Using Genetic Programming to Evaluate the Impact of Social Network Analysis in Author Name Disambiguation

Felipe Hoppe Levin and Carlos A. Heuser

Instituto de Informática, Universidade Federal do Rio Grande do Sul (UFRGS),
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil
{fhlevin, heuser}@inf.ufrgs.br

Abstract. In digital libraries, which have become extremely popular in the scientific community, often people want to find publications by an author using the author name as a query. However, since authors may have many denominations and one denomination may refer to many authors, name searches may present ambiguous results. To tackle this problem, several studies have been developed. Recently the use of social networks has been studied in author name disambiguation. In this article, we use a machine learning approach based on Genetic Programming to evaluate the impact of social network analysis in author name disambiguation. Through experiments using real-world data, we show that social network analysis greatly improves the quality of results. Also, we demonstrate that our approach is able to compete with state-of-the-art techniques.

Keywords: name disambiguation, relationship analysis, social networks, genetic programming, digital libraries.

1 Introduction

Digital Libraries (DLs) are complex information systems for storing and presenting online collections of information. A DL provides services for searching and browsing its collection and stores metadata that describes its content (e.g., author, publisher) as well as the relationships between its data. It is constructed, collected and organized with the goal of supporting the information needs of a specific community [3].

DLs have become an important source of information for the scientific community by presenting a centralized interface for searching and browsing publications. By grouping publications by metadata such as author, topic and publishing venue, users may employ the content of DLs for distinct analysis, such as coverage of topics, or evaluating a researcher's production.

When using DLs, users often assume that its content is free of errors and ambiguities. However, DLs gather data from different sources which often use different standards and abbreviations, leading to ambiguities. One of the most common is *name ambiguity* as there is a many-to-many relationship between persons and their denominations. A person may have many denominations, since first names may be abbreviated and middle names may be omitted. Also, different persons may share the same denomination. For example, two authors, *Mark Jones* and *Matthew*

Jones, may have their names abbreviated to *M. Jones*. A search for *M. Jones* would present these publications as belonging to the same author, leading to a problem known as *mixed citation* [12]. However, while some of *Mark Jones's* production is under the name *M. Jones,* other publications could be found at the author's full name and a search of *Mark Jones* would not present the author's complete production, leading to the *split citation* problem [12].

The *name disambiguation* problem has been subject to several studies and many methods and heuristics have been developed. Traditional methods compare syntactic attribute information (eg., name, title, venue) between ambiguous objects and, by using complex match functions, determine which objects represent the same real entity. In [13], we used social network analysis as an evidence for the disambiguation process, showing that by using such evidence the quality of results is improved.

A social network is a collection of people – or actors – where each actor is tied to a subset of the others [16]. In DLs, actors are *authors* which are tied when they have co-authored a publication. Collaboration between two authors implies an affinity between them: they may be interested in the same area or be affiliated to the same institution [15]. If the distance between these two authors in the network is small, they have a greater chance of having the same interests and being affiliated to the same institution and therefore have a greater chance of representing the same entity.

In [13], we evaluated the impact of adding social network analysis to traditional disambiguation methods based on author name similarity, demonstrating it significantly improves the quality of results. In this article we continue this research by showing that the use of social network analysis also improves methods based not only on author name similarity but in other evidences such as title and venue. To combine these evidences and the social network measures, we use a machine learning approach to create match functions for the disambiguation process. This approach is based on Genetic Programming (GP) which has been successfully used in author name disambiguation [4]. We show that, even when using other evidences such as venue and title, social networks continue to provide a significant improvement on quality. Also, we compare our results with those obtained by a related work [5], and show that our approach can compete with state-of-the-art methods.

The main contributions of this article are the following:

(1) presenting a machine learning approach for generating match functions using Genetic Programming,

(2) evaluating the impact of adding social network analysis to methods based on attributes such as name, venue and title,

(3) showing through experimental results that our approach can compete with a state-of-the-art method by comparing results obtained over real datasets.

This paper is organized as follows. Section 2 presents the concept of Author Social Network and its use in name disambiguation. Section 3 presents the GP algorithm for generating Match Functions. Section 4 describes the experiments performed in order to evaluate our approach. Section 5 covers related work. The paper is concluded in Section 6 with a description of future work.

2 Author Social Network

Usually, the input for a disambiguation process is a list of records representing paper references. In Fig. 1 we show a list of such records representing papers written by two or three authors each. This list can be represented as an Author Social Network (ASN), shown as a graph in Fig. 2, where nodes represent authors (square boxes) and papers (round boxes). Straight edges link a paper to its authors while dotted edges link two authors with the same initial letter and the same last name. Notice that the same person may author many papers and will be represented multiple times in the graph. Authors with the same initial and last name have a high possibility of being the same person and therefore our heuristic – others could be used – creates dotted edges to represent paths in the graph and establish relationships between other authors.

```
<P1; Robert Walker; Ben Goldman; Carl Parker>
<P2; Carl T. Parker; Robert D. Walker; George S. Brown>
<P3; Ben Goldman; Ruth Adams>
<P4; Rob Walker; Ruth Adams; George Brown>
```

Fig. 1. A list of records representing papers.

When comparing authors in the disambiguation process, evidences in the ASN, combined with other evidences such as the author name, may be used to assess if two authors are the same real person. In Fig. 2, authors P1.1 and P2.2 have very similar names but the ASN provides more evidence that these authors are the same person: there is a path linking them with length two (length is defined as the number of author-paper-author links in the path), which means they are closely related. As it was demonstrated in [13], this evidence means the two authors are much more likely to be the same person. Authors 1.1 and 4.1 also have similar names, but the path between them has length three. In [13], we demonstrated that relationships with lengths greater than two do not provide strong evidence that two authors are the same person.

In [13], we presented a set of relationship metrics to be used as evidence in matching authors in the disambiguation process. One of these metrics is *Relationship Distance (RD)*. In social networks, the distance between two actors on the network is the length of the shortest path between them [16]. As stated earlier, path length is defined as the number of author-paper-author links in the path. Therefore, *RD* is defined as follows:

Relationship Distance (RD). Let a_1 and a_2 be two authors. Then, RD(a_1, a_2) is the length of the shortest path between them, returning 0 if no path exists.

In our example, P1.1 and P2.2 are linked by two paths, the one that goes through P1-P2, with length 2, and the one which goes through P1-P3-P4-P2, with length four. Therefore, RD(P1.1, P2.2) is two, which is the length of the shortest path, P1-P2. This metric measures the importance of the relationship, since shortest distances mean authors are more closely related to one another.

Another relationship measure is the *Relationship Existence*, which returns true when there is a path between two authors at a minimum distance *d*.

Relationship Existence (RE). Let a_1 and a_2 be two authors being compared and *d* an integer. Then, RE(a_1, a_2, *d*) is true if $1 \leq \text{RD}(a_1, a_2) \leq d$, and false otherwise.

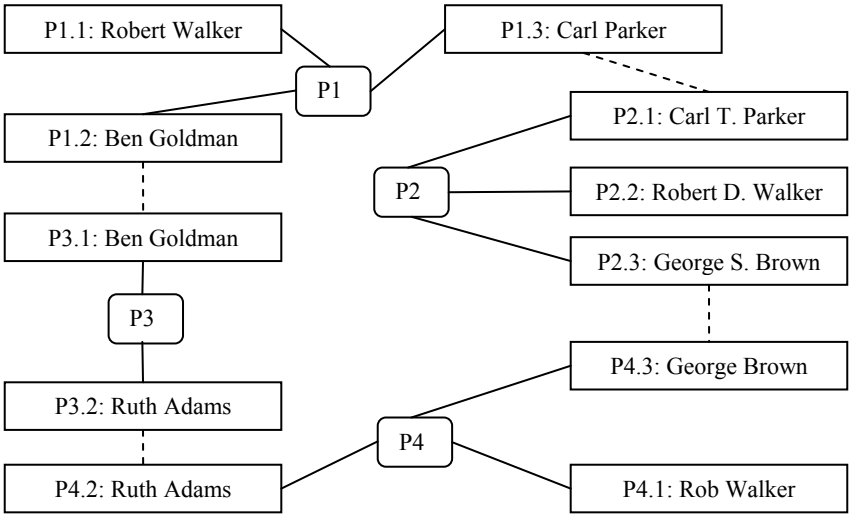

Fig. 2. A list of records representing papers.

In our example ASN, RE(P1.1, P2.2, 2) is true, since there is a path of length two linking them. However, RE(P1.1, P4.1, 2) is false, since both paths linking them have length three.

Relationship Quantity is the number of authors related to a specific author at a minimum distance *d*.

Relationship Quantity (RQ). Let *a* be an author, A the set of authors in the dataset and *d* an integer. Then, RQ(*a*, *d*) = |B|, where for all $b \in B$, $1 \leq RD(a, b) \leq d$ and $B \subset A$.

In our example in Fig. 2, P1.1 is related to all the authors in the network, but at different distances. Therefore, RQ(P1.1, 1) is four, since P1.1 is only related to four authors at distance one, P1.2, P1.3, P2.1 and P3.1. At distance two, P1.1 is related to five more authors, P3.2, P4.2, P2.3, P2.2 and P4.3, and therefore RQ(P1.1, 2) is nine. *RQ* is used to measure the connectivity of an author, i.e. how likely an author will have other authors related to it. In the disambiguation process, we may choose not to use relationship measures in authors with very low connectivity, since few or no authors are related to it. Also, when an author has a very high connectivity, *RE* loses its value as an evidence of duplicate authors, since a considerable amount of the network is linked to this author, most of which are not duplicates.

The last relationship measure is *Relationship Strength*, which is the number of paths between two authors at a maximum distance *d.*

Relationship Strength (RS). Let a_1 and a_2 be two authors and *d* an integer. Then, RS(a_1, a_2, *d*) is the number of paths between a_1 and a_2 with length *d* or lower.

In Fig. 2, there are two paths between P1.1 and P4.1, both having length two. Therefore, RS(P1.1, P4.1, 2) is two. The greater the *RS* between authors, the stronger relationship between them, meaning they have a greater chance to be duplicates.

3 Creating Match Functions with Genetic Programming

To determine if two authors are the same person, we must use a match function (MF). A MF, as defined in [1], is a function which takes two objects as input, returning true if they are duplicates and false otherwise. Also, a MF uses evidences in order to match these inputs. An *evidence* is, for example, the similarity value between author names or the existence of a relationship between them. Since many evidences may be used and it is difficult to determine the weight and threshold for each one, in this article we use GP to generate these MFs. Our GP algorithm combines a set of evidences randomly into MFs and, in an iterative process, improves these functions through a series of generations.

Table 1. Evidences Used to Generate Match Functions

Evidence	Operators	Values
Name Similarity (NameSim)	≥, ≤	0 to 1
Title Similarity (TitleSim)	≥, ≤	0 to 1
Venue Similarity (VenueSim)	≥, ≤	0 to 1
Initial Letter and Last Name Match (IniLastName)	=	0 or 1
Number of Title Words Match (NumTitleWords)	≥, ≤	0 to 8
Title Word Similarity (TitleWordSim)	≥, ≤	0 to 1
Is First Name Abbreviated (IsAbbrev)	=	0 or 1
Relationship Existence (RE)	=	0 or 1
Relationship Strength (RS)	≥, ≤	0 to 10
Minimum Relationship Quantity (MinRQ)	≥, ≤	0 to 10
Maximum Relationship Quantity (MaxRQ)	≥, ≤	0 to 10

Table 1 shows the evidences used to generate the MFs. Name, title and venue similarity compare these attributes using trigram similarity [6], which returns a value between 0 and 1, the higher, the more similar. The *IniLastName* evidence returns 1 (true) if the first letter and the last name of the author names being compared are the same and 0 (false) otherwise. The *NumTitleWords* evidence returns the number of equal title words in two titles being compared, while the *TitleWordSim* normalizes this number by the number of words in the biggest title. The *IsAbbrev* evidence returns true when the first name on one of the authors being compared is abbreviated and false otherwise. The *RE* and *RS* evidences were defined in the previous chapter and *MinRQ* returns the minimum *RQ* of two authors being compared while *MaxRQ* returns the maximum *RQ*. We used $d = 2$ in all relationship measures.

In Fig. 3, we show the GP algorithm used to generate MFs. In the first step (line 6) the initial population of MFs is generated. Each MF is generated randomly, but it follows a structure as shown in Fig. 4. According to [10], there are three requirements in order to properly use the GP technique: the problem must be modeled as a tree structure; the modeled tree must be automatically evaluated; the evolutionary operations applied over the tree must result in a valid tree. The tree structure shown in Fig. 4 fulfills the first requirement. Each evidence comparison (the 'evid' node) is composed by an evidence, an operator and a value, for example, $RE = 1$. We show the valid operators and values for each evidence on Table 1. The evidence comparisons are linked through 'and' operators, forming an 'and' group. And each 'and' group is

linked through ‘or’ operators, forming the MF. Therefore, a valid would be (*RE* = 1 and *NameSim* ≥ 0.5) or (*NameSim* ≥ 0.9), for example.

```
1.  GenerateMF(popSize: int, maxGen: int)
2.   i, j: int
3.   pop: array of match functions
4.   eval: array of double
5.  Begin
6.   pop ← GenerateInitialPopulation(popSize)
7.   For i from 1 to maxGen do
8.          For j from 1 to popSize do
9.                 eval[j] ← EvaluateFunction(pop[j])
10.         Selection(pop, eval)
11.         Crossover(pop)
12.         Mutation(pop)
13.  For j from 1 to popSize do
14.                eval[j] ← EvaluateFunction(pop[j])
15.  Return Order(pop, eval)
16. End
```

Fig. 3. GP algorithm to generate match functions.

After the population is initialized, every MF is evaluated using a fitness function (lines 8 and 9). In GP, a fitness function is used to evaluate individuals, selecting the best fitted for the next generation and discarding the rest, which is done in the Selection phase, in line 10. By running the disambiguation process with a specific MF and measuring the quality of results, we can automatically evaluate the MF, thus fulfilling the second requirement.

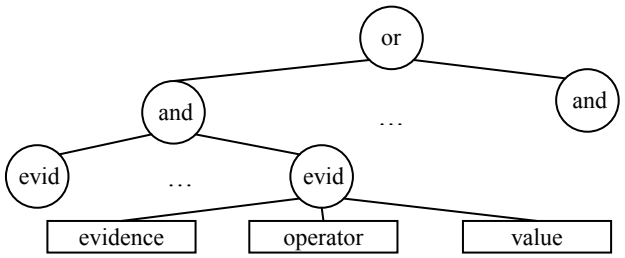

Fig. 4. Match Function Structure.

Next, the Crossover phase (line 11) creates new individuals by combining pairs of MFs. In our algorithm, this is done by randomly selecting one ‘and’ group from one individual and exchanging it by an ‘and’ group from another individual. The Mutation phase (line 12) randomly chooses individuals selected on the Selection phase and produces new individuals by introducing random mutations. This mutations create, remove or change an ‘and’ group, create, remove or change an evidence comparison or change a value node inside an evidence comparison. Both Crossover and Mutation operations create valid trees, fulfilling the third and last requirement.

The Selection, Crossover and Mutation phases run on a loop until a maximum generation is reached. The final population is then presented, ordered by its fitness.

4 Experimental Results

4.1 Datasets

In order to evaluate our approach we used 12 different datasets, 11 of which were extracted from the DBLP[1] digital library and one which was extracted from the BDBComp[2] digital library. The DBLP datasets and the BDBComp dataset have been used in [5] and [17], respectively, and were made available to us by the authors of those works. The BDBComp dataset is made up of 361 citations to papers first authored by people with the most frequent last names in the library, having 674 duplicate first author pairs. The DBLP datasets, shown on Table 1, are made up by citation to papers first authored by persons with some of the most frequent last names in the library. Every first author name in a dataset has the same initial letter and a common last name. For example, in dataset 'agupta', every first author name starts with 'a' and has 'Gupta' as a surname.

Table 1. DBLP Collection – Number of publications per dataset.

Dataset	Publications
agupta	576
akumar	243
cchen	801
djohnson	368
jmartin	112
jrobinson	171
jsmith	924
ktanaka	280
mbrown	153
mjones	260
mmiller	405

In our experiments, only the first author of each paper was disambiguated, since only first authors were hand-clustered by the datasets' creators. However, all co-authors were used to create the ASN.

4.2 Evaluation Measures

To evaluate the quality of the MFs generated, we used the following measures, defined in [11]: Average Cluster Purity (ACP), Average Author Purity (AAP), and K, which is the geometric mean between ACP and AAP. ACP evaluates the purity of generated clusters, i.e. whether the generated clusters include only records belonging to the reference clusters. The more pure the generated clusters, the closer to 1 ACP will be. The formula for ACP is:

[1] http://dblp.uni-tier.de/
[2] http://www.ldb.dcc.ufmg.br/bdbcomp/

$$\text{ACP} = \frac{1}{N} \sum_{i=0}^{q} \sum_{j=0}^{R} \frac{{n_{ij}}^2}{n_i} \quad (1)$$

where R is the number of reference clusters;
N is the total number of citation records in the dataset;
q is the number of clusters generated;
n_{ij} is the number of elements in generated cluster i belonging to reference cluster j;
n_i is the number of elements in generated cluster i.

AAP measures the level of fragmentation in the generated clusters in comparison to the reference clusters. The closer the value to 1, the less fragmented the generated clusters are. The formula for AAP is:

$$\text{AAP} = \frac{1}{N} \sum_{i=0}^{R} \sum_{j=0}^{q} \frac{{n_{ij}}^2}{n_j} \quad (2)$$

where n_j is the number of items in reference cluster j.

The K measure combines both AAP and ACP by calculating the geometric mean between them, expressed as $K = \sqrt{\text{AAP x ACP}}$. The best situation occurs when both AAP and ACP equals one.

4.3 Experiments

In the experiments, we generated three MFs using the GP algorithm from Section 3 to evaluate the impact in the quality of results by using evidences found in the ASN. The first function, called *NoNetworkMF*, does not use Social Network evidences, and is used as the baseline. The second function, *ExistenceOnlyMF*, adds only the *RE* evidence, while the third function, *FullNetworkMF*, used all Social Network evidences. Table 2 shows which evidences were used to generate each function.

To generate the MFs, we used the 'akumar', 'jsmith', 'ktanaka' and the BDBComp datasets as the training set, with a total of 1852 records. The rest of the datasets were used as the evaluation set, with a total of 2846 records. In the GP algorithm, we used the average K measure in the four training datasets as the fitness function. As parameters, we used a population of size 20 and 300 maximum generations. Since the algorithm makes random choices and is not deterministic, each MF was trained five times and the one with the best results on the training set was picked.

Next, we show the generated *NoNetworkMF*:

(NameSim ≥ 0.57 and VenueSim ≥ 0.85 and IniLastName = 1) or (NumTitleWords ≥ 7 and VenueSim ≥ 0.43 and IniLastName = 1) or (NameSim ≥ 0.94) . (2)

The generated MF shows that the name is the most important evidence and is used in every 'and' group of the function through the *NameSim* and *IniLastName* evidences. When names are almost identical (similarity greater than 0.94) no other evidence is used, otherwise venue and title evidences are used along with the name.

Table 2. Evidences Used to Generate each Match Function

Evidence	NoNetworkMF	ExistenceOnlyMF	FullNetworkMF
Name Similarity	Yes	Yes	Yes
Title Similarity	Yes	Yes	Yes
Venue Similarity	Yes	Yes	Yes
Initial Letter and Surname Match	Yes	Yes	Yes
Number of Title Words Match	Yes	Yes	Yes
Title Words Similarity	Yes	Yes	Yes
Is First Name Abbreviated	Yes	Yes	Yes
Relationship Existence	No	Yes	Yes
Relationship Strength	No	No	Yes
Minimum Relationship Quantity	No	No	Yes
Maximum Relationship Quantity	No	No	Yes

Next, we show the generated *ExistenceOnlyMF*:

(TitleWordSim ≥ 0.72 and IniLastName = 1) or (NameSim ≥ 0.4 and RE = 1 and IniLastName = 1) or (NameSim ≥ 0.97 and IsAbbrev = 0 and IniLastName = 1) or (IsAbbrev = 1 and IniLastName = 1 and VenueSim ≥ 0.39 and NameSim ≥ 0.71 and RE = 0 and TitleWordSim ≥ 0.28) or (TitleWordSim ≥ 0.49 and NameSim ≥ 0.45 and IsAbbrev = 1) or (TitleSim ≥ 0.59 and RE = 0 and IniLastName = 1) or (TitleSim ≥ 0.22 and IsAbbrev = 1 and IniLastName = 1 and VenueSim ≥ 0.39 and NameSim ≥ 0.7 and RE = 1) . (**2**)

In this MF, the name continues to be the most important evidence. But by adding the *RE* evidence we can see that related authors need less similar names to match while unrelated ones need more similar names or other evidence like similar venues and titles to match. Also, the *IsAbbrev* evidence, which was not picked in the previous function, appeared in *ExistenceOnlyMF*. In this function, when names are not abbreviated and are almost the same (similarity greater than 0.97), they match, but when they are abbreviated they need other evidence, like *RE*, to match.

Table 3. Match Function Comparison

Dataset	NoNetworkMF	ExistenceOnlyMF	FullNetworkMF
akumar	0.770	0.877	0.864
jsmith	0.561	0.773	0.836
ktanaka	0.666	0.918	0.903
bdbcomp	0.900	0.932	0.937
Training Set Average	**0.724**	**0.875**	**0.885**
agupta	0.608	0.699	0.880
cchen	0.523	0.569	0.573
djohnson	0.601	0.719	0.765
jmartin	0.728	0.826	0.872
jrobinson	0.522	0.858	0.808
mbrown	0.614	0.809	0.734
mjones	0.564	0.655	0.738
mmiller	0.656	0.806	0.911
Evaluation Set Average	**0.602**	**0.743**	**0.785**

Finally, the generated *FullNetworkMF*:

(IsAbbrev = 1 and NameSim ≥ 0.94 and MinRQ ≤ 1) or (TitleSim ≥ 0.39 and MaxRQ ≤ 5 and NameSim ≥ 0.87 and NumTitleWords ≥ 2 and IniLastName = 1) or (TitleWordSim ≥ 0.23 or NameSim ≥ 0.87 and IsAbbrev = 1 and VenueSim ≥ 0.35 and MaxRQ ≤ 3) or (VenueSim ≥ 0.67 and MaxRQ ≤ 3 and NameSim ≥ 0.98) or (IsAbbrev = 1 and NameSim ≥ 0.45 and NumTitleWords ≥ 4 and MinRQ ≤ 2 and IniLastName = 1 and RE = 0) or (IniLastName = 1 and RE = 1 and MaxRQ ≤ 9) or (RE = 1 and NameSim ≥ 0.72) or (RS ≥ 2 and IniLastName = 1) . (**3**)

In *FullNetworkMF* the other ASN evidences were also picked, along with *RE*. In this function, with *RS* greater or equal to 2 and the same initial letter and surname, there is a match. This means that with a high *RS*, less evidence is needed to match. *MaxRQ* and *MinRQ* are also used throughout the function. For example, there is a match when the names are similar, the authors are related and *MaxRQ* is no greater than 9. This means that, as the *RQ* increases, *RE* loses its value. This happens as an author with many relationships has a higher chance of being related to a different person with a similar name, a false positive, than an author with few relationships.

In Table 3, we compare quality results in the training datasets and in the evaluation datasets using the K measure. As our results show, when adding the *RE* evidence to the MF, we have a very significative increase in quality. When comparing *NoNetworkMF* to *ExistenceOnlyMF* we have a high increase in K value for all datasets. In the training set we have an average increase of more than 0.15, while in the evaluation set we have an average increase of more than 0.14.

Table 4. Comparison between the HHC method and FullNetworkMF

Dataset	HHC	FullNetworkMF	Difference
agupta	0.777	0.880	0.103
cchen	0.588	0.573	-0.015
djohnson	0.748	0.765	0.017
jmartin	0.885	0.872	-0.013
jrobinson	0.760	0.808	0.048
mbrown	0.855	0.734	-0.121
mjones	0.742	0.738	-0.004
mmiller	0.911	0.911	0.000
Evaluation Set Average	**0.783**	**0.785**	**0.002**

The difference from *ExistenceOnlyMF* to *FullNetworkMF* is not as great, and in some datasets *ExistenceOnlyMF* had a better performance, but in average we can see that using all Social Network evidences brings a significative improvement. In the evaluation set, we have an average improvement of more than 0.04.

To show our approach can compete with state-of-the-art methods, we compared *FullNetworkMF* to HHC method [5] using the same datasets. As we can see on Table 4, both methods had very similar results. Only in three datasets there was a difference of more than 0.02: in ‘agupta’ and ‘jrobinson’ *FullNetworkMF* won by 0.103 and 0.048 while in ‘mbrown’ HHC won by 0.121. In average, *FullNetworkMF* won by 0.002, which is considered as a tie between both methods.

5 Related Work

There has been some work using co-authorship networks in *name disambiguation*. In [2], authors are compared collectively and co-authorship relations are used as evidence that author names represent the same person. However, author names need to have a similar set of co-authors to be considered the same person. Sets of co-authors are also compared on [8], which uses searches on the web to obtain these sets. In [14], a network similarity is calculated as the probability from author *a* to reach author *b* and this similarity is used as evidence to match author names. And in [18], a context graph is constructed for each entity, using co-authorship relations for example, and similarity between graphs is measured. The main difference between these approaches and ours is that in our approach, instead of calculating the similarity of relationship networks, author names need only to be linked and the strength or even the existence of this link will define a threshold for the attribute similarity.

A generic approach has been presented in [9], using the entity-relationship graph on data disambiguation, which can be applied to author disambiguation. However, it only uses relationships on entities that haven't been matched using attributes, while in our approach we use this information when comparing all entities.

Other pieces of research use co-author information, but do not make use of social network analysis. In [5], along with the author name, evidences such as paper title, paper venue and co-author list are used to disambiguate authors. The methods presented in [7] and [20], are based on Machine Learning techniques and [19] uses information extracted from the web as evidence to match author names.

6 Conclusions

In this article, we have used Genetic Programming to evaluate the impact of using social network analysis to solve the author name disambiguation problem in Digital Libraries. We presented a machine learning approach to generate author match functions based on GP. We have also presented Match Functions generated by our approach using real-world datasets. Experimental results have shown that MFs that use social network evidences produce better results than MFs that don't make use of these evidences. By comparing our results to results obtained by the HHC method [5], we have shown that our method can compete with state-of-the-art approaches.

As future work, scalability and generalization issues could be explored. Also, our GP approach could be used to evaluate the impact of adding other evidences which are harder to extract, such as name origin (e.g., Indian, Chinese) or publication topic.

7 Acknowledgments

This work has been partially supported by the Brazilian National Institute of Science and Technology for the Web (CNPq grant no. 573871/2008-6) and by CNPq project no. 550891/2007-2.

References

1. Benjelloun, O., Garcia-Molina, H., Kawai, H., Larson, T.E., Menestrina, D., Su, Q., Thavisonboon, S., Widom, J.: Generic Entity Resolution in the SERF Project. IEEE Data Engeneering Bulletin, Vol. 29, p. 13-20. (2006)
2. Bhattacharya, I., Getoor, L.: Collective entity resolution in relational data. ACM Transactions on Knowledge Discovery from Data, Vol. 1, Issue 1, Article No. 5. (2007)
3. Borgman, C.L: What are Digital Libraries? Competing Visions. Information Processing and Management: an International Journal, Vol. 35, Issue 3, p. 227-243. (1999)
4. Carvalho, M.G., Laender, A.H.F., Gonçalves, M.A., Silva, A.S.: Replica Identification Using Genetic Programming. In: ACM SAC 2008, p. 1801-1806. Fortaleza, Brazil (2008)
5. Cota, R., Gonçalves, M.A., Laender, A.H.F.: A Heuristic-based Hierarchical Clustering Method for Author Name Disambiguation in Digital Libraries. In: 12th SBBD, p. 20-34. João Pessoa, Brazil (2007)
6. Elmagarmid, A.K., Ipeirotis, P.G., Verykios, V.S.: Duplicate Record Detection: A Survey. IEEE Transactions on Knowledge and Data Engineering, Vol. 19, p. 1-16. (2007)
7. Huang, J., Ertekin, S., Giles, C.L.: Efficient name disambiguation for large-scale databases. In: Proceedings of the 10th ECML PKDD, p.536-544. (2007)
8. Kang, I.-S., Na, S.-H., Lee, S., Jung, H., Kim, P., Sung, W.-K., Lee, J.-H.: On co-authorship for author disambiguation. Information Proc. and Management, Vol. 45, p. 84-97. (2009)
9. Kalashnikov, D., Mehrotra, S.: Domain-Independent Data Cleaning via Analysis of Entity-Resolution Graph. ACM TODS, Vol. 31, No. 2, p. 716-767. (2006)
10. Koza, J.R.: Genetic Programming: On the programming of computers by means of natural selection. MIT Press. (1992)
11. Lapidot, I.: Self-Organizing-Maps with BIC for Speaker Clustering. IDIAP Research Report 02-60, IDIAP Research Institute. Martigny, Switzerland (2002)
12. Lee, D., On, B.-W., Kang, J.: Effective and scalable solution for mixed and split citation problems. In: Proceedings of the 2nd IQIS, p. 69-76. Baltimore, Mariland. (2005)
13. Levin, F.H., Heuser, C.A.: Evaluating the Use of Social Networks in Author Name Disambiguation in Digital Libraries. In: 14th SBBD, p. 46-60. Fortaleza, Brazil (2009)
14. Malin, B.: Unsupervised name disambiguation via social network similarity. In: Proceedings of the Workshop on Link Analysis, Counterterrorism and Security, in conjunction with the SIAM International Conference on Data Mining, p. 93-102. Newport Beach, CA (2005)
15. Menezes, G.V., Ziviani, N., Laender, A.H.F., Almeida, V.: A Geographical Analysis of Knowledge Production in Computer Science. In: Proceedings of the 18th international conference on the World Wide Web, p. 1041-1050. Madrid, Spain (2009)
16. Newman, M.E.: The structure and function of complex networks. SIAM Review, 45(2):167-256 (2003)
17. Oliveira, J.W., Laender, A.H.F., Gonçalves, M.A.: Remoção de Ambigüidades na Identificação de Autoria de Objetos Bibliográficos. In: 10th SBBD, p. 205-219. Uberlândia, Brazil (2007)
18. On, B.-W., Elmacioglu, E., Lee, D., Kang, J., Pei, J.: An effective approach to entity resolution problem using quasi-clique and its application to digital libraries. In: Proceedings of the 6th ACM/IEEE-CS JCDL, p. 51-52. Chapel Hill, NC (2006)
19. Pereira, D.A., Ribeiro-Neto, B., Ziviani, N., Laender, A.H.F., Gonçalves, M.A., Ferreira, A.A.: Using web information for author name disambiguation. In: Proceedings of the 9th ACM/IEEE-CS JCDL, p. 49-58. Austin, TX (2009)
20. Treeratpituk, P., Giles, C.L.: Disambiguating authors in academic publications using random forests. In: Proceedings of the 9th ACM/IEEE-CS JCDL, p. 39-48. Austin, TX (2009)