matrix with each row or column representing a name. The number inside each cell of the matrix represents how frequently the two names occur together in a given context. For example, if the data set is *newspaper articles*, row i can be George W. Bush, column j can be Dick Cheney, and $cell(i, j)$ is how many times these two names appear together in *newspaper articles*. This matrix can also be represented as a weighted graph with the vertices being names and the weighted edges being the number of times two names appear together.

The first approach is to select two friends of the query name that will split all the friends of the query name into two clusters, with each selected friend in a separate cluster. From these two selections, probabilities that each friend belongs to each cluster can be calculated iteratively. This approach is based on [8].

The second approach also starts with the selection of the two friends. However, the other friends get assigned to each cluster based on their distance to the two selected friends. Distance is defined as the length of the shortest path between two vertices in the graph.

The third approach is Spectral Clustering. First, the matrix is scaled down to two dimensions by projecting it onto the first two eigenvectors. Then all the points (friends) are normalized onto a unit sphere. Finally, K-means is used to cluster the projected and normalized points into two clusters. This approach is based on [9].

A.2 Results

These three approaches have been tested on link data sets from movie databases, news articles, research papers, cocktail mixes, and CMU Robotics Institute web pages. To simulate two different individuals having a common name, two random names are selected from the link data set and merged into one name. Then, all the links of these two names are collected. From these links, all the friends of these two are extracted. For example, links can be news articles, and the two individuals can be both Michael Jordans. All the friends are Scottie Pippen, U.C. Berkeley, and so on.

As described above, each approach will assign clusters for each friend. Now a link that contains the name in question can also be assigned a cluster by the friends contained in the link, so that all the links can be partitioned into two clusters. The merge of two individuals can be undone to see how well the clustering performed. Back to the Michael Jordan example, if the clustering is completely successful, all the news articles that contain Michael Jordan as a basketball star will be in one cluster, and conversely, all the news articles that contain Michael as a professor will be in another cluster. If the clustering is completely random, each cluster will contain some basketball articles and some statistics articles.

Scoring is done by testing how "pure" are each clusters. 50% means completely random and 100% means perfect clustering. Note that because of this metric, scores cannot go below 50%. "Max Friends" means how many top friends were chosen for that test. Each test score is the average of one hundred runs.

Table A.1 shows that the first approach has the best scores over most of the data sets. The second approach has the top score only in the *Cocktail Mixes* data set, but it loses to the first approach in all other data sets. The third approach has the worst performance. This is because spectral clustering is susceptible of making a single friend as an entire cluster.

Table A.1: Performance of each three approaches

	Max Friends			
	10	100	1000	10000
First Approach				
Movie Actors	0.58	0.61	0.78	0.78
News Articles	0.50	0.54	0.67	0.69
Research Papers	0.61	0.74	0.75	0.75
Cocktail Mixes	0.54	0.52	0.52	0.52
CMU RI webpages	0.54	0.73	0.74	0.74
Second Approach				
Movie Actors	0.50	0.54	0.78	0.78
News Articles	0.50	0.55	0.59	0.59
Research Papers	0.58	0.72	0.72	0.72
Cocktail Mixes	0.55	0.63	0.62	0.62
CMU RI webpages	0.51	0.64	0.66	0.66
Third Approach				
Movie Actors	0.51	0.55	0.58	0.60
News Articles	0.51	0.54	0.59	DNE
Research Papers	0.55	0.64	0.64	0.64
Cocktail Mixes	0.55	0.65	0.65	0.65
CMU RI webpages	0.54	0.67	0.65	0.65