Large Scale Author Name Disambiguation in Digital Libraries

Madian Khabsa
Computer Science and Engineering
The Pennsylvania State University
University Park, PA 16802, USA
madian@psu.edu

Pucktada Treeratpituk
Science Park Promotion Agency
Ministry of Science and Technology
Bangkok, Thailand
pucktada@gmail.com

C. Lee Giles
Information Sciences and Technology
Computer Science and Engineering
The Pennsylvania State University
University Park, PA 16802, USA
giles@ist.psu.edu

***Abstract*—Person name disambiguation is essential to distinguish between persons that share the same name where unique identifiers are not present. In many domains this is a common problem including digital libraries where the same name can refer to multiple unique authors. Correctly attributing work and citations requires the digital library's database to be disambiguated. In this work we describe a large scale framework for disambiguating author names efficiently and effectively. The framework uses a density based clustering algorithm with a random forest based distance function to clusters unique authors. Effective use of blocking functions allows the clustering algorithm to be run in parallel. In our experiments we show that the framework disambiguates authors of more than 4 million papers in 24 hours.**

***Keywords*-Disambiguation, Clustering**

I. Introduction

In many applications person name disambiguation is important for the success of the application. In web search, it is estimated that 5-10%[1] of the entire web search traffic is person name queries which are better served when the results are disambiguated. Recently, major search engines like Bing started focusing on person name search by grouping person profiles form multiple social networking sites into one result.

In digital libraries, correctly attributing work to researchers and measuring impact of their work requires accurate disambiguation. Searching for people on the web or within a digital library is challenging when many people share the same name. Based on date from from 1990 U.S. Census Bureau, 90,000 different names are shared by 100 million people [2].

Generally, the ambiguity of person's name comes in three varieties: (1) the aliasing problem: when a person uses multiple name variations such as "Ronald W. Williams" and "R.W. Williams", and (2) the common name problem: when there is more than one person with the same name, which is especially problematic for high frequency names such as many Chinese names; and (3) the typographic errors - which could result from human input or automatic extraction systems which is the case in many digital libraries.

This work describes how to build a framework for efficient and parallel name disambiguation based on DBSCAN clustering algorithm and a random forest [3] learned distances function.

The remainder of this paper is organized as follows. Section II describe the components of the disambiguation framework. Section III presents the approach, and Section IV describes the experiments.

II. Components

Given a digital library with n publications, $P_1..P_n$, each publication P_i is authored by authors $a_{i1}..a_{im}$, the goal is to create a disambiguated digital library such that for each disambiguated author A_i we cluster all the variations of her name together. That is A_i contains all a_{kj} appearing in P_k where a_{kj} refers to A_i. In typical machine learning setting for solving this problem two components are needed to build the clusters $A_1..A_w$. First a clustering algorithm is needed to create each cluster A_i from the components a_{jk}. Secondly, a distance function is needed to measure if two author names, a_{jk} and a_{il}, refer to the same person or not. The distance function is used in the clustering algorithm to assemble name variations of the same person together or to break similar names that refer to different persons from being in the same cluster.

We use DBSCAN as clustering algorithm [1], [4]. DBSCAN is a density based clustering algorithm that does not require the number of clusters to be determined beforehand. Rather it recursively adds points to the cluster that are within a maximum distance from it and greedily keeps expanding each cluster until no points can be found within the allowed distance. DBSCAN requires two parameters, ε, the maximum allowed distance between two data points for them be allowed to be in the same cluster. The second parameter is *minPts*, the minimum number of data points within ε distance from the seed point of the cluster for the cluster to be considered dense. Only dense clusters are retained by DBSCAN as clusters with less than *minPts* are considered noise. Procedure 1 shows the pseudo code of DBSCAN.

The distance function in this case is learned using random forests classifier [5]. Given two author names appearing in two different papers, a profile is created for each author and the classifiers gives a score between 0 and 1 estimating the likelihood that these two authors refer to the same person. Random Forest is an ensemble of decision trees where each

tree classifies the profiles independently. The percentage of trees voting in favor of similarity is used to estimate the similarity distance. It uses variety of features including name-related information (names and emails), affiliation, coauthors (names and their affiliations), venue information (venues and years), content (abstracts and titles), keyphrases and citations.

Procedure 1 DBSCAN(D)

Input: D - static collections of records to be disambiguated

```
1: mark all records in D as UNVISITED
2: for all record p in D do
3:     if p is UNVISITED then
4:         mark p as VISITED
5:         N ← query(D, p, ε)
6:         if |N| < minPts then
7:             assign p → NOISE
8:         else
9:             expandCluster(p, N)
10:        end if
11:    end if
12: end for
```

Procedure 2 expandCluster(p, N)

```
1: cid ← nextClusterId()
2: assign p → cid
3: Q ← N /* put records in region into a queue */
4: while Q ≠ ∅ do
5:     q ← pop a record from Q
6:     if q is UNVISITED then
7:         mark q as VISITED
8:         N' ← query(D, q, ε)
9:         if |N'| ≥ minPts then
10:            Q ← Q + N'
11:        end if
12:    end if
13:    if q doesn't belong to any cluster then
14:        assign q → cid
15:    end if
16: end while
```

III. Approach

Previous work that used density based clustering with machine learning distance functions used to take days to disambiguate authors of less than 1 million papers [1]. In this approach, our contribution demonstrates how to parallelize the clustering algorithm such that the disambiguation could be done in less than a day for much larger databases. Naive methods for clustering would require computing the similarity n^2 times to be able to build the clusters.

The basic idea is to split authors into blocks where each block can be disambiguated independently from the rest of the blocks. Therefore, a given block should only contain authors that may belong to the same cluster, while at the same time no two author names that belong to the same cluster should end up in different blocks. By noting that different last name is enough to conclude that two names do not refer to the same person, we use a blocking function that split authors into blocks containing people with same last name and first initial.

In the preprocessing step, folders are created for each pair of initials within the database. Within each folder, a file is created containing all author Ids that share the same last name and first initial. Later, DBSCAN is run independently on each block in parallel using GNU Parallels.

IV. Experiments

The framework is used to disambiguate the database of CiteSeerX. It contains more than 4 million papers and 12 million author names affiliated with these papers. The machine is a virtual machine configured with RHEL 6, contains 12 cores and 32 GB of memory. All the data is hosted in remote MySQL database from which the algorithm will be fetching author information. When two author points are being compared, the features profile is created on the fly.

GNU Parallel [6] is configured to run as many processes as the number of the cores. It spawns a new process for each blocking file that contains authors within the same block. Processes are only spawned when another process terminates so that the CPU utilization is always 100% on all the cores. The number of active process at the same time never exceeds the predefined limit of the number of cores.

Building the clusters within each block in parallel allows for the entire database of CiteSeerX which contains more than 4 million papers to be disambiguated in less than 24 hours.

Acknowledgments

We gratefully acknowledge partial support from the National Science Foundation.

References

[1] J. Huang, S. Ertekin, and C. L. Giles, "Efficient Name Disambiguation for Large-Scale Databases," in *The 10th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD 2006)*, 2006, pp. 536–544.

[2] J. Artiles, J. Gonzalo, and F. Verdejo, "A Testbed for People Searching Strategies in the WWW," in *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. New York, New York, USA: ACM Press, 2005, p. 569.

[3] L. Breiman, "Random Forests," *Machine Learning*, 2001.

[4] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *Kdd*, vol. 96, 1996, pp. 226–231.

[5] P. Treeratpituk and C. L. Giles, "Disambiguating Authors in Academic Publications using Random Forests," *In Proceedings of the Joint Conference on Digital Libraries (JCDL'09)*, Jan. 2009.

[6] O. Tange, "Gnu parallel-the command-line power tool," *The USENIX Magazine*, vol. 36, no. 1, pp. 42–47, 2011.