

Master DMKM Report



Bibliographic Entity Automatic Recognition and Disambiguation

Hussein AL-NATSHEH
(Author ID: orcid.org/0000-0003-0665-5959)

Defended on July 8, 2015

Supervision : Dr. Louppe Gilles (CERN)
Prof. Lorenza Saitta (University of Eastern Piedmont)

Location : CERN (Geneva, Switzerland)



Abstract:

This master thesis reports an applied machine learning research internship done at digital library of the European Organization for Nuclear Research (CERN).

The way an author's name may vary in its representation across scientific publications creates ambiguity when it comes to uniquely identifying an author; In the database of any scientific digital library, the same full name variation can be used by more than one author. This may occur even between authors from the same research affiliation. In this work, we built a machine learning based author name disambiguation solution. The approach consists in learning a distance function from a ground-truth data, blocking publications of broadly similar author names, and clustering the publications using a semi-supervised strategy within each of the blocks.

The main contributions of this work are twofold; first, improving the distance model by taking into account the (estimated) ethnicity of the author's full name. Indeed, names from different ethnicities, for example Asian versus Arabic names, should be processed differently. This added feature led to a better clustering evaluation. It also got a high contribution percentage in the feature importances analysis. The second main contribution was to decide on a thresholding strategy to form a flat clustering from the agglomerative hierarchical clustering. Six different strategies were evaluated to estimate the number of clusters in each block. The strategy that provides the best evaluation results was using a blocking function that groups signatures with common last name and first name initial, then applying the semi-supervised clustering on the blocks that contains samples from the ground truth. The blocks that do not have any labeled sample will form a single cluster. A smaller contribution also made to the distance model including feature engineering and pairs sampling. Overall, the model accuracy is 98% compared to 94% if we only disambiguate on the common normalized last name and first name initial. My work contributed to raise the accuracy from 97% to slightly more than 98%. This is equivalent to reduce the error rate by about 35%. During the project, I have also contributed to an open source project which will eventually be deployed in the high-energy physics digital library of CERN (<http://inspirehep.net>).

There were many factors that led to achieve such an accurate disambiguation model. A key factor was having a ground-truth data which allowed us to design a very good semi-supervised clustering. Another factor was learning an accurate distance model with an appropriate feature engineering in which we manage to incorporate an external knowledge of the name ethnicity.

Résumé :

Les résultats du stage de recherche effectué au sein de la bibliothèque numérique de l'Organisation européenne pour la recherche nucléaire (CERN) concernant les applications de l'apprentissage automatique ont été présentés dans ce mémoire de Master.

La formulation des noms d'auteurs diffèrent d'une publication scientifique à une autre, ce qui crée une ambiguïté lorsqu'il s'agit d'identifier un auteur de manière exclusive. Dans la base de donnée de n'importe quelle bibliothèque numérique scientifique, la même variation du nom peut être utilisée par plus d'un auteur. Ce cas peut également se poser entre plusieurs auteurs d'un même domaine de recherche.

Dans le cadre de ces travaux, nous avons élaboré une solution basée sur l'apprentissage automatique (Machine Learning) pour effectuer la désambiguïsation des noms d'auteurs. Notre approche consiste à apprendre la fonction de distance à partir de données réelles, en rassemblant par bloc les publications dont les noms d'auteurs sont semblables, et regrouper les publications en utilisant une stratégie de clustering semi-supervisée parmi chaque bloc. Afin de résoudre ce problème, notre contribution exposée dans ce rapport a été double : D'une part, en améliorant la fonction de distance en tenant compte de l'appartenance ethnique (estimée) du nom d'auteur. En effet, les noms provenant de différentes ethnicités, par exemple les noms asiatiques et les noms arabes, doivent être traités différemment. Cette fonctionnalité supplémentaire permet une meilleure évaluation de la méthode de regroupement. D'autre part, en proposant une stratégie de "seuillage des valeurs" (thresholding strategy) pour constituer un partitionnement à plat (flat clustering) à partir d'un groupement agglomératif hiérarchique (agglomerative hierarchical clustering). Six stratégies différentes ont été considérées afin d'évaluer le nombre de clusters dans chaque bloc. Les méthodes d'extraction de caractéristiques (feature engineering) et un échantillonnage intelligent de paires (pairs sampling) ont également contribué au modèle de distance. Dans l'ensemble, la précision de ce modèle innovant est de 98%, contre 94% si nous désambiguïsons uniquement les initiales communes des noms et prénoms. Mes travaux de recherche ont contribué à augmenter la précision de 97% à un peu plus de 98%. Ceci équivaut à une réduction du taux d'erreur d'approximativement 35%. De plus, durant le projet, j'ai également participé à la réalisation d'un projet open-source qui sera finalement utilisé dans la bibliothèque numérique du CERN (<http://inspirehep.net>).

Plusieurs facteurs ont permis d'atteindre un modèle de désambiguïsation précis. L'un d'entre eux était d'avoir eu accès à des données réelles, nous permettant ainsi de concevoir un très bon regroupement semi-supervisé. L'apprentissage d'un modèle de distance très précis ainsi qu'une extraction de caractéristiques appropriée dans lequel nous avons réussi à incorporer de nouvelles connaissances externes sur l'ethnicité du nom représentent le deuxième facteur de réussite.

Contents

1	Hosting Institution	1
2	Acknowledgements	1
3	Background and Motivation	2
3.1	Introduction	2
3.2	Related Works	3
3.3	Summary of Contribution	4
4	Methodology	5
4.1	Overview	5
4.2	Blocking	7
4.3	Distance Model	8
4.3.1	Features Engineering	8
4.3.2	Estimating the Ethnicity from the Full Name	10
4.3.3	The Design Parameter of the Classifier	11
4.4	Clustering	11
4.4.1	Hierarchical Clustering	11
4.4.2	Estimating the Number of Clusters	11
5	Results and Evaluation	14
5.1	Pairs Sampling	14
5.2	Accuracy Measures for Clustering	14
5.2.1	Pairwise F-Measure	15
5.2.2	BCubed-F-Score	15
5.3	Distance Model	15
5.3.1	Feature Selection	15
5.3.2	Estimating the Ethnicity from the Full Name	17
5.4	Clustering Evaluation	17
5.4.1	Parameters Tuning	17
5.4.2	Number of Clusters Strategies	17
5.4.3	Evaluation of Pairs Sampling Strategies	20
6	Implementation	24
6.1	BEARD: Open Source Library for Automatic Entity Disambiguation	24
6.2	Python Scikit-Learn and API compatibility	24
7	Conclusion and Future Works	25

1 Hosting Institution

Founded in 1954, the CERN laboratory sits astride the Franco-Swiss border near Geneva. It was one of Europe's first joint ventures and now has 21 member states.

At CERN, the European Organization for Nuclear Research, physicists and engineers are probing the fundamental structure of the universe. They use the world's largest and most complex scientific instruments to study the basic constituents of matter, the fundamental particles. The particles are made to collide together at close to the speed of light. The process gives the physicists clues about how the particles interact, and provides insights into the fundamental laws of nature. The instruments used at CERN are purpose-built particle accelerators and detectors. Accelerators boost beams of particles to high energies before the beams are made to collide with each other or with stationary targets. Detectors observe and record the results of these collisions. The name CERN is derived from the acronym for its French origin name, a provisional body founded in 1952 with the mandate of establishing a world-class fundamental physics research organization in Europe. At that time, pure physics research concentrated on understanding the inside of the atom, hence the word "nuclear". Today, the understanding of matter goes much deeper than the nucleus, and CERN's main area of research is particle physics, the study of the fundamental constituents of matter and the forces acting between them. Because of this, the laboratory operated by CERN is often referred to as the European Laboratory for Particle Physics. CERN is a large organization. The hosted department was the Scientific Information Systems (SIS). One of the main duties of SIS is managing and maintaining the high-energy physics digital library INSPIREHEP.NET. I worked with the software development team who is specialized in applying machine learning solutions to the digital library.

2 Acknowledgements

My ultimate thanks goes to God who surrounded me with a lot of people who helped me through this master thesis. Special thanks to my parents and my beloved wife for supporting me in this studies. I want to thank the European Commission for funding my studies through this Erasmus Mundus master program. Thanks to all the professors who participated in teaching courses of the master program. I want to thank the professors who wrote recommendation letters for me supporting my application to this master. Namely, Prof. Omar Al-jarrah, Prof. Mazuz Salahat and Prof. Lo'ai Tawalbeh. Thanks to the hosting organization, CERN, specially my supervisor Dr. Gilles Louppe who guided me all over the master thesis work.

3 Background and Motivation

3.1 Introduction

Name disambiguation is a special cases of a bigger problem called Entity Resolution. Entity Resolution is the task of disambiguating manifestations of real world entities in various records or mentions by linking and grouping [12]. For example, there could be many text variations of a person name. In digital libraries, the most problematic entity to disambiguate is the author. Here comes the problem we are trying to solve in this project which is the author name disambiguation.

(Usai G.) an example for 2 different authors who share the same exact name in the database. One of the authors is from Chigago university and the other is affiliated to Istituto Nazionale di Fisica Nucleare, Cagliari. This is what is referred to as the homonym case. The other case is the synonym, which is the case of different names of the same real author. for example, (Ozcan Erkan) and (Ozcan Veysi E.) which is 2 name variations of the same real author. So, the problem could be defined as 2 main sub-problems: One, is not to merge 2 different authors into 1 profile, while the other, is not to split the same author into 2 profiles.

Incorrect splitting or merging of author profiles based on the author's name is harmful when it comes to list the publications of a single author; The results sometimes contain publications from different real authors who have common author name variation. In other cases, the list of publications is not complete because of missing some name text variations. That leads to some problems for example the ranking (H-Index) which often used in appraisals or hiring of the researcher.

Of course, there is some simple techniques that digital libraries adopt to mitigate the author disambiguation problem. We will categorize them into 3 categories: First, Heuristics approaches; for example, some might only use the author name variation as the identifier of the author. This could be considered as the worst method to use. Others, transform full names into the surname and given names initials. Then, it is assumed that two identical records are matches. A bit further process, but still simple, is to look into the affiliation of the author assuming that the researcher would not change the affiliation much. Obviously, it is not the case in many disciplines as researchers usually have many affiliations during their career.

The second category is manual curation; for example, some digital libraries web interfaces allow a subscribed author to claim the correct list of his publication. These claims then approved by the manual curation admin/editor who looks at them case by case. A global initiative called ORCID (<http://orcid.org>) is an open, non-profit, community-driven effort to create and maintain a registry of unique researcher identifiers and a transparent method of linking research activities and outputs to these identifiers . However, this is still considered new and a lot of authors, and sometime publishers, are either not aware of it or are not motivated to use it for old or the new publications. Currently some digital libraries including the one of CERN allow the users to claim publications by signing up with their ORCID account so they can clean up their retrieved publication list. However, in many cases, authors actually are not motivated to provide such an input. Another drawback of this is the risk of getting a fault input which brings more efforts in manual curation.

The third category of solutions is using advanced machine learning algorithms to automate the process. In some digital libraries, machine learning is used to help human curation. This is an active field of research in which our work follows to solve the problem. Microsoft academics (academic.research.microsoft.com) is a relatively recent example of a digital library that applies machine learning based techniques for author disambiguation. In 2013, they announced a challenge KDD 2013 Cup on Kaggle.com [21]. Since then, author disambiguation problem got more interest by the machine learning researchers community. CiteSeer, Google Scholar, among others, also utilize machine learning algorithms.

The database we have experimented the solution with was inspirehep.net. It is the CERN digital library of high energy physics (HEP) publications. HEP papers are kind of special special in term of the relatively very big numbers of coauthors per publication. There is the concept of experiment collaboration which could have hundreds of researchers all getting the credit on any publication published by the collaboration. Machine learning approaches of author name disambiguation that base the solution on publication similarity might have a problem with having too many co-authors on the same publication.

A counter example we have in the database is having 2 co-authors with the exact full name on the same publication. In such case, the 2 authors have exactly the same publication, so, any similarity measure will group them together. This is more common with Asian author names.

Asian author names cause one of the biggest challenges that most digital libraries. The case is similar with other origins/languages like Indian or Arabic. The root cause of that is converting the name from its original languages into English or in general into a Latin letters language. While translating the publication, the written strings usually based on the sound that the origin name could be mapped to. In some cases, there are many possible string of Latin characters to represent a foreign sound that is not exist in the Latin language. Still, Asian names, specially Chinese names, are the most challenging ones having short names with very similar sounds in the perspective of Latin languages. The number of Chinese authors is also relatively high comparing with other origins which correlated to the high population of China.

3.2 Related Works

There is a large body of work revolving around author disambiguation. Unfortunately, the significant variation from article to article in the datasets used makes direct comparison of various techniques very difficult. Recently, the authors of [9] surveyed many of the existing techniques and algorithms in the domain, categorizing them into two fundamental categories: author grouping methods and author assignment methods. Author assignment methods attempt to perform disambiguation by learning a model for a real-life author based on available information (article title, journal name, etc.). These techniques tend to be limited in that they either require a large training set for supervised learning (which is often not feasible) or, for clustering methods, an estimated number of real-life authors for the dataset (which is often not available) [4]. They are, however, quite effective when this information is available. One method [22] estimates this parameter with relative success.

Most methods, including ours, fall into the (author grouping) category. These algorithms generally use a similarity metric for comparing articles based on available meta data. Distance based partitioning clustering generates group of publications that ideally correspond to distinct real-life authors. The similarity metrics used depend on the metadata available for an article, which generally consists of a list of author names, the article title, and the publication venue in the minimal case, and sometimes contains other information like the affiliation or keywords. Common metrics often make use of string comparison methods for the various fields, like Jaccard similarity [15] or cosine similarity on vectorized data representation like TFIDF, which is described in section 4.3.1, while others try to learn a similarity metric from the data with supervised techniques [8, 32].

Methods also differ in the way that they treat the importance of each field when computing similarity. In [22], for example, the authors use a multi-step approach, clustering first based on author names and then using other information later to merge clusters. Another interesting method follows a multiple layered clustering approach [16]. The unsettled dataset is first passed through an email clustering layer. The second layer based on co-authorship then processes the undetermined citations from the first layer. Finally, the third layer based on title text similarity uses fuzzy clustering for splitting and merging clusters. weighted averages combination of these 3 ordered layers resulted in very good precision and recall clustering quality measures on their manually built test set. The good thing about this approach is that one may add more different layers as needed. However, some layers like the email, is not always available and difficult to obtain.

In terms of clustering methods, many various algorithms are used, with most falling into one of the following categories: [9] partitioning, hierarchical agglomerative clustering, density-based clustering, and spectral clustering. The most commonly-used algorithms involve some kind of agglomerative clustering. In [5], authors use the terminology Record linkage as a synonym of entity resolution. They propose a three step method for clustering. In the first step, the algorithm performs blocking to separate the dataset into buckets of articles that most likely aren't interrelated. Next, blocking is performed to initialize the clusters such that each contains articles that are nearly certain to correspond to the same author based on a similarity measure. Following this, clusters are merged iteratively based on an inter-cluster similarity measure, stopping once no two clusters have a similarity higher than some threshold. However, the

authors do not specify how to determine this threshold, which is an important aspect of the algorithm given that it provides the stopping criteria.

A recent article [13] on incremental record linkage describes a general pipeline for entity resolution in terms of three steps: blocking, which consists of dividing a dataset into groups of articles that almost certainly correspond to different entities ; similarity computation, which computes the similarity between all pairs of articles in a block; and graph clustering based on a similarity graph, where nodes are articles and edges between pairs of articles are weighted by their similarity. (Edges only exist if the similarity is above a predefined, manually set threshold). Their algorithm makes use of a technique known as correlation clustering, which attempts to minimize a penalty metric based on the sum of the intra-cluster distances and the inter-cluster similarities. They propose an algorithm that allows the addition of new records to a database iteratively without having to re-disambiguate the entire dataset from scratch. The algorithm does not fully minimize the correlation penalty, as this problem is shown to be NP-Complete , but rather attempts to find a high quality clustering according to this metric in a shorter time-frame.

The approach of simple, initials-based methods can tackle the disambiguation of author name [25]. It seems to be simple but its challenge relies on using first initial method or all initials method. By experiment, based on the simulations in five diverse disciplines, just first initial method shows some accuracy of 97% correctly identifies. And taking all initials into account is typically two times accurate. Using the combination of the two methods reduces the fraction of incorrectly identified authors by 10-30% over the first initial method.

There are some clustering algorithms that don't need to priorly specify the number of clusters like DBSCAN, as a density-based clustering method [9]. However, DBSCAN requires some design parameters setting as well. This parameters setting can lead to different number of clusters. So, the problem of setting the number of clusters is still there but it is more dependant on the data nature than just setting the number in advanced. DBSCAN has been used in author disambiguation in many related works like [18].

Our approach capitalized on the idea of working on labeled pairs of publications which was utilized in [20]. They used a collection of high precision features to bootstrap a training set with positive and negative examples of co-referring authors. The bootstrap clusters are used as a ground truth data in their disambiguation pipeline. Indeed, our approach has a similar pipeline however, we did not use bootstrapping as we already had ground-truth data. We also use very similar clustering evaluation metrics of pairwise F-score which will be presented in section 5.2.

Rather than just using some distance measures like cosine or Jaccard similarity, some related works, like in [20] and [18], use a custom distance function that is learned on labeled data. The labeled training data here is a list of pairs signatures. As will be described in section 4.1, signature is a unique combination between a publication and an author name variation mention on that publication. The label then tell weather the 2 signatures are for the same real author or not. Authors of [18] built the distance function by learning a random forests classifier that gives a score between 0 and 1 estimating the likelihood that the 2 signatures are for the same person.

Similarly to DBSCAN, Evolving Clustering Method (ECM) [29], also it does not need to determine the number of clusters but we did not find a related work used it for author name disambiguation. One extra algorithm that does not need to explicitly determine the number of clusters in advanced in Adaptive Resonance Theory Neural Networks (ART) [7]. Yet, it needs to set the upper limit of the number of clusters among other design parameters. ARTgep [2].] that optimize the selection of ART design parameter.

There are a few non-parametric method to compare different setting of the number of clusters [11] like stability, gap statistic, distortion jump, Calinsky and Harabasz. Silhouette score [27] is an example of these category of methods that we tried in our experiments. We used it as an unsupervised method to compare with our semi-supervised approach which will be described in section 4.4.2.

3.3 Summary of Contribution

The main work contribution during the master thesis internship at the hosting organization, CERN, could be summarized as follows:

- Adding and comparing extra similarity measures namely scalable Jaccard similarity and co-authors graph-based similarity measure.
- Implementing and evaluating unsupervised clustering scores for estimating the best number of clusters namely Silhouette and Gap statistics.
- Feature engineering in which the main contribution was adding a new feature which is estimating the ethnicity of the name. This new feature provides better clustering results with better handling of Asian names. My contribution to the feature engineering also includes evaluating and selecting the best subset features that provides high accuracy with feasible computational time.
- Evaluating and deciding on the best thresholding strategy for semi-supervised clustering. This is a very challenging task which determine the number of clusters. Best Threshold estimation using (weighted median, weighted average, and greedy search), clustering unsupervised score, and combinations of these strategies with semi-supervised clustering. This includes what to do if we do not have any ground truth training pair in the block.
- Evaluating and deciding on the better blocking strategy comparing last name first initial blocking versus double-metaphone last name blocking with threshold strategy.
- Implementing and evaluating two possible methods for pairs sampling given a blocking strategy. The methods were balanced sampling in which we oversample difficult rare cases versus uniformly random sampling.
- Comparing different hierarchical clustering methods: single, complete, average and weighted.

4 Methodology

4.1 Overview

Before presenting the methodology, we need to define some terminology we will use to describe the data entities. An author’s full name, or author name, is the string variation that was mention in the database record to refer to the author. The full name is a composition of the author last name and the author other names. For example, in the full name (Kerth, Leroy T.), the last name is Kerth and the other names will be (Leroy T.). We also use the terminology initials. In our full name example, the initials will be (L T). First initial then will be (L). So, when we same for example that we group by last name first initial, it means in our example, (Kerth L). What we refer to as the real author is the real person behind the publication. We will use the term signature a lot as the entity we apply clustering on. signature is a unique combination between a publication and an author name mention on that publication. The term blocking means grouping the signature by a certain version of the author name. For example, blocking by the author last name means grouping together all signatures that have in common the same author last name.

Our methodology consists in 3 main processes; signatures blocking, distance model learning, and clustering. The general diagram of the pipeline is illustrated in Figure 1. In order to reduce the computational complexity, we apply a blocking function. Each block contains a set of signatures that might belong to the same real author. For example, signatures that share the same last name. The other main process is to learn the distance model on a training set sampled from these blocks. A dissimilarity matrix is then computed for each block using that distance model. A hierarchical clustering algorithm is then applied to dissimilarity matrix of each block. After a flat clustering is extracted from each block clustering using a cutting threshold strategy, all clusters generated from all blocked are finally combined into one clustering result.

The result is all the signatures labeled with a predicted cluster ID. The output file of the pipeline is a dictionary of cluster IDs as keys and a set of signatures as values. The signatures inside each cluster represent all author name variation with their associated publications. The 3 main processes of the algorithm are described in sections 4.2, 4.3, and 4.4.

Figure 1: The General Pipeline of the Disambiguation Algorithm

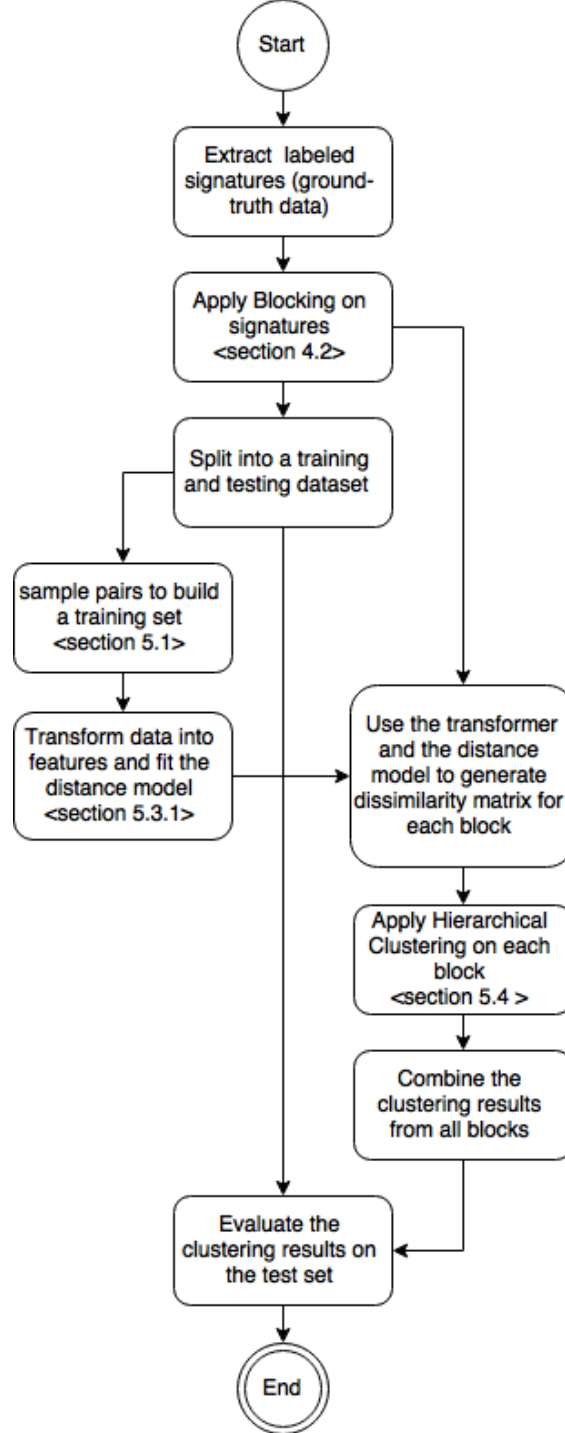
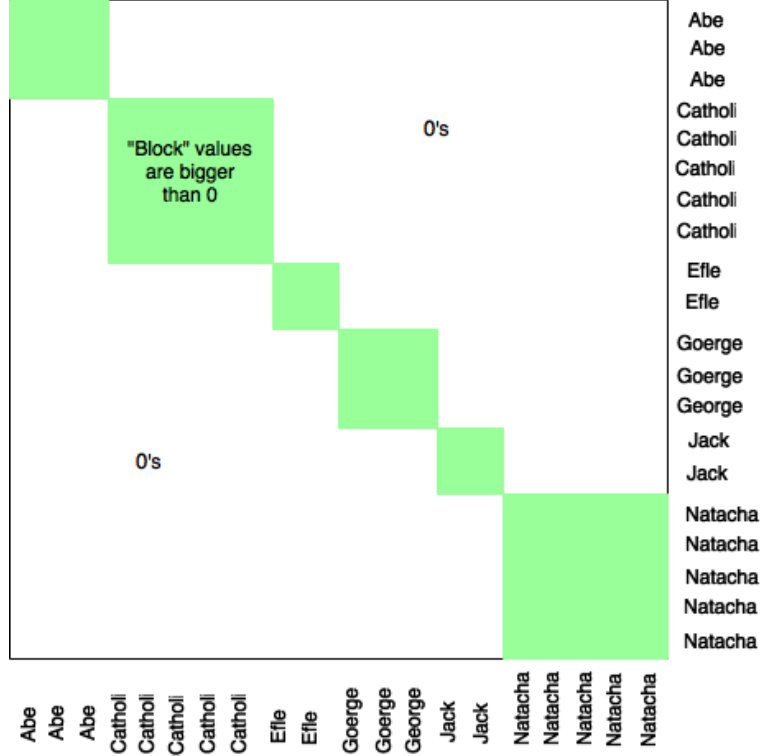


Figure 2: Demo illustration of a similarity matrix of 20 signatures labeled with their last names showing that most of the matrix values are zeros



4.2 Blocking

As adopted in many related works, clustering is done on blocks rather on the whole data at once [18]. This is not only more scalable but also proven to lean into better clustering results. The orders of magnitude in computing the distance affinity matrix is $O(N^2)$, where N is the number of signatures. However, as in Figure 2 of a demo similarity matrix of last names, most of the values are zeros. Blocking only similar entities will almost linearize the problem. So, only similar entities with potential ambiguity problem would be blocked together to be disambiguated. Saying though, blocking could be a tricky phase when it comes to the clustering results. A very simple blocking strategy could be grouping together author names that share the last name. The problem with this approach is that the size of a single block could be very big for computation. It could also group together some entities that have less probable to be of the same authors. For example, 2 author names with different first name initials. The other extreme is to block each unique name together which obviously would lead to separate signatures that belongs to the same author just because the name was shortened for example.

In our dataset, what we found statistically a good compromise between the blocking on the last name or on the full name is to block on the last name and first initial. In the solution implementation, there is another blocking strategy but more complex. The complex blocking strategy utilizes a phonetic string matching algorithm called Double-Metaphone [26]. Transforming the name into its phonetic representation helps to group names with very similar pronunciation. For example, Loay, Louai, and Loai which is actually the same Arabic name but written differently in English. In the Double-Metaphone strategy, all the authors names first converted to their Double-Metaphone version. Then, they are grouped by the Double-Metaphone version of the last name into blocks. Another complexity of this strategy is requiring to set the maximum size of the block. When the block exceeds the maximum size, it starts to group by the first initial of the Double-Metaphone version to sub block the signatures. The actual implementation of this strategy also includes more heuristics to handle some special common cases of names with many tokens. In the experiment section, we will compare the last name first initial blocking strategy versus Double-Metaphone blocking strategy.

A common string normalization function is applied to the author full name before we block on the last

name or last name first initial(s). This normalization is as simple as only keeping the ASCII characters and transform to the lower case. This simple transformation leads to a better blocking being insensitive to the upper/lower case and possible undesired special characters. For example; (Mac‘Cleod) as a last name will be grouped with the last name (Maccleod) by transforming both of them into (maccleod).

4.3 Distance Model

What we refer to as a distance model here is a learned distance function that takes a pair of signatures and computes how probable they are not the same person. The distance model is built to be used in computing the dissimilarity matrix, what we may also refer to as affinity matrix, of signatures of each block. In order to learn distance model we will need to extract and select a set of features out of a training dataset. The quality of this distance model will determine the quality of the whole disambiguation solution since it is the input to the clustering algorithm we will discuss in section 4.4.

Using many features of the pair of signatures, A decidable question will be how to distribute the weight of each feature to the final dissimilarity value. First thought might be to equally distribute the weight. However, some features are more important than others for the author name disambiguation task. To overcome this question, we have implemented a supervised learning of the distance classifier with a representative training dataset. Accordingly, the classifier learn how to use theses features minimizing the prediction error.

4.3.1 Features Engineering

Each signature of each pair has 2 main pieces of information: The author full name and associated publication details. Each publication comes with title, list of co-authors, abstract, references, affiliation, collaboration, set of keywords, journal, year of publication. Out of each pairs which has these fields in each signature, we extract the pairwise vector similarity to construct features from which a distance function can be learned.

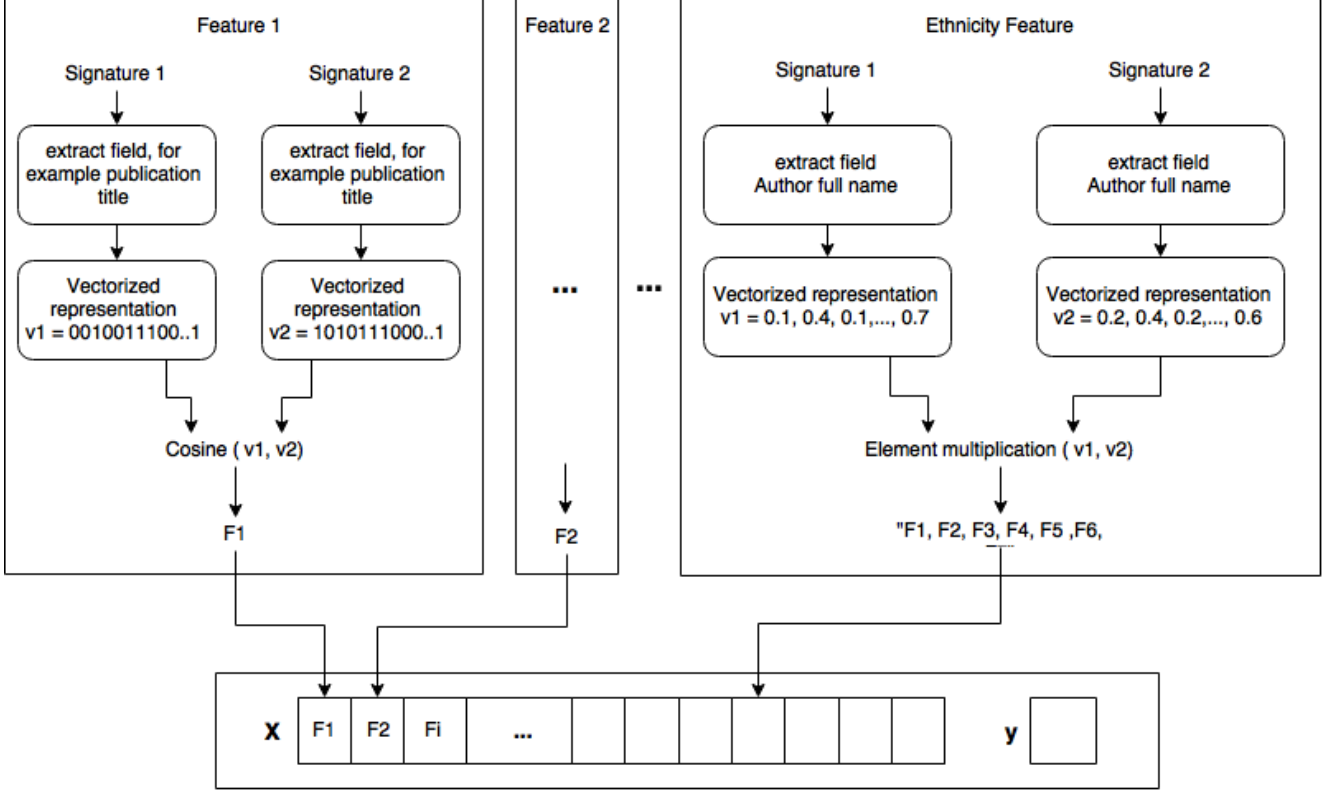
Some data fields, like (the publication set of topics), have a lot of missing values. So we decided not to use them in the first implementation. However, another machine learning project is running in parallel to our project aiming to extract such fields from the PDF file of publications. This will estimate a lot of missing values where we can evaluate including these data fields in the feature engineering process as a future work.

Out of the author full name, we extracted 3 features; full name, other names, and name initials. We applied text preprocessing like removing anything other than ASCII characters and transform all to lowercase. Same text preprocessing is applied to other textual features like the abstract, publication title and keywords. later in this section, we will discuss the vectorization method and the similarity measures we have used in the feature engineering process. The main process is shown in Figure 3.

Vectorization Extracting features from text certainly requires vectorization. As commonly applied in text mining applications, we have applied Term Frequency - Inverse Document Frequency (TF-IDF) vectorization [23]. TF-IDF is a statistical measure of term importance to a document in corpus. Common terms have lower weight than rare ones. Since the author name is a short string document, we extracted bi-gram, tri-gram and 4-gram.

Similarity Measures and Combining the Pairs of Vectors We implemented a few similarity measures like cosine and Jaccard similarity. We need such similarity measure in the feature engineering process when we combine the pair of vectors forming as illustrated in Figure 3. As seen in Equation 1, cosine similarity is a similarity measure between two vectors A and B of an inner product space that measures the cosine of the angle between them. [28]. This applies for any number of dimensions. We also wrote an efficient implementation of Jaccard similarity [15] taking care of both the sparse and the dense case of the input arrays. That is for example when using for co-authors feature, the number of authors is high. The vectorized representation of this feature will be sparsed. The intersection was computed by dot product while the union was computed by counting the nonzeros of the vector summation. Efficiency

Figure 3: Feature engineering process



implementation is critical in this contest as the number of pairs is still high for each block. However, by running few experiments comparing Jaccard and cosine similarity, we found cosine similarity measure performing better.

$$\text{cosineSimilarity}(A, B) = \frac{A \cdot B}{\|A\| \|B\|} = \frac{\sum_{i=1}^n A_i \times B_i}{\sqrt{\sum_{i=1}^n A_i^2} \times \sqrt{\sum_{i=1}^n B_i^2}} \quad (1)$$

For some features, direct text comparison may not always be the most appropriate way to find distances among the papers. Dealing with referenced papers and co-authorships, for which better patterns can be found when building citations graphs. Such graphs allow links to be found in the data when two papers are found to be similar with a third. Accordingly, we have tried out the graph-based similarity measure; a graph-based similarity is a similarity measure that we can extract from a graph representation of entities as node and relationship between nodes as edges.

For instance, when it comes to co-authorship researchers usually work inside a community which has similar topics of research. A researcher can publish a great amount of work with some authors, for which text comparison using TD-IDF would yield good results. It is often the case, however, that an author works with different co-authors who in turn work together. So, By looking at the papers written by coauthors of a given a paper we can consequently find better links in data.

We have examined this idea of computing similarities between signatures by drawing a citation graph. The metric used is the number of shared neighbors divided by the maximum number of neighbors they could have (the size of the biggest neighbors set of the two papers) for normalization. However, when we run some experiments on sub training datasets, we found out that the feature of this graph-based similarity measure does not perform well in the features importance analysis of the distance model estimator. This feature is also computationally costly and quadratic to the size of co-authors set. Accordingly, we decided not to use this similarity measure in our solution. Indeed, the only similarity measure we have utilized in our experiments to combine the pairs of vectors was the cosine similarity. The other combination functions we have used was the absolute difference combiner for the years difference feature and the element multiplication combiner for the ethnicity features which will be introduced in section 4.3.2

Table 1: Summary of the untreated ethnicity data extracted from IPUMS-USA

Code	Label	Count
1	White	20,691,731
2	Black	3,251,482
3	American Indian or Alaska Native	148,821
4	Chinese	52,597
5	Japanese	52,731
6	Other Asian or Pacific Islander	27,555
7	Other race, nec	698

4.3.2 Estimating the Ethnicity from the Full Name

As briefed in the introduction, having Asian names mixed with Latin names in the dataset causes a lot of mis-clustering results. The winners of Microsoft Academic author-paper identification challenge [21] took care of this specialty of Asian names. This inspired us to add a feature about the ethnicity of the name of the given pair of signatures. The first task for bringing this feature was to have a dataset in which the full name is labeled with ethnicity group. It was a very challenging task as such data is considered sensitive and confidential. Fortunately, we managed to have an access to the Integrated Public Use Microdata Series (IPUMS). IPUMS is part of Minnesota Population Center (MPC) which is one of the world leading developers of demographic data resources. We had to fill a special form and agree to only use it for the master thesis purposes in order to have access to two of its data repositories namely: Harmonized data on people in the U.S. census and American Community Survey, from 1850 to the present (IPUMS-USA), which we have to cite exactly as [Steven Ruggles, J. Trent Alexander, Katie Genadek, Ronald Goeken, Matthew B. Schroeder, and Matthew Sobek. Integrated Public Use Microdata Series: Version 5.0 [Machine-readable database]. Minneapolis, MN: Minnesota Population Center [producer and distributor], 2010.], and Complete-count data from 1800s censuses of Canada, Great Britain, Germany, Iceland, Norway, Sweden, and the U.S. (NAPP).

We had some challenges with data. Even the two sources of data are huge, we could not find full name information except for the year 1930 and earlier. The other problem is the unbalanced counts of ethnicity main categories. Most of the population were labeled with the race White 20 million while Asian names, Chinese and Japanese and Other Asian or Pacific Islander, were summed to 130 thousands. The data also suffers from some missing data and some records needed to be cleaned. To overcome these data challenges, we wrote a script to clean and apply balanced sampling where we ended up with 310,000 (50K from the top 6 races and 10K from others) cleaned and balanced names. As seen in the original distribution of IPUMS-USA data in Table 1, we are missing a lot of other ethnicity groups like Indians or Arabs. Given the time limitation and the importance of the original main internship topic which is not about ethnicity prediction, we consider this dataset sufficient to at least differentiate some Asian ethnicity versus Latin. It would be probably a better idea to consider a language estimator instead of the ethnicity. We manage to generate a dataset of more than 50 languages but many of them have very few examples. Enriching our ethnicity and language dataset would be part of our future work.

We decided to use Linear Support Vector Classifier (LSVC) as the model to learn on our full-name-to-ethnicity dataset. Support vector classifier is a binary one. However, we use one-versus-others multi-classification version of the support vector machines. We used cross validation with 3 folds and a testset of 25% looking for the best design parameters. For feature engineering, we applied TF-IDF vectorization with n-grams ranges (1, 5). All full-names have transformed to the lower case and ASCII-fied before vectorization. In our model, we have 7 ethnicity groups. The result of this classifier is a vector of decision function computes the decision value of the input name to belong to the ethnicity class. The result vector is finally normalized with a Sigmoid function.

Having the ethnicity estimator ready, we now need to use it to construct the new pair feature. To do so, the pair of author names got transformed with the ethnicity estimator into 2 vectors of probabilities.

The vectors pair is then combined with element multiplication to form the final corresponding feature as a vector of 7 float values. As shown in Figure 3, each value represent the estimated belonging to each of the 7 ethnicity groups.

4.3.3 The Design Parameter of the Classifier

Out of many available supervised classifier, we decided to use gradient boosted regression trees (GBRT) [10]. This design decision was made in the early prototyping comparing the classification accuracy with other classifiers. In brief, Gradient Boosting produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. It builds an additive model in a forward stage-wise fashion; it generalizes them by allowing optimization of arbitrary differentiable loss functions.

We have tuned 3 design parameters: One, the maximum depth of the individual regression estimators. Another is the number of boosting stages to perform. In gradient boosting, a large number usually results in better performance. However, this is constraint to over-fitting and the computational time. The third parameter we tuned was the learning rate. A good combination between the number of estimator and the learning rate can avoid over-fitting. So, in order to find a best GBRT design parameters, we use cross-validation with 3 folds and receiver operating characteristic (ROC), or ROC curve computing the Area Under the Curve (AUC) from prediction scores [30].

4.4 Clustering

4.4.1 Hierarchical Clustering

Out of our review of related works and reflecting to our database, we decided to start with Hierarchical Agglomerative Clustering (HAC) [33]. In HAC, each observation starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy. The merges are determined in a greedy manner. The results of hierarchical clustering are usually presented in a dendrogram.

Passing the predefined affinity matrix of each block out of using the distance model, we still need to specify some design parameters for the HAC. We have used the recommended default parameters value by software library we used except for the clustering method. Based on few initial experiments, we decided to use Unweighted Pair Group Method using Arithmetic mean firstly proposed by McQuitty(UPGMA linkage)[24]. UPGMA method computes the distance between any two clusters C_i and C_j by taking the average of all distances between pairs of elements in C_i and elements in C_j , that is, the mean distance between elements of each cluster. The main properties of UPGMA method are: (1) It tends to join clusters with small variances. (2) Intermediate between single and complete link methods [19]. (3) Takes account of cluster structure. and (4) Relatively robust. WPGMA is another method, also proposed by McQuitty, which is similar to UPGMA but observations in small clusters weighted more highly than observations in large clusters. Indeed, all the mentioned methods belong to the family of Lance-Williams formula shown in Equation 2 [19] where $\alpha_i, \alpha_j, \beta$ and γ are parameters as shown in Table 2.

$$d_{(ij)k} = \alpha_i d_{ik} + \alpha_j d_{jk} + \beta d_{ij} + \gamma |d_{ik} - d_{jk}| \quad (2)$$

where:

- d_{ij} , d_{ik} , and d_{jk} be the pairwise distances between clusters C_i , C_j , and C_k , respectively.
- $d_{(ij)k}$ be the distance between the new cluster $C_i \cup C_j$ and C_k .

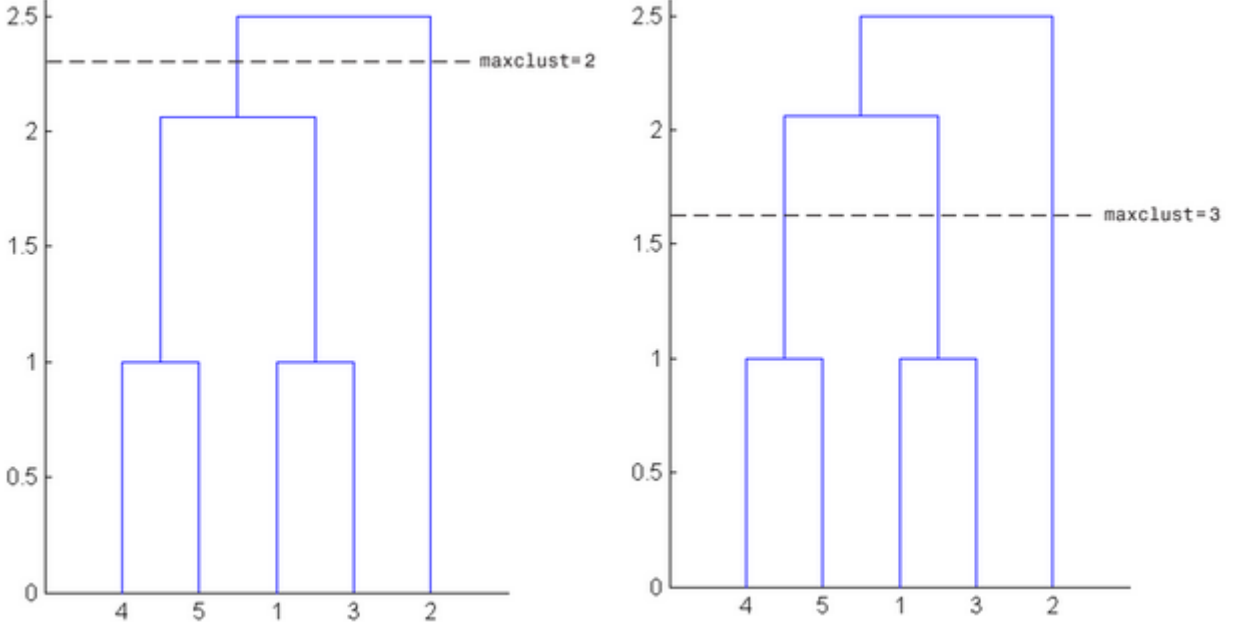
4.4.2 Estimating the Number of Clusters

In order to form a flat partitioning out of the hierarchical clustering results, we need to decide on a threshold value to cut the dendrogram and form a certain number of clusters. Each possible cut results into a different number of clusters ranging from the total number of all observations to a single cluster that groups all of the them.

Table 2: The parameters that determine the different clustering methods

Method	α_i	α_j	β	γ
Single link	0.5	0.5	0	-0.5
Complete link	0.5	0.5	0	0.5
UPGMA (Average)	$\frac{ C_i }{ C_i + C_j }$	$\frac{ C_j }{ C_i + C_j }$	0	0
WPGMA (Weighted)	0.5	0.5	0	0

Figure 4: Choosing the best threshold to form the flat clustering using a semi-supervised HAC



Learned (Semi-Supervised) The first attempt to generate a flat clustering out of the HAC results was by learning the best cut from the available ground truth. The semi-supervised algorithm simply iterates over all possible thresholds of the HAC results optimizing for the best clustering accuracy score given the ground truth. Accordingly, the algorithm will choose a cutting threshold in which the signatures belongs to the ground truth cluster will also belong to the same predicted cluster. A demo of this threshold estimation is illustrated in Figure 4 [1]. For this demo example of Figure 4, if we know from the ground-truth data that signature1 and signature5 belongs to the same real author, then the best cut would be the one that forms 2 clusters as in the left dendrogram. Forming 3 clusters with the cut shown in the right dendrogram will provide lower clustering evaluation not satisfying the ground-truth data. As there might be more than one threshold cutting that would equally satisfy the ground truth, our algorithm favors the highest threshold. This tends to merge more signatures which means less number of clusters.

The assumption is that there is sufficient percentage of claimed pairs to generalize per block. However, first, that is not the case for some blocks, and second, the ground truth data is not a well representative sample; it does not have enough observations for Asian names in which, we believe, the most challenging ambiguities comes from. We conclude that it is necessary to have the ability of estimating the best threshold in case we do not having ground truth. To do so, we have proposed 3 strategies to be evaluated: (1) Searched, (2) Simple, and (3) density-based. The experiment of these strategies will be discussed in section 5.4.2.

Searched For blocks in which we do not have any truth label, we have to find a strategy to determine the threshold to apply the flat clustering. An idea that came to mind is to compute the average of the the

learned thresholds of other blocks and apply it. A better estimation might be to take the most frequent threshold instead of the average. Even better, we can iterate over a finite set of possible thresholds and determine the best performing threshold over all blocks. The last idea seems to have a high computational cost but it is not really the case if it is implemented well; we defined the labeling function as a property of our object oriented implementation. Calling this property after changing the threshold is very fast comparing to recompute the clustering again.

So, we can for example iterate with 1000 thresholds on the range from $[0, 1]$ with a step of 0.001. The threshold that is globally the best over all blocks will be used as a default threshold for the blocks in which we do not have any true label. Although relabeling the signatures with setting threshold is fast, it is still sum up to a considerable computational time if we do it 1000 times. So, we can iterate over 100 steps first, then make another 100 smaller steps around the best found threshold of the first 100 steps search. This sums up to 200 iterations and even provides more detailed search result. Another way to optimize the searching process is to narrow the range from $[0, 1]$ to be around the average or best most frequent threshold. For example, if we found that the best most frequent threshold is 0.65, we can narrow the searching range to $[0.35, 0.9]$.

Simple Another way to tackle the problem of blocks that do not have any true labeled signature is to simply keep the block as a single cluster. This sounds not an ideal solution, however, having a good blocking strategy, for example, with the last name and first initial, it could be the best guess. One of the advantages of this strategy is being very fast having no extra computations.

Density-based There are a few metrics for clustering validation and estimating the number of clusters. Traditional clustering validity criteria could be grouped into 3 main categories: variance, density and its continuity, and separation. For our clustering algorithm we worked on metrics based on density as we already have an affinity matrix that we can reuse for clustering validation. Validity assessment approaches works better when the clusters are mostly compact. However, there are a number of applications where we have to handle arbitrarily shaped clusters (e.g. spatial data, medicine, biology) in which are not any more sufficient [14]. As part of the internship work, we have examined some of these metrics and see how good are they on the dataset.

The first implemented unsupervised methodology to choose a good threshold dynamically was using Silhouettes [27]. The implementation of using Silhouettes Score was by maximizing the score over all resulted thresholds of HAC. The mathematical notation of Silhouettes is shown in Equation 3. The threshold that leads to the best score is then used to form the flat clustering.

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))} \quad (3)$$

Where:

- $a(i)$: is the average dissimilarity of i with all other data within the same cluster, and
- $b(i)$: is the lowest average dissimilarity of i to any other cluster, of which i is not a member

Another method we have tried out was Gap Statistics in which we estimate the optimal number of cluster for data X with a given cluster estimator [31]. The estimation done by comparing distortion of clustered real data with distortion of clustered random data. However, when we implementing Gap-Statistics and compare it with Silhouettes on a subset of the dataset, we found that Silhouettes perform better and faster. So, we decided to use Silhouettes as the density-based strategy to choose the best number of clusters for blocks that do not have any labeled sample.

As discussed in the related words, DBSCAN was heavily used as a way to form clusters without explicitly determine the number of clusters as input parameter. However, DBSCAN have another design parameter to set which also define how dense is each cluster would be. Accordingly, we assume that using density-based clustering validity score like Silhouettes to decide on the cutting threshold of HAC

would lead to a similar result. We believe that using a semi-supervised HAC is better in our case as we have a sufficient proportion of claimed publications (ground-truth). Incorporating the ground truth using a semi-supervised clustering is assumed to perform better than other density based estimation of the number of clusters. We have experimented this assumption in section 5.4.2. However, experimenting DBSCAN would still be part of our future works.

5 Results and Evaluation

5.1 Pairs Sampling

A written script was available to extract a data set from the claimed publications. Claimed publications are the ones that the author ids are true verified by a human. This is a big advantage of our database in which we capitalize on for designing our disambiguation algorithm. The main concept we have in the dataset is what we call a signature. To recall, a signature is a unique combination of a publication ID and an associated author name mentioned in the publications list of authors. A ground-truth dataset was fortunately available. That is because the system has a claiming functionality in which we know the real author identity for a given publication.

A data file was extracted from the ground-truth data. The structure of that file is the true cluster IDs, the label, and the group of signature IDs, the cluster. Of course, as a ground truth, this is a hard partitioning not a fuzzy clustering. That is to say that each signature ID associated to one and only one cluster label. This would be used after to evaluate the predicted clustering against the ground-truth.

The database we are working on contains more than 1 million publication. It has more than 10 million signatures in which we 1.2 millions are labeled (ground-truth data). At the time I have started the internship, the percentage of the claimed data (ground-truth) was about ten percent. Thanks to the inspirehep.net interface, this number percentage is increasing by time. However, we have based our training-testing splitting ratio to 10% to be similar to the proportion of what ground-truth data. That being said, all the evaluation experiments would be based on 10% training versus 90% testing set.

The dataset is structured as set of pairs where each pair consists of 2 signatures ids and is labeled with a boolean value whether they are from the same author or not. If they are for the same author, the label would be 0. In the contrary, the pair of signatures that do not belong to the same real author will be labeled with 1. This labeled pairs would be used in training a distance model that would be used to generate the dissimilarity matrix of each block.

The number of possible pairs we can generate from the true clusters data file which contains 1.2 millions signatures is huge. So, we need to sample a representative set of pairs to train the distance model. There are many possible ways to sample the pairs. We have experimented 2 possible ways: first was uniformly random sampling, however, the extracted pairs were mostly easy examples like different full names being different real authors or matched full names being the same real author. In order to have more example of rare cases like matched full name being different real author, we tried another sampling method which we called balanced sampling. In this method, we balance over 4 possible cases: Same real author with the same full name, same real author but different full name, different real authors but the same name or different real authors with different full names. Since the case of different authors but same full name is relatively very rare comparing with other cases, we had to base quarter of the number of samples to the possible pairs we can get for that rare case. We also have a computational time limitation and memory limitation to take into consideration we ended up with a sample set of 1 million pairs.

Noteworthy, the pairs sampling was applied only on the training dataset while all the test would be unseen to the distance model. Accordingly, we would better evaluate the clustering algorithm that uses that distance model on the test dataset.

5.2 Accuracy Measures for Clustering

Classification accuracy measures are not only necessary for evaluation the clustering final result but also for applying our semi-supervised clustering. In this section we will mention the accuracy measure we have used.

5.2.1 Pairwise F-Measure

Precision is the ability not to label as positive a sample that is negative [20]. The best value is 1 and the worst is 0. Recall is the ability to successfully find all the positive samples. The best value is 1 and the worst is 0. F-score (Harmonic mean) can be thought as a weighted harmonic mean of the precision and recall, where an F-score reaches its best value at 1 and worst at 0. For mathematical notation shown in Equation 4 and 5. Equation 6 is the F-measure that combines precision and recall to provide a clustering accuracy score. F-measure is also called balanced F-score. To have a pair-wised F-Measure, we normalize by the number of pairs.

$$Precision(C_{true}, C_{predicted}) = \frac{|pairs(C_{true}) \cap pairs(C_{predicted})|}{|pairs(C_{predicted})|} \quad (4)$$

$$Recall(C_{true}, C_{predicted}) = \frac{|pairs(C_{true}) \cap pairs(C_{predicted})|}{|pairs(C_{true})|} \quad (5)$$

$$F(C_{true}, C_{predicted}) = \frac{2 \cdot Precision(C_{true}, C_{predicted}) \cdot Recall(C_{true}, C_{predicted})}{Precision(C_{true}, C_{predicted}) + Recall(C_{true}, C_{predicted})} \quad (6)$$

5.2.2 BCubed-F-Score

BCubed-F-score, or B^3 is the same as pairwise F-measure except that we normalize over the number of signatures rather than the number of pairs [3]. Using this measure that counts for the cluster size, we can better judge our overall desired clustering comparing to the human judgment. In our experiments, we used BCubed-F-score as our based evaluation metric. B^3 looks at each article and asks what the intersection is between the cluster for that article author in the ground-truth clustering and the cluster for that article author in the predicted clustering [20]. Given a set of articles A , the B^3 -F-score is defined by the Equations 7, 8, 9, and 10.

$$cluster(C, a) = c : c \in C \wedge a \in c \quad (7)$$

$$Precision(A, C_{true}, C_{predicted}) = \frac{1}{|A|} \sum_{a \in A} \frac{|cluster(C_{true}, a) \cap cluster(C_{predicted}, a)|}{|cluster(C_{predicted}, a)|} \quad (8)$$

$$Recall(A, C_{true}, C_{predicted}) = \frac{1}{|A|} \sum_{a \in A} \frac{|cluster(C_{true}, a) \cap cluster(C_{predicted}, a)|}{|cluster(C_{true}, a)|} \quad (9)$$

$$F(A, C_{true}, C_{predicted}) = \frac{2 \cdot Precision(A, C_{true}, C_{predicted}) \cdot Recall(A, C_{true}, C_{predicted})}{Precision(A, C_{true}, C_{predicted}) + Recall(A, C_{true}, C_{predicted})} \quad (10)$$

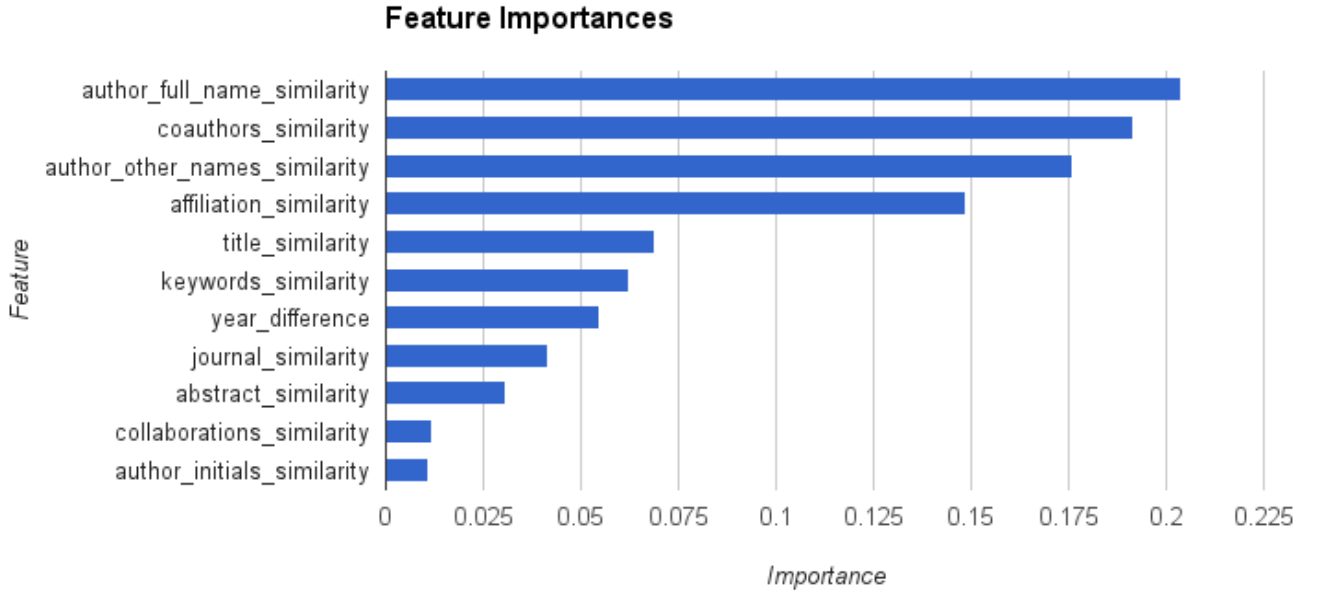
5.3 Distance Model

5.3.1 Feature Selection

The list of features we used are:

- Author full name similarity
- Coauthors similarity

Figure 5: Features importance chart

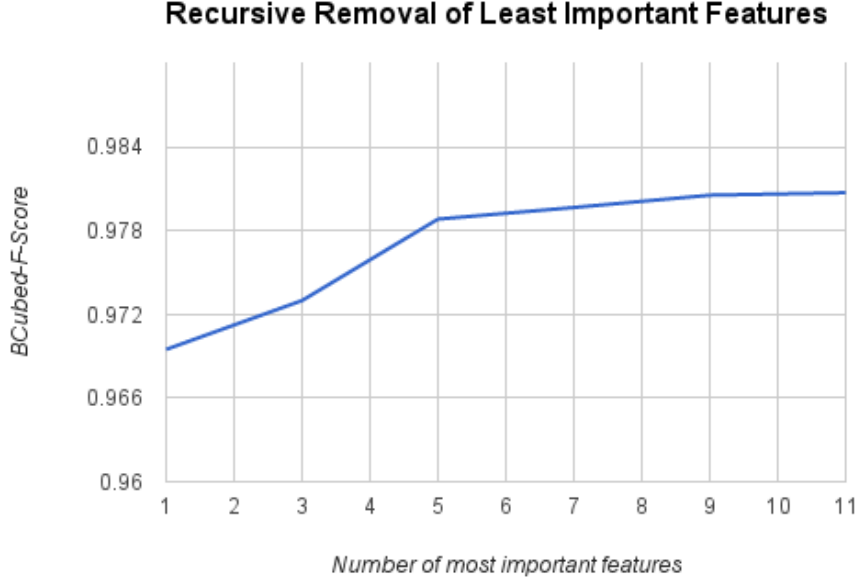


- Author other names similarity
- Affiliation similarity
- Title similarity
- Keywords similarity
- Year difference
- Journal similarity
- Abstract similarity
- Collaborations similarity
- Author initials similarity

In order to select the best features, we tried 3 evaluation criteria. First one was to consider each feature at a time and evaluate the classifier accuracy score. Another is by using feature importances analysis attribute that is built in the machine learning library we used which is presented in section 4.4.2. The third method was to apply recursive elimination of the features set. The first method of using one-at-a-time score function, I have ranked the features and accordingly tried a few combinations on a sample of the dataset. The top 3 features together performed 95% comparing to all of them. However, the other 2 methods are better evaluation since they consider a combination of features rather than one at a time.

Feature Evaluation The GBRT, presented in section 4.3.3, has a nice attribute by which it provide importance distribution of the features. The features importance distribution of all features we have used is shown in Figure 5. We can see that some features have very low importance percentage. later in this section, we will evaluate the clustering results when we remove some of these least important features. Looking at Figure 5, we can tell that if we keep the top 4 features, the model would still perform well.

Figure 6: Features recursive elimination from the least important



Recursive Features Elimination Here we tried to recursively remove 2 features at a time from the least importance to the best and see how the clustering accuracy is affected. As we can see in Figure 6, the accuracy score start to drop when we use less than 5 features. Including all the features and applying it to 10% of the whole data took about four hours while using only the most important features took only 48 minutes. However, we noticed that 1 of the features, coauthors similarity, was the most time consuming feature, 2:30 hours, even though it was the second most important one. So, we run an experiment using all the features except that one. The run took about two hours and the accuracy score only reduced from 0.9807 to 0.9789.

5.3.2 Estimating the Ethnicity from the Full Name

Comparing the clustering results with and without adding the ethnicity feature, we found that the new feature actually enhances the accuracy as shown in Figure 7. The comparison made on the best clustering strategy we can up with which will be shown in section 3.4.2. A shown in Figure 7, the number of predicted clusters with the new feature is also closer to the true number number of clusters. We show how the new feature importances distribution with the new added ethnicity features separated, in Figure 8, and combined, in Figure 9. We can see that the Chinese ethnicity feature scored the highest comparing with the other ethnicity features. This is aligned with our assumption on the specialty of Asian names and Chinese names in particular.

5.4 Clustering Evaluation

5.4.1 Parameters Tuning

The main design parameter we want to evaluate in HAC is the method. We have run an experiment to compare 4 methods: Single, complete, average, and weighted. The results, as in Figure 10, show that the best was the average method (UPGMA) which is described in section 2.4.1.

5.4.2 Number of Clusters Strategies

For each the blocking criteria (Double-Metaphone and Last Name First name Initial), we evaluated 3 strategies of dealing with the blocks that do not have any labeled sample, as discussed in section 4.4.2.

Figure 7: (left) Comparing the absolute difference with the true number of clusters; (right) BCubed accuracy comparison between having the ethnicity feature and without having it

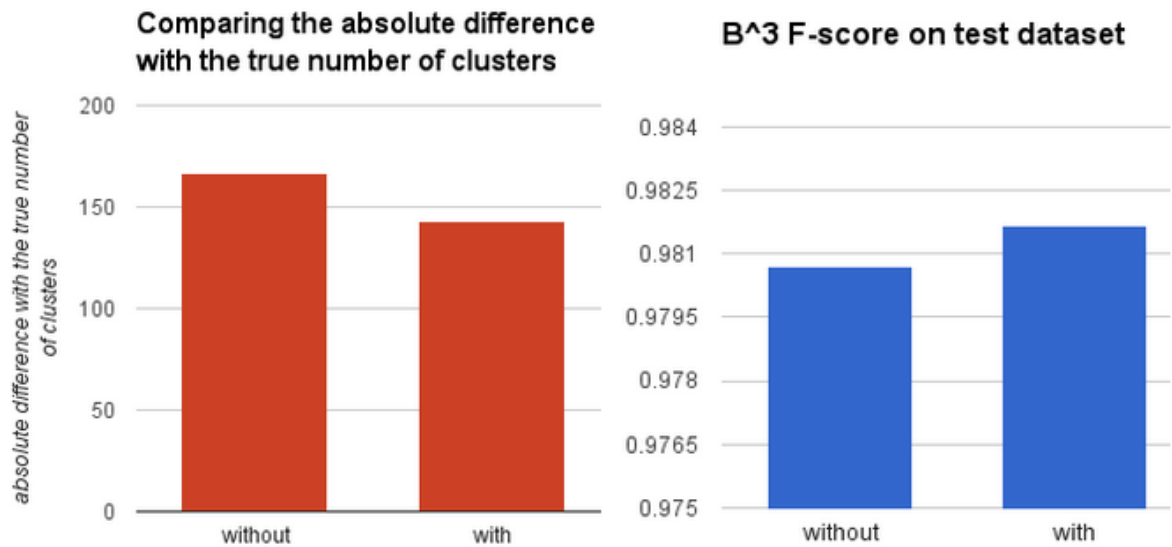


Figure 8: Features importance chart including ethnicity features

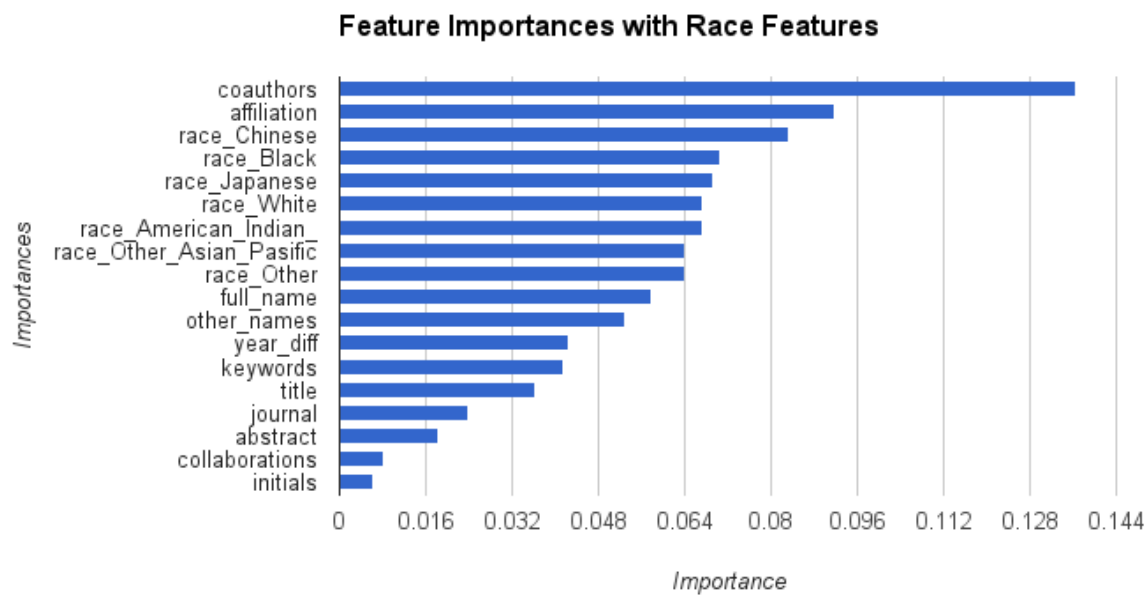


Figure 9: Features importance chart including the combined ethnicity feature

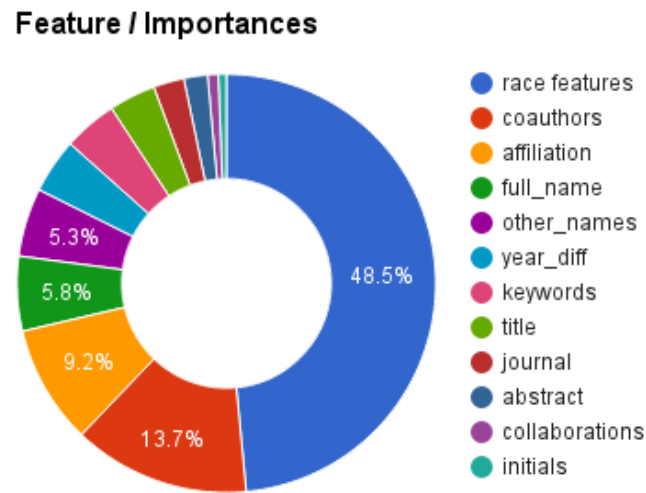


Figure 10: Comparing different clustering methods

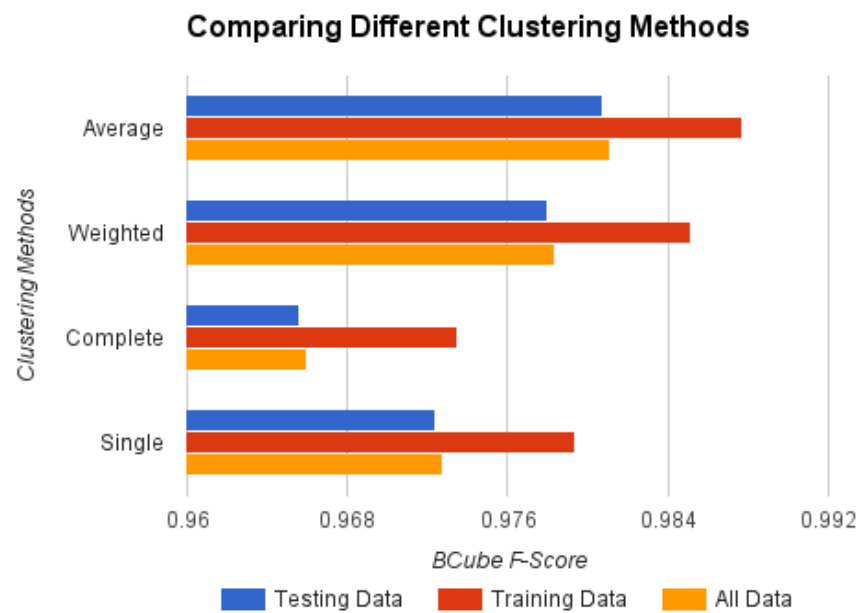
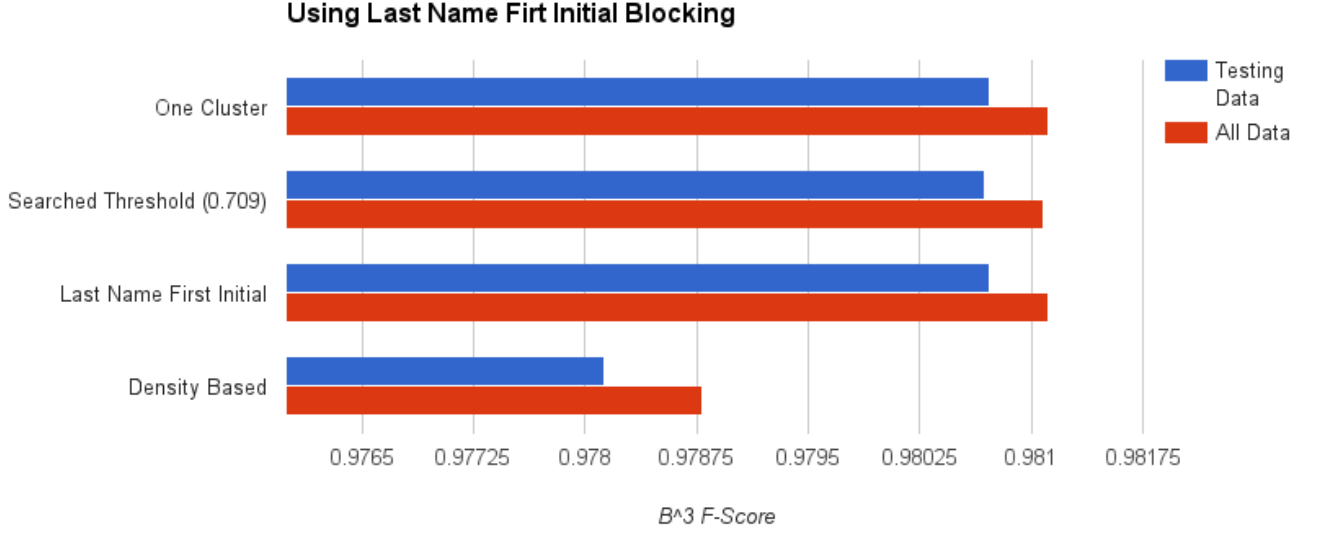


Figure 11: Comparing the different strategies applied on last name first initial blocking based on the clustering evaluation score



The baseline in both of the blocking strategies is just consider such a block as a single cluster.

Last Name and First Name Initial (LNFI) Blocking In this experiment, the baseline is identical with the last name first initial strategy as the input blocking strategy is the same. Results shown in Figure 11 shows that last name first initial is slightly better than the searched threshold strategy. This is given that the last name first initial does not cost any additional computational time. In Figure 12 we can see how close each strategy predicted number of clusters is to the true number of clusters.

Double-Metaphone Blocking As we can see in Figure 13, the best 2 strategies were the searched threshold and the last name first initial with very close scoring results. However, having the computational overhead using the search threshold strategy much higher than the last name first initial, we consider the last a better strategy. In term of the closeness to the true number of clusters, density based strategy was the best as shown in Figure 14. comparing with the number of clusters of the other 2 strategies, density-based strategy tends to split the block into a higher number of clusters. In the case of Double-Metaphone, this was a good attribute having bigger block sizes than the last name first initial blocking.

Finally, To decide between using Double-Metaphone and Last Name First Initial blocking functions, we compare the best performing strategies of both. As shown in Figure 15, the best strategy is the Last Name First name Initial.

5.4.3 Evaluation of Pairs Sampling Strategies

We discussed the pairs sampling in section 5.1. Here we show a comparison between using a balanced pairs sampling versus uniformly random pairs sampling in Figure 16. The experiment showed that having a balanced training dataset perform better than using uniformly random sampled pairs to train the distance model. In Figure 17, we show that the number of predicted clusters using balanced pairs sampling is closer to the true number of clusters than the uniformly random sampling strategy.

Figure 12: Comparing the different strategies applied on last name first initial blocking based on the predicted number of clusters

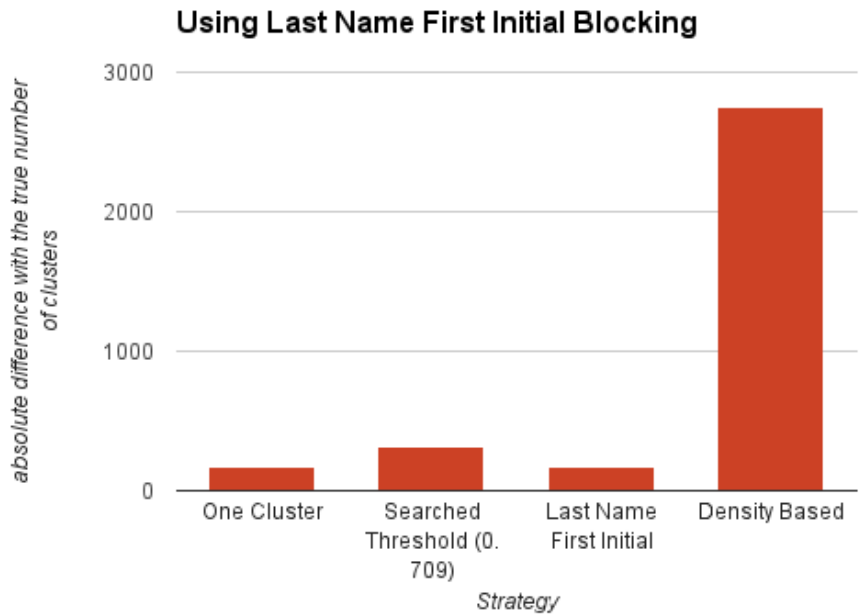


Figure 13: Comparing the different strategies applied on double metaphone blocking based on the clustering evaluation score

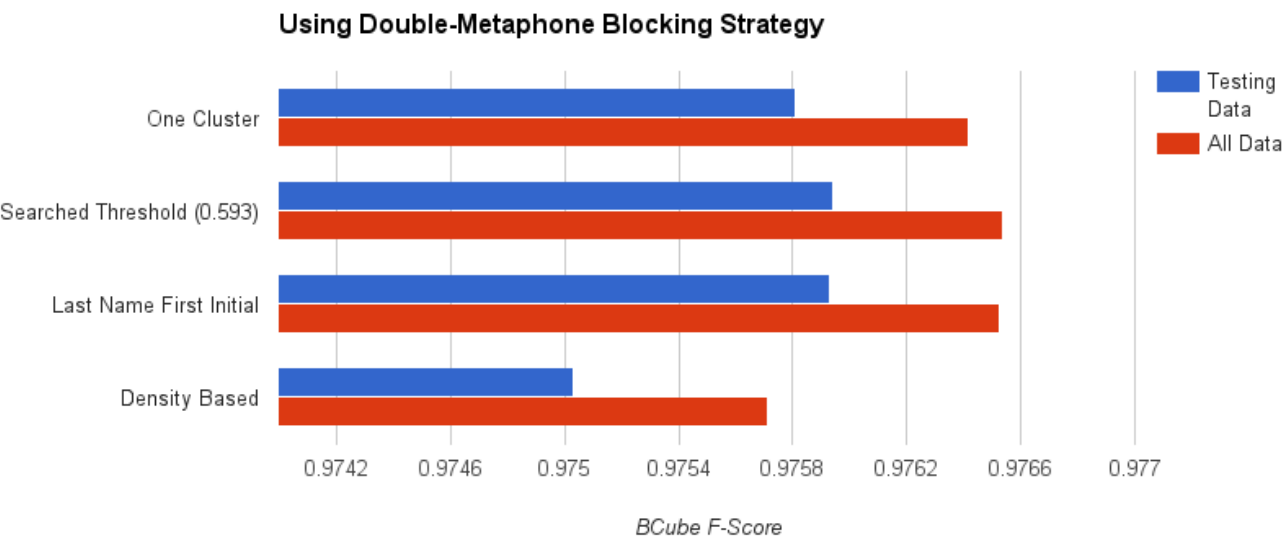


Figure 14: Comparing the different strategies applied on Double Metaphone blocking based on the predicted number of clusters

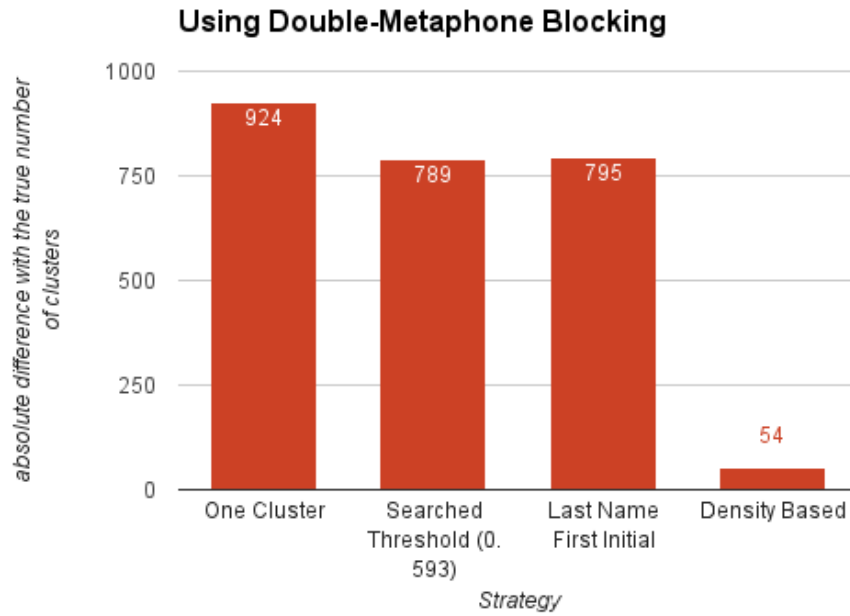


Figure 15: Comparing the best strategies of both Double Metaphone blocking and Last Name First Initial

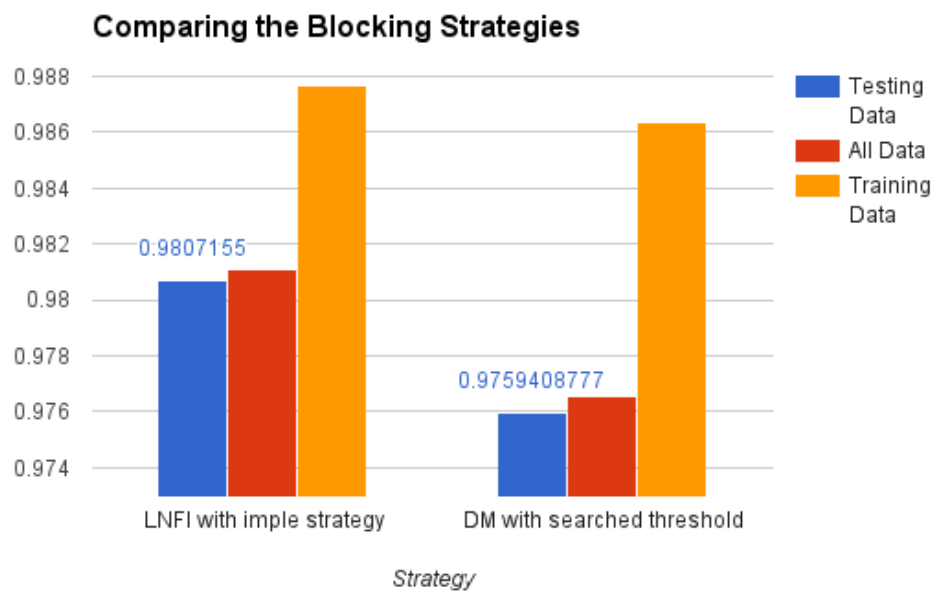


Figure 16: Comparing the pairs sampling strategies based on the clustering evaluation score

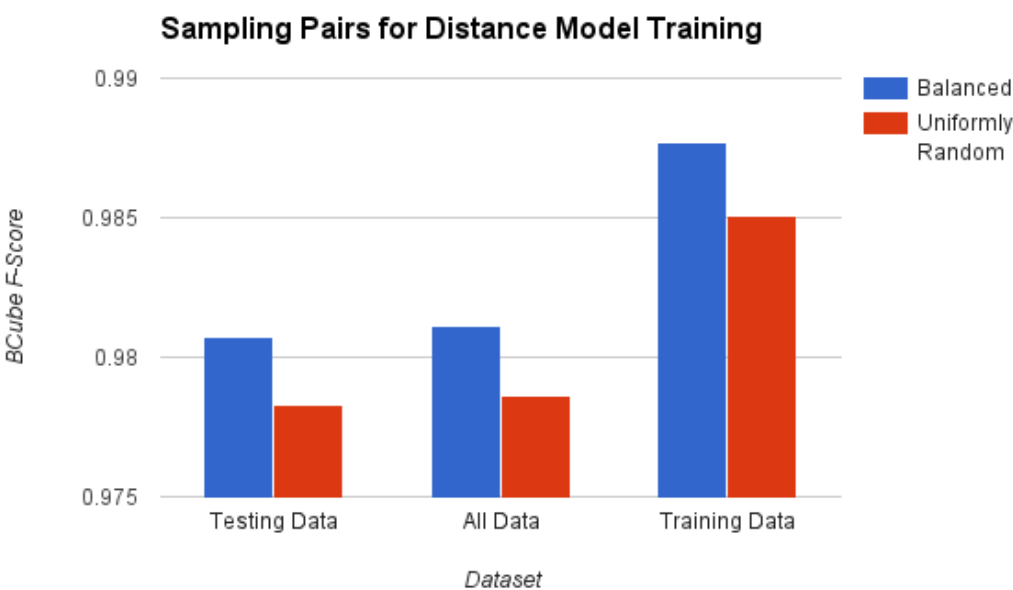


Figure 17: Comparing the pairs sampling strategies based on the number of predicted clusters



6 Implementation

6.1 BEARD: Open Source Library for Automatic Entity Disambiguation

Bibliographic Entity Automatic Recognition and Disambiguation (BEARD) is an open source library that I took part of developing it during my internship at CERN. It is designed to be deployed either separately or mainly as a module to a bigger open-source software called invenio. Invenio also developed at maintained at CERN. Now Invenio is used in many digital libraries like inspirehep.net, [Zenodo.org](http://zenodo.org), infoscience.epfl.ch, traces.uab.cat, romdoc.upb.ro, and more. Eventually, BEARD will be deployed to CERN high-energy physic digital library (<http://inspirehep.net>). Any new knowledge out of human curation or author claiming feedback came from the user web interface will be in reflected into the semi-supervised clustering model which is plan to run periodically, for example, daily. Cases of having claimed publications of an author with different last name first initial variation will force the author identification in the database over the results disambiguation model.

6.2 Python Scikit-Learn and API compatibility

BEARD is a scikit-learn API compatible library built in Python. Scikit-learn API consists of three complementary interfaces: an estimator interface for building and fitting models, a predictor interface for making predictions and a transformer interface for transforming data [6]. BEARD is designed in a highly optimized and efficient computing implementation supporting parallel processing. It also utilizes SciPy and NumPy [17] for fully vectorized data representation and mathematical operation. SciPy, in which we used its hierarchical clustering implementation, is open-source software for mathematics, science, and engineering. It includes modules for statistics, optimization, integration, linear algebra, Fourier transforms, signal and image processing.

Scikit-learn API has a distinguished attribute which is the ability to compose new estimators from several base estimators. This is very helpful in combining typical machine learning workflows into a single object which is itself an estimator. Composition of estimators can be done in two ways: either in a parallel fashion through FeatureUnion objects or sequentially through Pipeline objects [6]. Another nice feature in scikit-learn is Grid-Search-Cross-Validation which was very helpful in selecting the best model evaluating a combination of design parameters of an estimator. Among a lot of very useful data preparation functions, TFIDF vectorizer was one of key importance functions we utilized. It allowed us to easily select the n-gram range in the feature engineering process.

7 Conclusion and Future Works

Using same author names variation by many authors in the databases of the digital libraries causes a problem in distinguishing who is the real author of a publication. Matched author names usually grouped as one author profile while they are more than one real author. The other way around also causes a problem when the same real author use different names in different publications. There are many machine learning based related work to solve this problem. Our work provided high accuracy using semi-supervised clustering on blocks of similar author names publications. We used a learned distance model on labeled pairs in which we extracted a set of features among them the estimated ethnicities of the pair of names. My contribution to the distance model reduced about 35% of the error rate.

Utilizing the available ground-truth data, we managed to take the best design decision. We evaluated different strategies to estimate the best number of clusters for the blocks we do not have any labeled sample. Results showed that the best strategy is to use last name first initial. It is also computationally the fastest strategy comparing with iterating over range of global cutting thresholds or using density-based clustering score to find the best number of clusters. Balanced sampling that over-sample difficult and rare cases like same real author with different names, is also better than applying uniformly random sampling from the ground truth data. In term of hierarchical clustering method, the best performing one running an experiment on our dataset was the average method.

Incorporating an external knowledge of the name ethnicity to the distance model showed a promising enhancement that motivates further work. We are planning to collect a better dataset of not only ethnicity but also languages of full names. Another future work would be evaluating DBSCAN against our semi-supervised model. Another item we want to explore further is the blocking strategy. Last Names First Initial is a strict condition even though it perform better than Double-Metaphone grouping on the last name. We believe that we can find a better strategy that is less strict than last name first initial and more strict than Double-Metaphone on the last name.

In most of related works, the common challenge was always to find a sufficient and representing ground-truth data. We believe that our ground truth data was sufficient, thanks to the claiming feature of the digital library web service. However, we are not sure of the diversity and the represent-ability of it. The ground truth data does not cover for example as much Asian names as Western names. This increases the need of working further on the ethnicity and language estimation from the full names. Including and evaluation more features like the publication topics will also be part of the future works.

This work will be eventually deployed for CERN digital library of high-energy physics. This deployment will allow us to better evaluate and continuously enhance the disambiguation model. We can get feedback from the authors themselves and from the digital library editors who are currently doing manual curation of the data.

References

- [1] <http://ch.mathworks.com/help/stats/hierarchical-clustering.html>.
- [2] Hussein T Al-Natsheh and Taisir M Eldos. Performance optimization of adaptive resonance neural networks using genetic algorithms. In *Foundations of Computational Intelligence, 2007. FOCI 2007. IEEE Symposium on*, pages 143–148. IEEE, 2007.
- [3] Enrique Amigó, Julio Gonzalo, Javier Artiles, and Felisa Verdejo. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information retrieval*, 12(4):461–486, 2009.
- [4] Indrajit Bhattacharya and Lise Getoor. A latent dirichlet model for unsupervised entity resolution. In *SDM*, volume 5, page 59. SIAM, 2006.
- [5] Indrajit Bhattacharya and Lise Getoor. Collective entity resolution in relational data. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):5, 2007.
- [6] Lars Buitinck, Gilles Louppe, and Mathieu Blondel. Api design for machine learning software: experiences from the scikit-learn project. In *ECML/PKDD 2013 Workshop: Languages for Data Mining and Machine Learning*, 2013.
- [7] Gail A Carpenter, Stephen Grossberg, and Michael A Arbib. Adaptive resonance theory. *Encyclopedia of Machine Learning*, 1987.
- [8] Aron Culotta, Pallika Kanani, Robert Hall, Michael Wick, and Andrew McCallum. Author disambiguation using error-driven machine learning with a ranking loss function. In *Sixth International Workshop on Information Integration on the Web (IIWeb-07), Vancouver, Canada, 2007*.
- [9] Anderson A Ferreira, Marcos André Gonçalves, and Alberto HF Laender. A brief survey of automatic methods for author name disambiguation. *Acm Sigmod Record*, 41(2):15–26, 2012.
- [10] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.
- [11] André Fujita, Daniel Y Takahashi, and Alexandre G Patriota. A non-parametric method to estimate the number of clusters. *Computational Statistics & Data Analysis*, 73:27–39, 2014.
- [12] Dr. Lise Getoor and Dr. Ashwin Machanavajjhala. Entity resolution for big data: A summary of the kdd 2013 tutorial, 08 2013.
- [13] Anja Gruenheid, Xin Luna Dong, and Divesh Srivastava. Incremental record linkage. *Proceedings of the VLDB Endowment*, 7(9):697–708, 2014.
- [14] Maria Halkidi, Yannis Batistakis, and Michalis Vazirgiannis. Clustering validity checking methods: part ii. *ACM Sigmod Record*, 31(3):19–27, 2002.
- [15] Paul Jaccard. The distribution of the flora in the alpine zone. *New phytologist*, 11(2):37–50, 1912.
- [16] Wenrong Jian, Anbao Wang, Cuihong Wu, Jian Chen, and Jihong Yan. Approach for name ambiguity problem using a multiple-layer clustering. In *Computational Science and Engineering, 2009. CSE’09. International Conference on*, volume 4, pages 874–878. IEEE, 2009.
- [17] Eric Jones, Travis Oliphant, P Peterson, et al. Open source scientific tools for python, 2001.
- [18] Madian Khabsa, Pucktada Treeratpituk, and C Lee Giles. Large scale author name disambiguation in digital libraries. In *Big Data (Big Data), 2014 IEEE International Conference on*, pages 41–42. IEEE, 2014.
- [19] Godfrey N Lance and William Thomas Williams. A general theory of classificatory sorting strategies ii. clustering systems. *The computer journal*, 10(3):271–277, 1967.

- [20] Michael Levin, Stefan Krawczyk, Steven Bethard, and Dan Jurafsky. Citation-based bootstrapping for large-scale author disambiguation. *Journal of the American Society for Information Science and Technology*, 63(5):1030–1047, 2012.
- [21] Chun-Liang Li, Yu-Chuan Su, Ting-Wei Lin, Cheng-Hao Tsai, Wei-Cheng Chang, Kuan-Hao Huang, Tzu-Ming Kuo, Shan-Wei Lin, Young-San Lin, Yu-Chen Lu, et al. Combination of feature engineering and ranking models for paper-author identification in kdd cup 2013. In *Proceedings of the 2013 KDD Cup 2013 Workshop*, page 2. ACM, 2013.
- [22] Shaohua Li, Gao Cong, and Chunyan Miao. Author name disambiguation using a new categorical distribution similarity. In *Machine learning and knowledge discovery in databases*, pages 569–584. Springer, 2012.
- [23] Hans Peter Luhn. A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of research and development*, 1(4):309–317, 1957.
- [24] Louis L McQuitty. Improved hierarchical syndrome analysis of discrete and continuous data. *Educational and Psychological Measurement*, 26(3):577–582, 1966.
- [25] Staša Milojević. Accuracy of simple, initials-based methods for author name disambiguation. *Journal of Informetrics*, 7(4):767–773, 2013.
- [26] Lawrence Philips. Hanging on the metaphone. *Computer Language*, 7(12 (December)), 1990.
- [27] Peter Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20(1):53–65, 1987.
- [28] Amit Singhal. Modern information retrieval: A brief overview. *IEEE Data Eng. Bull.*, 24(4):35–43, 2001.
- [29] Qun Song and Nikola Kasabov. Ecm-a novel on-line, evolving clustering method and its applications. *Foundations of cognitive science*, pages 631–682, 2001.
- [30] JA Swets. Indices of discrimination or diagnostic accuracy signal detection theory and roc analysis in psychology and diagnostics: Collected papers, 1996.
- [31] Robert Tibshirani, Guenther Walther, and Trevor Hastie. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):411–423, 2001.
- [32] Vetle I Torvik, Marc Weeber, Don R Swanson, and Neil R Smalheiser. A probabilistic similarity metric for medline records: A model for author name disambiguation. *Journal of the American Society for information science and technology*, 56(2):140–158, 2005.
- [33] Joe H Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244, 1963.