Ethnicity sensitive author disambiguation of semi-labeled corpora of scientific articles

Gilles Louppe CERN Switzerland
Hussein Al-Natsheh CERN Switzerland
Mateusz Susik CERN Switzerland
Eamonn Maguire CERN Switzerland

Abstract

Author name disambiguation in bibliographic databases is the problem of grouping together scientific publications written by a same person, accounting for potential homonyms and/or synonyms. Among solutions to this problem, digital libraries are increasingly offering tools for authors to manually curate their publications and claim which ones are theirs. Indirectly, these tools allow for the inexpensive collection of large annotated training data, which can be further leveraged to build a complementary automated disambiguation system capable of inferring patterns for identifying publications written by a same person. Building upon more than 1 million crowdsourced annotations, we propose an automated author disambiguation solution exploiting this data (i) to learn an accurate classifier for identifying corefering authors and (ii) to guide the clustering of scientific publications by distinct authors in a semi-supervised way. To the best of our knowledge, our analysis is the first to be carried out on data of this size and coverage. It reveals that... [GL: To be completed with actual results.]

1 Introduction

In academic digital libraries, author name disambiguation is the problem of grouping together publications written by a same person. Author name disambiguation is often a difficult problem because an author may use different spellings or name variants across her career (synonymy) and/or distinct authors may share the same name (polysemy). Most notably, author disambiguation is often even more troublesome in the case of researchers from non-Western cultures, where personal names may be traditionally less diverse (leading to homonym issues) or for which transliteration to Latin characters may not be unique (leading to synonym issues). With the fast growth of the scientific literature, author disambiguation has become a pressing issue because the accuracy of information managed at the level of individuals directly affects the relevance search of results (e.g., when querying for all publications written by a given author), the reliability of bibliometrics and author rankings (e.g., citation counts or other impact metrics, as studied in (Strotmann and Zhao, 2012)) or the relevance of scientific network analysis. [GL: Add more references.]

Efforts and solutions to author disambiguation have been proposed from various communities (Liu et al., 2014). On the one hand, libraries have maintained authorship control through manual curation, either in a centralized way by hiring collaborators or by developing services inviting authors to register their publications themselves (e.g., Google Scholar, Inspire-HEP, ...). Recently, efforts also started to create persistent digital identifiers assigned to researchers (e.g., ORCID, ResearchedID, ...), with the further objective to embed these identifiers in the submission workflow of publishers or repositories (e.g., Elsevier, arXiv, Inspire-HEP, ...), thereby univocally solving any disambiguation issue. With the large cost of centralized manual authorship control, or until the larger adoption of crowdsourced solu-

tions, the impact of these efforts are unfortunately limited by the efficiency, motivation and integrity of their active contributors. Similarly, the success of persistent digital identifier efforts is conditioned to a large and ubiquitous adoption by both researchers and publishers. For these reasons, fully automated machine learning-based methods have been proposed during the past decade to provide immediate, less costly and efficient solutions to author disambiguation. In this work, our goal is to explore and demonstrate how both approaches can coexist and benefit from each other. In particular, we study how labeled data obtained through manual curation (either centralized or crowdsourced) can be exploited (i) to learn an accurate classifier for identifying corefering authors and (ii) to guide the clustering of scientific publications by distinct authors in a semi-supervised way. Our analysis of parameters and features on this large training set reveals that...

[GL: To be completed with actual results.]
[GL: Also mention error analysis, suggesting that uncertain disambiguation results should be suggested to human curators, thereby closing the loop and creating a virtuous circle.]
[GL: New idea: phonetization in blocking]
[GL: New idea: ethnicity features]

The remaining of this report is structured as follows. In Section 2, we first briefly review machine learning solutions for author disambiguation. The components of our method are then defined in Section 3 and its implementation described in Section 4. Experiments are carried out in Section 5, where we explore and validate features for the supervised learning of a linkage function and compare strategies for the semi-supervised clustering of publications. Finally, conclusions and future works are discussed in Section 6.

2 Related works

As reviewed in (Smalheiser and Torvik, 2009; Ferreira et al., 2012; Levin et al., 2012), author disambiguation algorithms are usually composed of two main components: (i) a linkage function determining whether two publications have been written by a same author and (ii) a clustering algorithm producing clusters of publications assumed to be written by a same author. Approaches can be inventoried along several axes, depending on the type and amount of data available, the way the linkage function is learned or defined, or the clustering procedure used to group publications. Methods relying on supervised learning usually make use of a small set of hand-labeled pairs of publications identified as being either from the same or from different authors for learning automatically a linkage function between publications (Han et al., 2004; Huang et al., 2006; Culotta et al., 2007; Treeratpituk and Giles, 2009; Tran et al., 2014). Because training data is often not easily available, unsupervised approaches propose instead to use domain-specific, often manually designed, linkage functions tailored for author disambiguation (Malin, 2005; McRae-Spencer and Shadbolt, 2006; Song et al., 2007; Soler, 2007; Kang et al., 2009; Fan et al., 2011; Schulz et al., 2014). These later approaches have the advantage of not requiring hand-labeled data, but often do not perform as well as the supervised approaches. To reconcile both worlds, semi-supervised methods make use of small, manually verified, clusters of publications and/or of high-precision domain-specific rules to build a training set of pairs of publications, from which a linkage function is then built using supervised learning (Ferreira et al., 2010; Torvik and Smalheiser, 2009; Levin et al., 2012). Coincidentally, semi-supervised approaches also allow for the tuning of the clustering algorithm when the later is applied to a mixed set of labeled and unlabeled publications, e.g., by maximizing some clustering performance metric on the known clusters (Levin et al., 2012).

Because of the lack of a large and publicly available dataset of curated clusters of publications, studies on author disambiguation are usually constrained to validate their results on manually built datasets of limited size and scope (from a few hundreds to a few thousands of papers, with a sparse coverage of ambiguous cases), making the true performance of these methods often difficult to assess with high confidence. In addition, despite devoted efforts to construct them, these datasets are very rarely released publicly, making it even more difficult to compare methods on a common benchmark.

s :	Title	Lorem ipsum dolor sit amet, consectetur adipiscing elit
	Author	Doe, John
	Affiliation	University of Foo
	Co-authors	Smith, John; Chen, Wang
	Year	2015

Figure 1: An example of signature s for "Doe, John". A *signature* is defined as unique piece of information identifying an author on a publication, along with any other metadata that can be derived from it, such as publication title, co-authors or date of publication.

In this context, we position the work presented in this paper as a semi-supervised solution for author disambiguation, but with the significant advantage of having a very large collection of more than 1 million crowdsourced annotations of publications whose true authors are identified. In particular, the extent and coverage of this data allows us to revisit, validate and nuance previous findings regarding supervised learning of linkage functions and to better explore strategies for semi-supervised clustering. Additionally, by releasing the data publicly, we hope to set a benchmark on which further research on author disambiguation and related topics can build upon.

3 Semi-supervised author disambiguation

Formally, let us assume a set of publications $\mathcal{P} = \{p_0, ..., p_{N-1}\}$ along with the set of unique individuals $\mathcal{A} = \{a_0, ..., a_{M-1}\}$ having together authored all publications in $\mathcal{P}$. Let us define a signature $s \in p$ from a publication as a unique piece of information identifying one of the authors of p (e.g., the author name, his affiliation, along with any other metadata that can be derived from p, as illustrated in Figure 3). Let us denote by $\mathcal{S} = \{s | s \in p, p \in \mathcal{P}\}$ the set of all signatures that can be extracted from all publications in $\mathcal{P}$ and by $S = |\mathcal{S}|$ the total number of signatures.

In this framework, author disambiguation can be stated as the problem of finding a partition $\mathcal{C} = \{c_0, ..., c_{M-1}\}$ of $\mathcal{S}$ such that $\mathcal{S} = \cup_{i=0}^{M-1} c_i$, $c_i \cap c_j = \phi$ for all $i \neq j$, and where subsets c_i, or clusters, each corresponds to the set of all signatures belonging to the same individual a_i. Alternatively, the set $\mathcal{A}$ may remain (possibly partially) unknown, such that author disambiguation boils down to finding a partition $\mathcal{C}$ where subsets c_i each corresponds to the set of all signatures from a same individual (without knowing who). Finally, in the case of partially annotated databases as studied in this work, the setting extends with the partial knowledge $\mathcal{C}' = \{c'_0, ..., c'_{M-1}\}$ of $\mathcal{C}$, such that $c'_i \subseteq c_i$, with c'_i possibly empty. Or put otherwise, the setting extends with the knowledge that all signatures $s \in c'_i$ belong to the same author.

As inspired from several previous works described in Section 2, we cast in this work author disambiguation into a semi-supervised clustering problem. Our algorithm is made of three parts: (i) a blocking scheme whose goal is to roughly pre-cluster signatures $\mathcal{S}$ into smaller groups in order to reduce computational complexity; (ii) the construction of a linkage function d between signatures using supervised learning; (iii) the semi-supervised clustering of all signatures within a same block, using d as a pseudo distance metric.

[GL: Figure required here (Eamonn)]

3.1 Blocking

As in previous works, the first part of our algorithm consists in dividing signatures $\mathcal{S}$ into disjoint subsets $\mathcal{S}_{b_0}, ..., \mathcal{S}_{b_{K-1}}$, or *blocks* (Fellegi and Sunter, 1969), and then to carry out author disambiguation independently on each one of these blocks. In doing so, computational complexity of clustering (see Section 3.3) typically reduces from $O(S^2)$ to $O(\sum_b S_b^2)$, which becomes much more tractable as the number of signatures increases. Since disambiguation is performed independently per block, a good blocking strategy should ideally be designed such that signatures from the same author are all mapped to the same block,

otherwise directly preventing their correct clustering in later stages. As a result, blocking should balance between reduced complexity and maximum recall.

The simplest and most common strategy for blocking, further denoted to as SFI, groups signatures together if they share the same surname(s) and the same first given name initial (e.g., "Doe, J"). Despite satisfying performance, such a simple strategy fails in several cases, as listed below for pairs of signatures that should be clustered together but that are put into distinct blocks:

1. There are different ways of transcripting an author name or signatures contain a typo (e.g., "Mueller, R." and "Muller, R.", "Tchaikovsky, P." and "Czajkowski, P.").
2. An author has multiple surnames and, on some signatures, all the surnames but the last one are contained within given names (e.g., "Martinez Torres, A." and "Torres, A. Martinez").
3. An author has multiple surnames and, on some signatures, only the first surname is present (e.g., "Smith-Jones, A." and "Smith, A.")
4. An author as multiple given names and they are not always all mentioned (e.g., "Smith, Jack" and "Smith, A. J.")
5. An author changed his surname (e.g., because of marrying somebody).

To account for these issues we propose instead to block signatures based on the phonetic representation of the normalized surname. Normalization consists in stripping accents (e.g., "Jabłoński, Ł" $\rightarrow$ "Jablonski, L") and name affixes that inconsistently appear in signatures (e.g., "van der Waals, J. D." $\rightarrow$ "Waals, J. D."), while phonetization is based either on the Double Metaphone (Philips, 2000), the NYSIIS (Taft, 1970) or the Soundex (The National Archives, 2007) algorithms. Together, these processing steps allow to group most name variants of a same person in the same block while not significantly increasing computational complexity, thereby solving case 1.

In the case of multiple surnames (cases 2 and 3), we further propose to block signatures in two phases. In the first phase, all the signatures with one surname are clustered together. Every different surname token creates a new block. In the second phase, the signatures with multiple surnames are compared with the blocks for the first and last surname. If the first surnames of an author were already used as the last given names on some of the signatures, the new signature is assigned to the block of the last surname (case 2). Otherwise, the signature is assigned to the block of the first surname (case 3). Finally, to prevent the creation of too large blocks, signatures are further divided along their first given name initial. Cases 4 and 5 are not explicitly handled.

3.2 Linkage function

Supervised classification. The second part of the algorithm is the automatic construction of a pair-wise linkage function between signatures, for later use during clustering to group all signatures from a same author. Formally, the goal is to build a function $d : \mathcal{S} \times \mathcal{S} \mapsto [0, 1]$, such that $d(s_1, s_2)$ tends to 0 if both signatures s_1 and s_2 belong to the same author, and 1 otherwise. Coincidentally, this problem can be cast as a standard supervised classification task, where inputs are pairs of signatures and outputs are classes 0 (same authors) and 1 (distinct authors). In this work, we evaluate Random Forests (RF, Breiman (2001)), Gradient Boosted Regression Trees (GBRT, Friedman (2001)) and Logistic Regression (Fan et al., 2008) as classifiers.

Input features. In most cases, supervised learning algorithms assume the input space $\mathcal{X}$ to be numeric (e.g., $\mathbb{R}^p$), making them not directly applicable to structured input spaces such as $\mathcal{S} \times \mathcal{S}$. Following previous works [GL: add references], pairs of signatures (s_1, s_2) are first transformed to vectors $v \in \mathbb{R}^p$ by building so-called similarity profiles (Treeratpituk and Giles, 2009), and on which supervised learning is then carried on. In this work, we design and evaluate 15 standard input features based on the comparison of signature fields, as reported in the first half of Table 1. As an illustrative example, the *Full name* feature

Table 1: Input features for learning a linkage function

Feature	**Combination operator**
Full name	Cosine similarity of $(2,4)$-TF-IDF
Given names	Cosine similarity of $(2,4)$-TF-IDF
First given name	Jaro-Winkler distance
Second given name	Jaro-Winkler distance
Given name initial	Equality
Affiliation	Cosine similarity of $(2,4)$-TF-IDF
Co-authors	Cosine similarity of TF-IDF
Title	Cosine similarity of $(2,4)$-TF-IDF
Journal	Cosine similarity of $(2,4)$-TF-IDF
Abstract	Cosine similarity of TF-IDF
Keywords	Cosine similarity of TF-IDF
Collaborations	Cosine similarity of TF-IDF
References	Cosine similarity of TF-IDF
Subject	Cosine similarity of TF-IDF
Year difference	Absolute difference
White	Product of estimated probabilities
Black	Product of estimated probabilities
American Indian or Alaska Native	Product of estimated probabilities
Chinese	Product of estimated probabilities
Japanese	Product of estimated probabilities
Other Asian or Pacific Islander	Product of estimated probabilities
Others	Product of estimated probabilities

corresponds to the similarity between the (full) author name fields of the two signatures, as measured using as combination operator the cosine similarity between their respective (n,m)-TF-IDF vector representations[1]. Similarly, the *Year difference* feature measures the absolute difference between the publication date of the articles to which the two signatures respectively belong.

Observing that author names from different cultures, origins or ethnic groups are likely to be disambiguated using different strategies (e.g., pairs of signatures with French author names versus pairs of signatures with Chinese author names), and as corroborated in (Treeratpituk and Giles, 2012) or (Chin et al., 2014), we propose to complement our feature set with 7 additional new features, each evaluating the degree of belonging of both signatures to several ethnic groups, as reported in the second half of Table 1. More specifically, using census data extracted from (Ruggles et al., 2008), we build a logistic regression classifier for mapping the $(1,5)$-TF-IDF representation of an author name to one of the 7 ethnic groups reported in that dataset. Given a pair of signatures (s_1, s_2), the proposed ethnicity features are each computed as the estimated probability of s_1 to belong to the corresponding ethnic group, multiplied by the estimated probability of s_2 to belong to the same group. In doing so, the expectation is for the linkage function to become sensitive to the actual origin of the authors depending on the values of these features. Indirectly, let us also note that these features hold discriminative power since, if author names are predicted to belong to different ethnic groups, then they are also likely to correspond to distinct actual persons.

[GL: Add a few examples of transformed pairs?]

Building a training set. The distinctive aspect of our work is the knowledge of more than 1 million of crowdsourced annotations $\mathcal{C}' = \{c'_0, ..., c'_{M-1}\}$, indicating together that all signature $s \in c'_i$ are known to correspond to the same individual a_i. In particular, this data can be used to generate positive pairs $(x = (s_1, s_2), y = 0)$ for all $s_1, s_2 \in c'_i$, for all i. Similarly, negative pairs $(x = (s_1, s_2), y = 1)$ can be extracted for all $s_1 \in c'_i, s_2 \in c'_j$, for all $i \neq j$.

[1] (n,m) denotes that the TF-IDF vectors are computed from character n, $n+1$, ..., m-grams. When not specified, TF-IDF vectors are otherwise computed from words.

The most straightforward approach for building a training set on which to learn a linkage function is to sample an equal number of positive and negative pairs, as suggested above. By observing that the linkage function d will eventually be used only on pairs of signatures from the same block S_b, a further refinement for building a training set is to restrict positive and negative pairs (s_1, s_2) to only those for which s_1 and s_2 belong to the same block. In doing so, the trained classifier is forced to learn intra-block discriminative patterns rather than inter-block differences. Furthermore, as noted in (Lange and Naumann, 2011), pairs of signatures are in the vast majority non-ambiguous: if both signatures share the same author names, then they correspond to the same individual, otherwise they do not. Rather than sampling pairs uniformly at random, we therefore propose to oversample difficult cases when building the training set (i.e., pairs of signatures with different author names corresponding to same individual, and pairs of signatures with identical author names but corresponding to distinct individuals) in order to improve the overall accuracy of the linkage function.

3.3 Semi-supervised clustering

The last component of our author disambiguation pipeline is clustering, that is the process of grouping together, within a block, all signatures from the same individual (and only those). As for many other works on author disambiguation, we make use of hierarchical clustering (Ward Jr, 1963) for building clusters of signatures in a bottom-up fashion. The method consists in iteratively merging together the two most similar clusters until all clusters are eventually merged together at the top of the hierarchy. Similarity between clusters is evaluated using either complete, single or average linkage, using as pseudo-distance metric the probability that s_1 and s_2 correspond to distinct authors, as calculated from the custom linkage function d.

To form flat clusters out of the hierarchy, one must decide on a maximum distance threshold above which clusters are considered to correspond to distinct authors. Let us denote by $\mathcal{S}' = \{s | s \in c', c' \in \mathcal{C}'\}$ the set of all signatures for which partial clusters are known. Let us also denote by $\widehat{\mathcal{C}}$ the predicted clusters for all signatures in $\mathcal{S}$, and by $\widehat{\mathcal{C}}' = \{\widehat{c} \cap \mathcal{S}' | \widehat{c} \in \widehat{\mathcal{C}}\}$ the predicted clusters restricted to signatures for which partial clusters are known. From these, we evaluate the following semi-supervised cut-off strategies:

- *No cut:* all signatures from the same block are assumed to be from the same author.
- *Global cut:* the threshold is chosen globally over all blocks, as the one maximizing some score $f(\mathcal{C}', \widehat{\mathcal{C}}')$.
- *Block cut:* the threshold is chosen locally at each block b, as the one maximizing some score $f(\mathcal{C}'_b, \widehat{\mathcal{C}}'_b)$. In case $\mathcal{C}'_b$ is empty, then all signatures from b are clustered together.

[GL: Add a figure showing the three possible splits]

4 Implementation

As part of this work, we developed a stand-alone application for author disambiguation, publicly available online[2] for free reuse or study. Our implementation builds upon the Python scientific stack, making use of the Scikit-Learn library (Pedregosa et al., 2011) for the supervised learning of a linkage function and of SciPy (Jones et al., 01) for clustering. All components of the disambiguation pipeline have been designed to follow the Scikit-Learn API (Buitinck et al., 2013), making them easy to maintain, understand and reuse. Our implementation is made to be efficient, exploiting parallelization when available, making it ready for production environments.

[2] `https://github.com/glouppe/beard`

5 Experiments

5.1 Data

The author disambiguation solution proposed in this work, along with its enhancements, are evaluated on data extracted from the INSPIRE portal (Gentil-Beccot et al., 2009), a digital library for scientific literature in high-energy physics. Overall, the portal holds more than 1 million publications, forming in total a set $\mathcal{S}$ of more than 10 millions signatures. Out of these, around 13% have been *claimed* by their original authors or marked as such by professional curators. Together, they constitute a trusted set $(\mathcal{S}', \mathcal{C}')$ of 15388 distinct individuals sharing 36340 unique author names spread within 1201763 signatures on 360066 publications. This data covers several decades in time and dozens of author nationalities worldwide.

Following INSPIRE terms of use, the signatures $\mathcal{S}'$ and their corresponding clusters $\mathcal{C}'$ are released online[3] under the CC0 license. To the best of our knowledge, data of this size and coverage is the first to be publicly released in the scope of author disambiguation research.

5.2 Evaluation protocol

Experiments carried out to study the impact of the proposed algorithmic components and refinements, as described in Section 3, follow a standard 3-fold cross-validation protocol, using $(\mathcal{S}', \mathcal{C}')$ as ground-truth dataset. To replicate the $|\mathcal{S}'|/|\mathcal{S}| \approx 13\%$ ratio of claimed signatures with respect to the total set of signatures, as on the INSPIRE platform, cross-validation folds are constructed by sampling a training set $\mathcal{S}'_{\text{train}} \subseteq \mathcal{S}'$ of 13% of claimed signatures, and by using the remaining signatures $\mathcal{S}'_{\text{test}} = \mathcal{S}' \setminus \mathcal{S}'_{\text{train}}$ for testing. Accordingly, $\mathcal{C}'_{\text{train}} = \{c' \cap \mathcal{S}'_{\text{train}} | c' \in \mathcal{C}'\}$ represents the partial known clusters on the training fold, while $\mathcal{C}'_{\text{test}}$ are those used for testing.

As commonly done in author disambiguation research (Levin et al., 2012), we evaluate the predicted clusters over testing data $\mathcal{C}'_{\text{test}}$, using both B3 [GL: add ref] and pairwise [GL: add ref] precision, recall and F-measure, as defined below:

$$P_{\text{B3}}(\mathcal{C}, \widehat{\mathcal{C}}, \mathcal{S}) = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \frac{|c(s) \cap \widehat{c}(s)|}{|\widehat{c}(s)|} \tag{1}$$

$$R_{\text{B3}}(\mathcal{C}, \widehat{\mathcal{C}}, \mathcal{S}) = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \frac{|c(s) \cap \widehat{c}(s)|}{|c(s)|} \tag{2}$$

$$F_{\text{B3}}(\mathcal{C}, \widehat{\mathcal{C}}, \mathcal{S}) = \frac{2P_{\text{B3}}(\mathcal{C}, \widehat{\mathcal{C}}, \mathcal{S}) R_{\text{B3}}(\mathcal{C}, \widehat{\mathcal{C}}, \mathcal{S})}{P_{\text{B3}}(\mathcal{C}, \widehat{\mathcal{C}}, \mathcal{S}) + P_{\text{B3}}(\mathcal{C}, \widehat{\mathcal{C}}, \mathcal{S})} \tag{3}$$

$$P_{\text{pairwise}}(\mathcal{C}, \widehat{\mathcal{C}}) = \frac{|p(\mathcal{C}) \cap p(\widehat{\mathcal{C}})|}{|p(\widehat{\mathcal{C}})|} \tag{4}$$

$$R_{\text{pairwise}}(\mathcal{C}, \widehat{\mathcal{C}}) = \frac{|p(\mathcal{C}) \cap p(\widehat{\mathcal{C}})|}{|p(\mathcal{C})|} \tag{5}$$

$$F_{\text{pairwise}}(\mathcal{C}, \widehat{\mathcal{C}}) = \frac{2P_{\text{pairwise}}(\mathcal{C}, \widehat{\mathcal{C}}) R_{\text{pairwise}}(\mathcal{C}, \widehat{\mathcal{C}})}{P_{\text{pairwise}}(\mathcal{C}, \widehat{\mathcal{C}}) + R_{\text{pairwise}}(\mathcal{C}, \widehat{\mathcal{C}})} \tag{6}$$

and where $c(s)$ (resp. $\widehat{c}(s)$) is the cluster $c \in \mathcal{C}$ such that $s \in c$ (resp. the cluster $\widehat{c} \in \widehat{\mathcal{C}}$ such that $s \in \widehat{c}$), and where $p(\mathcal{C}) = \cup_{c \in \mathcal{C}} \{(s_1, s_2) | s_1, s_2 \in c, s_1 \neq s_2\}$ is the set of all pairs of signatures from the same clusters in $\mathcal{C}$. Intuitively, precision evaluates if signatures are grouped only with signatures from the same true clusters, while recall measures the extent to which all signatures from the same true clusters are effectively grouped together. The F-measure is the harmonic mean between these two quantities. In the analysis below, we rely primarily on B3 measures for discussing the results, as pairwise measures tend to favor large clusters (because the number of pairs is quadratic with the cluster size), hence

[3] `https://github.com/glouppe/paper-author-disambiguation`

Table 2: Average precision, recall and f-measure scores on test folds.

	B3			Pairwise		
Description	P	R	F	P	R	F
Baseline	0.9901	0.9760	0.9830	0.9948	0.9738	0.9842
Blocking = SFI	0.9901	0.9760	0.9830	0.9948	0.9738	0.9842
Blocking = Double metaphone	0.9856	0.9827	0.9841	0.9927	0.9817	0.9871
Blocking = NYSIIS	0.9875	0.9826	0.9850	0.9936	0.9814	0.9875
Blocking = Soundex	0.9886	0.9745	0.9815	0.9935	0.9725	0.9828
No name normalization	0.9887	0.9697	0.9791	0.9931	0.9658	0.9793
Classifier = GBRT	0.9901	0.9760	0.9830	0.9948	0.9738	0.9842
Classifier = Random Forests	0.9909	0.9783	0.9846	0.9957	0.9752	0.9854
Classifier = Linear Regression	0.9749	0.9584	0.9666	0.9717	0.9569	0.9643
Training pairs = Non-blocked, uniform	0.9793	0.9630	0.9711	0.9756	0.9629	0.9692
Training pairs = Blocked, uniform	0.9854	0.9720	0.9786	0.9850	0.9707	0.9778
Training pairs = Blocked, balanced	0.9901	0.9760	0.9830	0.9948	0.9738	0.9842
Standard features	–	–	–	–	–	–
Standard + Ethnicity features	0.9901	0.9760	0.9830	0.9948	0.9738	0.9842
Clustering = Average linkage	0.9901	0.9760	0.9830	0.9948	0.9738	0.9842
Clustering = Single linkage	0.9741	0.9603	0.9671	0.9543	0.9626	0.9584
Clustering = Complete linkage	0.9862	0.9709	0.9785	0.9920	0.9688	0.9803
No cut	0.9024	0.9828	0.9409	0.8298	0.9776	0.8977
Global cut	–	–	–	–	–	–
Block cut	0.9901	0.9760	0.9830	0.9948	0.9738	0.9842

unfairly giving preference to authors with many publications. By contrast, B3 measures weight clusters linearly with respect to their size. General conclusions drawn below remain however consistent for pairwise measures.

5.3 Results and discussion

Baseline. Results presented in Table 2 are all discussed with respect to a baseline solution using the following combination of components:

- Blocking: same surname and the same first given name initial strategy (SFI);
- Linkage function: all 22 features defined in Table 1, gradient boosted regression trees as supervised learning algorithm and a training set of pairs built by balancing easy and difficult cases.
- Clustering: agglomerative clustering using average linkage and block cuts found to maximize $F_{\mathrm{B3}}(\mathcal{C}'_{\mathrm{train}}, \widehat{\mathcal{C}}'_{\mathrm{train}}, \mathcal{S}'_{\mathrm{train}})$.

Blocking. The very good precision of the baseline (0.9901) but its lower recall (0.9760) suggests that the blocking strategy might be the limiting factor to further overall improvements. As Table 3 indeed indicates, the maximum recall (i.e., if within a block, all signatures were clustered optimally) for SFI is 0.9828 , hence not letting much more room for improvement. At the price of fewer and therefore slightly larger blocks (as reported in the right column of Table 3), the proposed phonetic-based blocking strategies show better maximum recall (all around 0.9905), thereby pushing further the upper bound on the maximum overall performance of author disambiguation. As Table 2 indeed shows, switching to either Double metaphone or NYSIIS phonetic-based blocking allows to improve the overall F-measure score, trading precision for recall. In particular, the NYSIIS-based phonetic blocking shows to be the most effective when applied to the baseline (with an F-measure of 0.9850) while also being the most efficient computationally (with 10857 blocks versus 12978 for the baseline).

Finally, let us also note that Table 3 corroborates the estimation of (Torvik and Smalheiser, 2009), stating that SFI blocking has a recall around 98% on real data.

Table 3: Maximum recall of blocking strategies, and their number of blocks on $\mathcal{S}'$.

Blocking	R^*_{B3}	R^*_{pairwise}	# blocks
SFI	0.9828	0.9776	12978
Double metaphone	0.9907	0.9863	9753
NYSIIS	0.9902	0.9861	10857
Soundex	0.9906	0.9863	9403

Name normalization. As discussed before, the seemingly anodyne step of normalizing author names (stripping accents, removing affixes), as done in the baseline, reveals to be quite important. Results of Table 2 indeed clearly suggests that not normalizing significantly reduces performance (yielding an F-measure of 0.9830 when normalizing, but decreasing to 0.9791 when raw author name strings are used instead).

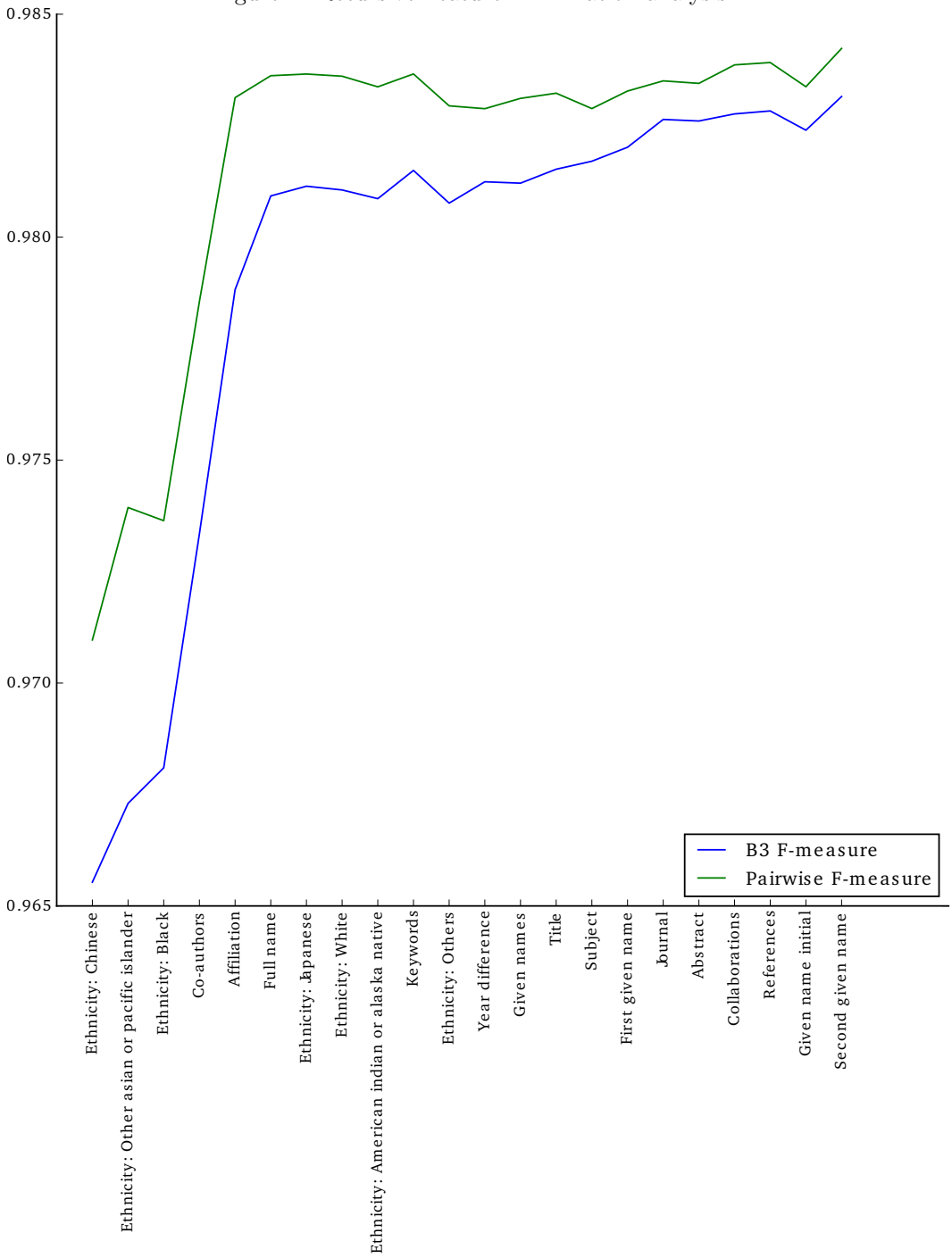
Linkage function. Let us first comment on the results regarding the supervised algorithm used to learn the linkage function. As Table 2 indicates, both tree-based algorithms appear to be significantly better fit than Linear Regression (0.9830 and 0.9846 for GBRT and Random Forests versus 0.9666 for Linear Regression). This result is consistent with (Treeratpituk and Giles, 2009) which evaluated the use Random Forests for author disambiguation, but contradicts results of (Levin et al., 2012) for which Logistic Regression appeared to be the best classifier. Provided hyper-parameters are properly tuned, the superiority of tree-based methods is in our opinion not surprising. Indeed, given the fact that the optimal linkage function is likely to be non-linear, non-parametric methods are expected to yield better results, as the experiments here confirm.

Second, properly constructing a training set of positive and negative pairs of signatures from which to learn a linkage function reveals to yield quite an improvement. A random sampling of positive and negative pairs, without taking blocking into account, significantly impacts the overall performance (0.9711). When pairs are drawn from only from blocks, performance increases (0.9786), which confirms our intuition that d should be built only from pairs it will be used to eventually cluster. Finally, making the classification problem more difficult by oversampling complex cases proves to be relevant, by further improving the disambiguation results (0.9830).

Using Recursive Feature Elimination Guyon et al. (2002), we next evaluate the usefulness of all 15 standard and 7 additional ethnicity features for learning the linkage function. The analysis consists in using the baseline algorithm first using all 22 features, to determine the least discriminative from feature importances (Louppe et al., 2013), and then re-learn the baseline using all but that one feature. That process is repeated recursively until eventually only one feature remains. Results are presented in Figure 2 for one the three folds, starting from the far right with the baseline and *Second given name* being the least important feature, and ending on the left with all features eliminated but *Chinese*. As the figure illustrates, the most important features are ethnic-based features (*Chinese*, *Other Asian*, *Black*) along with *Co-authors*, *Affiliation* and *Full name*. Adding the remaining other features only brings marginal improvements, with *Journal*, *Abstract*, *Collaborations*, *References* *Given name initial* and *Second given name* being almost insignificant. Overall, these results highlight the added value of the proposed ehtnic-based features. Their duality in modeling both the similarity between author names and their origins make them very strong predictors for author disambiguation. [GL: Add note for Table.] The results also corroborates results from (Ferreira et al., 2010), who found that the similarity between co-authors was a highly discriminative feature. It indeed appears in 4th in our ranking and brings very significant improvements. In case computational complexity is a concern, this analysis also underlines the fact that decent performance can be achieved using only a very small set of features, as also observed in (Treeratpituk and Giles, 2009) or (Levin et al., 2012).

Semi-supervised clustering. The last part of our experiments concerns the study of agglomerative clustering and the best way to find a cut-off threshold to form clusters. Results from Table 2 first clearly indicates that average linkage is significantly better than both single and complete linkage.

Figure 2: Recursive Feature Elimimation analysis.



[GL: compare threshold strategies]
[GL: comment on the fact that not clustering is not that bad]

6 Conclusions

[GL: Mention Hussein's thesis for further experiments]

References

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., Niculae, V., Prettenhofer, P., Gramfort, A., Grobler, J., Layton, R., VanderPlas, J., Joly, A., Holt, B., and Varoquaux, G. (2013). API design for machine learning software: experiences from the scikit-learn project. *CoRR*, abs/1309.0238.

Chin, W.-S., Zhuang, Y., Juan, Y.-C., Wu, F., Tung, H.-Y., Yu, T., Wang, J.-P., Chang, C.-X., Yang, C.-P., Chang, W.-C., et al. (2014). Effective string processing and matching for author disambiguation. *The Journal of Machine Learning Research*, 15(1):3037–3064.

Culotta, A., Kanani, P., Hall, R., Wick, M., and McCallum, A. (2007). 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*.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., and Lin, C.-J. (2008). Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874.

Fan, X., Wang, J., Pu, X., Zhou, L., and Lv, B. (2011). On graph-based name disambiguation. *Journal of Data and Information Quality (JDIQ)*, 2(2):10.

Fellegi, I. P. and Sunter, A. B. (1969). A theory for record linkage. *Journal of the American Statistical Association*, 64:1183–1210.

Ferreira, A. A., Gonçalves, M. A., and Laender, A. H. (2012). A brief survey of automatic methods for author name disambiguation. *Acm Sigmod Record*, 41(2):15–26.

Ferreira, A. A., Veloso, A., Gonçalves, M. A., and Laender, A. H. (2010). Effective self-training author name disambiguation in scholarly digital libraries. In *Proceedings of the 10th annual joint conference on Digital libraries*, pages 39–48. ACM.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232.

Gentil-Beccot, A., Mele, S., Holtkamp, A., O'Connell, H. B., and Brooks, T. C. (2009). Information resources in high-energy physics: Surveying the present landscape and charting the future course. *Journal of the American Society for Information Science and Technology*, 60(1):150–160.

Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine learning*, 46(1-3):389–422.

Han, H., Giles, L., Zha, H., Li, C., and Tsioutsiouliklis, K. (2004). Two supervised learning approaches for name disambiguation in author citations. In *Digital Libraries, 2004. Proceedings of the 2004 Joint ACM/IEEE Conference on*, pages 296–305. IEEE.

Huang, J., Ertekin, S., and Giles, C. L. (2006). Efficient name disambiguation for large-scale databases. In *Knowledge Discovery in Databases: PKDD 2006*, pages 536–544. Springer.

Jones, E., Oliphant, T., Peterson, P., et al. (2001–). SciPy: Open source scientific tools for Python. [Online; accessed 2015-08-10].

Kang, I.-S., Na, S.-H., Lee, S., Jung, H., Kim, P., Sung, W.-K., and Lee, J.-H. (2009). On co-authorship for author disambiguation. *Information Processing & Management*, 45(1):84–97.

Lange, D. and Naumann, F. (2011). Frequency-aware similarity measures: why arnold schwarzenegger is always a duplicate. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 243–248. ACM.

Levin, M., Krawczyk, S., Bethard, S., and Jurafsky, D. (2012). Citation-based bootstrapping for large-scale author disambiguation. *Journal of the American Society for Information Science and Technology*, 63(5):1030–1047.

Liu, W., Islamaj Doğan, R., Kim, S., Comeau, D. C., Kim, W., Yeganova, L., Lu, Z., and Wilbur, W. J. (2014). Author name disambiguation for pubmed. *Journal of the Association for Information Science and Technology*, 65(4):765–781.

Louppe, G., Wehenkel, L., Sutera, A., and Geurts, P. (2013). Understanding variable importances in forests of randomized trees. In *Advances in Neural Information Processing Systems*, pages 431–439.

Malin, B. (2005). Unsupervised name disambiguation via social network similarity. In *Workshop on link analysis, counterterrorism, and security*, volume 1401, pages 93–102.

McRae-Spencer, D. M. and Shadbolt, N. R. (2006). Also by the same author: Aktiveauthor, a citation graph approach to name disambiguation. In *Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries*, pages 53–54. ACM.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Philips, L. (2000). The double metaphone search algorithm. *C/C++ Users J.*, 18(6):38–43.

Ruggles, S., Sobek, M., Fitch, C. A., Hall, P. K., and Ronnander, C. (2008). *Integrated public use microdata series*. Historical Census Projects, Department of History, University of Minnesota.

Schulz, C., Mazloumian, A., Petersen, A. M., Penner, O., and Helbing, D. (2014). Exploiting citation networks for large-scale author name disambiguation. *EPJ Data Science*, 3(1):1–14.

Smalheiser, N. R. and Torvik, V. I. (2009). Author name disambiguation. *Annual review of information science and technology*, 43(1):1–43.

Soler, J. (2007). Separating the articles of authors with the same name. *Scientometrics*, 72(2):281–290.

Song, Y., Huang, J., Councill, I. G., Li, J., and Giles, C. L. (2007). Efficient topic-based unsupervised name disambiguation. In *Proceedings of the 7th ACM/IEEE-CS joint conference on Digital libraries*, pages 342–351. ACM.

Strotmann, A. and Zhao, D. (2012). Author name disambiguation: What difference does it make in author-based citation analysis? *Journal of the American Society for Information Science and Technology*, 63(9):1820–1833.

Taft, R. L. (1970). Name search techniques. Technical Report Special Report No. 1, New York State Identification and Intelligence System, Albany, NY.

The National Archives (2007). The soundex indexing system.

Torvik, V. I. and Smalheiser, N. R. (2009). Author name disambiguation in medline. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 3(3):11.

Tran, H. N., Huynh, T., and Do, T. (2014). Author name disambiguation by using deep neural network. In *Intelligent information and database systems*, pages 123–132. Springer.

Treeratpituk, P. and Giles, C. L. (2009). Disambiguating authors in academic publications using random forests. In *Proceedings of the 9th ACM/IEEE-CS joint conference on Digital libraries*, pages 39–48. ACM.

Treeratpituk, P. and Giles, C. L. (2012). Name-ethnicity classification and ethnicity-sensitive name matching. In *AAAI*. Citeseer.

Ward Jr, J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244.