# Ethnicity Sensitive Author Disambiguation Using Semi-supervised Learning

Gilles Louppe[1], Hussein T. Al-Natsheh[2], Mateusz Susik[3(✉)],
and Eamonn James Maguire[1]

[1] CERN, Geneva, Switzerland
{g.louppe,eamonn.james.maguire}@cern.ch
[2] Univ Lyon, ISH, USR 3385, CNRS, and ERIC Lab, EA 3083, Lyon, France
hussein.al-natsheh@ish-lyon.cnrs.fr
[3] University of Warsaw, Warszawa, Poland
msusik@student.uw.edu.pl

**Abstract.** Building on more than one million crowdsourced annotations that we publicly release, we propose a new automated disambiguation solution exploiting this data (i) to learn an accurate classifier for identifying coreferring 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. With respect to the state of the art, we validate the general pipeline used in most existing solutions, and improve by: (i) proposing new phonetic-based blocking strategies, thereby increasing recall; (ii) adding strong ethnicity-sensitive features for learning a linkage function, thereby tailoring disambiguation to non-Western author names whenever necessary; and (iii) showing the importance of balancing negative and positive examples when learning the linkage function.

## 1  Introduction

In academic digital libraries, author name disambiguation is the problem of grouping together publications written by the same person. It is often difficult because an author may use different spellings or name variants across their career (synonymy) and/or distinct authors may share the same name (polysemy). Most notably, author disambiguation is often more troublesome for 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 since 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 [28]); and/or the relevance of scientific network analysis [21]. Thus, even small improvements in the field significantly improve the usability of the digital libraries to some users.

Solutions to author disambiguation have been proposed from various communities [18]. On the one hand, libraries have maintained authorship control through manual curation, either in a centralized way by hiring professional collaborators or through developing services that invite authors to register their publications themselves (e.g., Google Scholar or Inspire-HEP). Recent efforts to create persistent digital identifiers assigned to researchers (e.g., ORCID or ResearcherID), with the objective to embed these identifiers in the submission workflow of publishers or repositories (e.g., Elsevier or arXiv), would univocally solve any disambiguation issue. As the centralized manual authorship control is expensive and the success of persistent digital identifiers requires large and ubiquitous adoption by both researchers and publishers, fully automated machine learning-based methods have been proposed to provide immediate, less costly, and satisfactory solutions to author disambiguation. In this work, we study how labeled data obtained through manual curation (either centralized or crowdsourced) can be exploited (i) to learn an accurate classifier for identifying coreferring authors, and (ii) to guide the clustering of scientific publications by distinct authors in a semi-supervised way. Our analysis of parameters and features of this large dataset reveal that the general pipeline commonly used in existing solutions is an effective approach for author disambiguation. Moreover, we propose better strategies for blocking (i.e., partitioning) based on the phonetization of author names to increase recall and ethnicity-sensitive features for learning a linkage function which tailor our author disambiguation to non-Western author names.

The remainder of this report is structured as follows. In Sect. 2, we briefly review machine learning solutions for author disambiguation. The components of our method are then defined in Sect. 3 and its implementation described in Sect. 4. Experiments are carried out in Sect. 5, where we compare approaches to the problem and explore feature choice. Finally, conclusions and future works are discussed in Sect. 6.

## 2  Related Work

As reviewed in [7,17,26], author disambiguation algorithms are usually composed of two main components: (i) a linkage function determining whether two publications have been written by the same author; and (ii) a clustering algorithm producing clusters of publications assumed to be written by the same author. Approaches can be classified 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 different authors to automatically learn a linkage function between publications [4,12,13,32,33].

Training data is usually not easily available, therefore unsupervised approaches propose the use of domain-specific, manually designed, linkage functions tailored towards author disambiguation [14,20,25,27]. These approaches

have the advantage of not requiring hand-labeled data, but generally do not perform as well as supervised approaches. To reconcile both worlds, semi-supervised methods make use of small, manually verified clusters of publications and/or 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 [8,17,31]. Semi-supervised approaches also allow for the tuning of the clustering algorithm when the latter is applied to a mixed set of labeled and unlabeled publications, e.g., by maximizing some clustering performance metric on the known clusters [17].

In this context, we position this work as a semi-supervised solution for author disambiguation, with the significant advantage of having a very large collection of more than 1 million crowdsourced annotations of publications whose true authors are identified. 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. Furthermore, by releasing our data in the public domain, we provide a benchmark on which further research on author disambiguation and related topics can be evaluated.

## 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 Fig. 1). 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 $\mathcal{P}$.



**Fig. 1.** An example 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.

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 correspond to the set of all signatures from the same individual (without knowing who). Finally, in the case of partially annotated databases as studied in this work, the set extends with the partial knowledge $\mathcal{C}' = \{c'_0, ..., c'_{M-1}\}$ of $\mathcal{C}$, such that $c'_i \subseteq c_i$, where $c'_i$ may be empty.

The distinctive aspect of our work is the knowledge of more than 1 million crowdsourced annotations, indicating together that all signature $s \in c'_i$ are known to correspond to the same individual $a_i$.

Our algorithm is composed of three parts (Fig. 2): (i) a blocking scheme whose goal is to pre-cluster signatures $\mathcal{S}$ into smaller groups; (ii) the construction of a linkage function $d$ between signatures using supervised learning; and (iii) the semi-supervised clustering of all signatures within the same block, using $d$ as a pseudo distance metric.
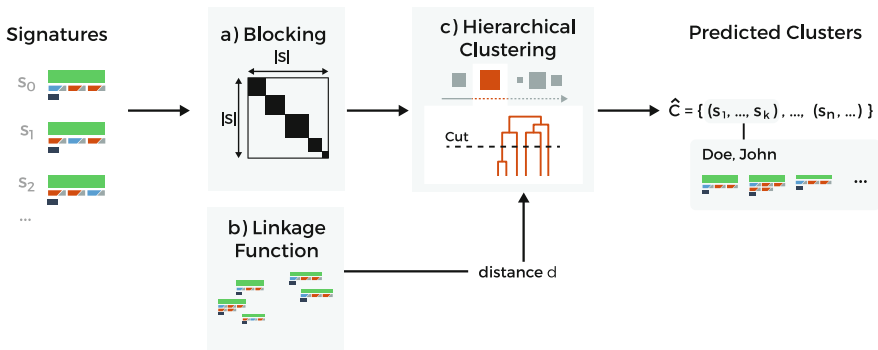


**Fig. 2.** Pipeline for author disambiguation: (a) signatures are *blocked* to reduce computational complexity, (b) a linkage function is built with supervised learning, (c) independently within each block, signatures are grouped using hierarchical agglomerative clustering.

## 3.1 Blocking

As in previous works, the first part of our algorithm consists of dividing signatures $\mathcal{S}$ into disjoint subsets $\mathcal{S}_{b_0}, ..., \mathcal{S}_{b_{K-1}}$, or *blocks* (i.e. partitions) [6], followed by carrying out author disambiguation on each one of these blocks independently. By doing so, the computational complexity of clustering (see Sect. 3.3) typically reduces from $O(|\mathcal{S}|^2)$ to $O(\sum_b |\mathcal{S}_b|^2)$. Since disambiguation is performed independently per block, a good blocking strategy should be designed such that signatures from the same author are all mapped to the same block, otherwise their correct clustering would not be possible in later stages of the workflow. As a result, blocking should be a balance between reduced complexity and maximum recall.

The simplest and most common strategy for blocking, referred to hereon in as *Surname and First Initial (SFI)*, groups signatures together if they share the same surname(s) and the same first given name initial. Despite satisfactory

performance, there are several cases where this simple strategy fails to cluster related pairs of signatures together, including:

1. There are different ways of writing an author name, or signatures contain a typo (e.g., "Mueller, R." and "Muller, R.").
2. An author has multiple surnames (or a patronymic) and some signatures place the first part of the surname within the 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 has multiple given names and they are not always all recorded (e.g., "Smith, Jack" and "Smith, A. J.")
5. An authors surname changed (e.g., due to marriage).

To account for these issues we propose instead to block signatures based on the phonetic representation of the normalized surname. Normalization involves stripping accents (e.g., "Jabłoński, L" → "Jablonski, L") and name affixes that inconsistently appear in signatures (e.g., "van der Waals, J. D." → "Waals, J. D."), while phonetization is based either on the Double Metaphone [23], the NYSIIS [29] or the Soundex [30] phonetic algorithms for mapping author names to their pronunciations. Together, these processing steps allow for grouping of most name variants of the same person in the same block with a small increase in the overall computational complexity, thereby solving case 1.

In the case of multiple surnames (cases 2 and 3), we propose to block signatures in two phases. In the first phase, all the signatures with a single 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. The biggest limitation of this method is leaving the cases 4 and 5 unhandled. This method might result in blocking together signatures from many different authors with similar names.

## 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 use during the clustering step which groups all signatures from the same author.

Formally, the goal is to build a function $d : \mathcal{S} \times \mathcal{S} \mapsto [0, 1]$, such that $d(s_1, s_2)$ approaches 0 if both signatures $s_1$ and $s_2$ belong to the same author, and 1 otherwise. This problem can be cast as a 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, [1]), Gradient Boosted Regression Trees (GBRT, [9]), and Logistic Regression [5] as classifiers.

**Input Features.** Following previous works, pairs of signatures $(s_1, s_2)$ are first transformed to vectors $v \in \mathbb{R}^p$ by building so-called similarity profiles [33] on which supervised learning is carried out. In this work, we design and evaluate fifteen standard input features [7,17] based on the comparison of signature fields, as reported in the first half of Table 1. Noteworthy, the author metadata, both provided or derived, are far more important than the publication content itself. As an illustrative example, the *Full name* feature 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].

Authors from different 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) [3,34]. For example, scientist coming from China might more/less often change affiliations, and this dependency, if learn't by the classifier, should improve the fit. To support our disambiguation algorithm, we added seven features to our feature set, with each evaluating the degree of belonging of both signatures to an ethnic group.

More specifically, using census data extracted from [24], we build a support vector machine classifier (using a linear kernel and one-versus-all classification scheme) for mapping the $(1, 5)$-TF-IDF representation of an author name to one of the ethnic groups, as defined in United States federal censuses. These groups are: *White*, *Black of African American*, *American Indian and Alaska Native*, *Asian*, *Native Hawaiian and Other Pacific Islander*, *Japanese*, *Chinese*, *Others*. Given a pair of signatures $(s_1, s_2)$, the proposed ethnicity features are each computed as the estimated probability of $s_1$ belonging to the corresponding ethnic group, multiplied by the estimated probability of $s_2$ belonging to the same group. Each of the seven races is used to create a single new feature. 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, these features also hold discriminative power since if author names are predicted to belong to different ethnic groups, then they are also likely to correspond to distinct people.

*Building a Training Set.* 1 million of crowdsourced annotations (see Sect. 3) 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 $i \neq j$.

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 [16], most signature pairs

---

[1] $(n, m) - TF\text{-}IDF$ vectors are *TF-IDF* vectors computed from $n$, $n + 1$, ..., $m$-grams.

**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 |
| Any ethnicity feature | Product of probabilities estimated by SVM |

are 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 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.
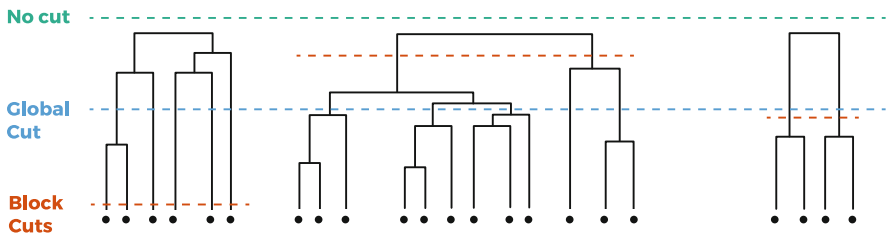
### 3.3  Semi-supervised Clustering

The last component of our author disambiguation pipeline is clustering - 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 [35] for building clusters of signatures in a bottom-up fashion. The method involves iteratively merging together the two most similar clusters until all clusters are merged together at the top of the hierarchy. Similarity between clusters is evaluated using either complete, single or average linkage, using as a pseudo-distance metric the probability that $s_1$ and $s_2$ correspond to distinct authors, as calculated from the custom linkage function $d$ from Sect. 3.2.

To form flat clusters from 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, as illustrated in Fig. 3:

– *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.



**Fig. 3.** Semi-supervised cut-off strategies to form flat clusters of signatures. Every dendogram represents a single block.

## 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 [22] for the supervised learning of a linkage function and of SciPy for clustering. All components of the disambiguation pipeline have been designed to follow the Scikit-Learn API [2], making them easy to maintain, understand and reuse. Our implementation is made to be efficient, exploiting parallelization when available, and ready for production environments. It is also designed to be runnable in an incremental fashion in which our approach is considered to be scalable. We adopt the blocking phase in order to reduce the computational complexity from $O(N^2)$ to $O(\sum N_i^2)$, which in practice tends to $O(N)$ when $N_i \ll N$. This also means that instead of having to run the disambiguation process on the whole signature set, the process could be run only on specified blocks if desired.

---

[2] https://github.com/inspirehep/beard.

## 5    Experiments

All the solutions proposed in this work are evaluated on data extracted from the *INSPIRE* portal [10], a digital library for scientific literature in high-energy physics. Overall, the portal holds more than 1 million publications $\mathcal{P}$, forming in total a set $\mathcal{S}$ of more than 10 million signatures. Out of these, around 13 % have been *claimed* by their original authors, marked as such by professional curators or automatically assigned to their true authors thanks to persistent identifiers provided by publishers or other sources. Together, they constitute a trusted set $(\mathcal{S}', \mathcal{C}')$ of 15,388 distinct individuals sharing 36,340 unique author names spread within 1,201,763 signatures on 360,066 publications. This data covers several decades in time and dozens of author nationalities worldwide.

Following the *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.

### 5.1    Evaluation Protocol

Experiments carried out to study the impact of the proposed algorithmic components and refinements, 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 13 % of claimed signatures to form a training set $\mathcal{S}'_{\mathrm{train}} \subseteq \mathcal{S}'$. The remaining signatures $\mathcal{S}'_{\mathrm{test}} = \mathcal{S}' \setminus \mathcal{S}'_{\mathrm{train}}$ are used for testing. Therefore, $\mathcal{C}'_{\mathrm{train}} = \{c' \cap \mathcal{S}'_{\mathrm{train}} | c' \in \mathcal{C}'\}$ represents the partial known clusters on the training fold, while $\mathcal{C}'_{\mathrm{test}}$ are those used for testing.

As commonly performed in author disambiguation research, we evaluate the predicted clusters over testing data $\mathcal{C}'_{\mathrm{test}}$, using both B3 and pairwise precision, recall and F-measure, as defined below:

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

$$F_{\mathrm{B3}}(\mathcal{C}, \widehat{\mathcal{C}}, \mathcal{S}) = \frac{2 P_{\mathrm{B3}}(\mathcal{C}, \widehat{\mathcal{C}}, \mathcal{S}) R_{\mathrm{B3}}(\mathcal{C}, \widehat{\mathcal{C}}, \mathcal{S})}{P_{\mathrm{B3}}(\mathcal{C}, \widehat{\mathcal{C}}, \mathcal{S}) + P_{\mathrm{B3}}(\mathcal{C}, \widehat{\mathcal{C}}, \mathcal{S})} \qquad (2)$$

$$P_{\mathrm{pairwise}}(\mathcal{C}, \widehat{\mathcal{C}}) = \frac{|p(\mathcal{C}) \cap p(\widehat{\mathcal{C}})|}{|p(\widehat{\mathcal{C}})|} \qquad R_{\mathrm{pairwise}}(\mathcal{C}, \widehat{\mathcal{C}}) = \frac{|p(\mathcal{C}) \cap p(\widehat{\mathcal{C}})|}{|p(\mathcal{C})|} \qquad (3)$$

$$F_{\mathrm{pairwise}}(\mathcal{C}, \widehat{\mathcal{C}}) = \frac{2 P_{\mathrm{pairwise}}(\mathcal{C}, \widehat{\mathcal{C}}) R_{\mathrm{pairwise}}(\mathcal{C}, \widehat{\mathcal{C}})}{P_{\mathrm{pairwise}}(\mathcal{C}, \widehat{\mathcal{C}}) + R_{\mathrm{pairwise}}(\mathcal{C}, \widehat{\mathcal{C}})} \qquad (4)$$

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

---

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

the set of all pairs of signatures from the same clusters in $\mathcal{C}$. The F-measure is the harmonic mean between these two quantities. In the analysis below, we rely primarily on the B3 F-measure for discussing results, as the pairwise variant tends to favor large clusters (because the number of pairs is quadratic with the cluster size), hence unfairly giving preference to authors with many publications. By contrast, the B3 F-measure weights clusters linearly with respect to their size. General conclusions drawn below remain however consistent for pairwise F.

## 5.2   Results and Discussion

**Baseline.** The simplest baseline against which we compare our results consists in grouping all signatures sharing the same (normalized) surname(s) and the same (normalized) first given name initial. It provides a simple and fast solution yielding decent results, as reported at the top of Table 2.

**State-of-the-Art.** Most methods proposed in related works have released neither their software, nor their data, making a fair comparison very difficult. Yet, we believe solutions reported in the literature can be closely matched to our generic pipeline, provided the blocking strategy, the linkage function and the clustering algorithm are properly aligned. In particular, we consider hereon as the *state-of-the-art* solution 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 from $(\mathcal{S}'_{\text{train}}, \mathcal{C}'_{\text{train}})$, by balancing easy and difficult cases.
– Clustering: agglomerative clustering using average linkage and block cuts found to maximize $F_{\text{B3}}(\mathcal{C}'_{\text{train}}, \widehat{\mathcal{C}}'_{\text{train}}, \mathcal{S}'_{\text{train}})$.

Below we study each component individually and discuss results with respect to the underlined state-of-the-art solution.

**Blocking Choices.** The good precision of the state-of-the-art (0.9901), but its lower recall (0.9760) suggest that the blocking strategy might be the limiting factor to further overall improvements. Our experiments showed the maximum $B3$ recall (i.e., if within a block, all signatures were clustered optimally) for SFI is 0.9828, which corroborates the estimation of this technique on real data by [31]. At the price of fewer and therefore slightly larger blocks, the proposed phonetic-based blocking strategies show better maximum recall (all around 0.9905). Better recall pushes further the upper bound on the maximum performance of author disambiguation, as the signatures that belong to the same author and different groups can not be clustered together by our algorithm. Let us remind that the reported maximum recalls for the blocking strategies using phonetization are also raised due to the better handling of multiple surnames, as described in Sect. 3.1. As Table 2 shows, switching to either Double metaphone or NYSIIS phonetic-based blocking allows to improve the overall F-measure score. In particular, the

**Table 2.** Average precision, recall and F-measure scores on test folds. Components correspond to the state-of-the-art choices.

| Description | B3 | | | Pairwise | | |
|---|---|---|---|---|---|---|
| | $P$ | $R$ | $F$ | $P$ | $R$ | $F$ |
| Baseline | 0.9024 | 0.9828 | 0.9409 | 0.8298 | 0.9776 | 0.8977 |
| 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 |
| 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** |
| 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 (baseline) | 0.9024 | 0.9828 | 0.9409 | 0.8298 | 0.9776 | 0.8977 |
| Global cut | 0.9892 | 0.9737 | 0.9814 | 0.9940 | 0.9727 | 0.9832 |
| Block cut | 0.9901 | 0.9760 | **0.9830** | 0.9948 | 0.9738 | **0.9842** |
| Combined best settings | 0.9888 | 0.9848 | **0.9868** | 0.9951 | 0.9831 | **0.9890** |
| Best settings without ethnicity features | 0.9862 | 0.9819 | 0.9841 | 0.9937 | 0.9815 | 0.9876 |

NYSIIS-based phonetic blocking shows to be the most effective when applied to the state-of-the-art (with an F-measure of 0.9850) while also being the most efficient computationally (with 10,857 blocks versus 12,978 for the baseline).

**Linkage Function Choices.** 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 [33] which evaluated the use of Random Forests for author disambiguation, but contradicts results of [17] 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 yields a significant 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 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 (see Sect. 3.2) proves to be relevant, by further improving the disambiguation results (0.9830).

Moreover, we observed that the ethnicity features serve a purpose. When these features were not included in the features set, using the best combined settings, the algorithm yields worse performance (0.9841).

Using Recursive Feature Elimination [11], we next evaluate the usefulness of all fifteen standard and seven additional ethnicity features for learning the linkage function. The analysis consists in using the state-of-the-art algorithm first using all twenty two features, to determine the least discriminative from feature importances [19], and then re-learn the state-of-the-art algorithm using all but that one feature. That process is repeated recursively until eventually only one feature remains. Results are presented in Fig. 4 for one of the three folds with the state-of-the-art, starting from the far right, *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. Overall, these results highlight the added value of the proposed ethnicity features. Their duality in modeling both the similarity between author names and their origins make them very strong predictors for author disambiguation. The results also corroborate those from [14] or [8], who found that the similarity between co-authors was a highly discriminative feature. If computational complexity is a concern, this analysis also shows how decent performance can be achieved using only a very small set of features, as also observed in [33] or [17].

**Semi-supervised Clustering Choices.** The last part of our experiment 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 indicate that average linkage is significantly better than both single and complete linkage.

Clustering together all signatures from the same block (i.e., baseline) is the least effective strategy (0.9409), but yields anyhow surprisingly decent accuracy, given the fact it requires no linkage function and no agglomerative clustering – only the blocking function is needed to group signatures. In particular, this
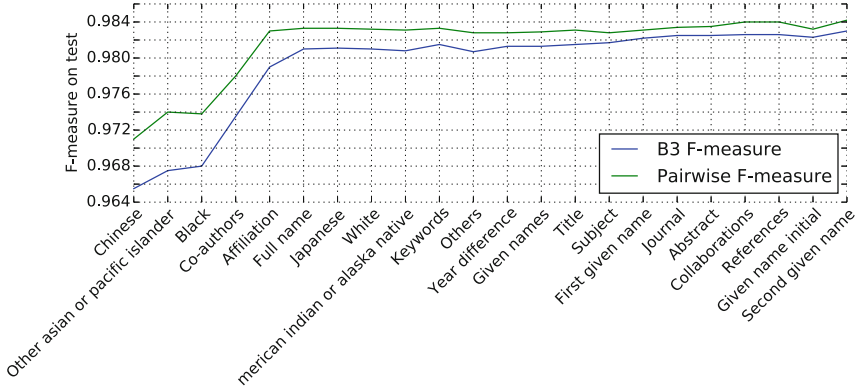
**Fig. 4.** Recursive feature elimination analysis.

result reveals that author names are not ambiguous in most cases[4] and that only a small fraction of them requires advanced disambiguation procedures. On the other hand, both global and block cut thresholding strategies give better results, with a slight advantage for the block cuts (0.9814 versus 0.9830), as expected. In case $\mathcal{S}'_b$ is empty (i.e. partial clusters are not known for any of the signatures from the block), this therefore suggests that either using a cut-off threshold learned globally from the known data would in general give results only marginally worse than if the claimed signatures had been known.

**Combined Best Settings.** When all best settings are combined (i.e., Blocking = NYSIIS, Classifier = Random Forests, Training pairs = blocked and balanced, Clustering = Average linkage, Block cuts), performance reaches 0.9862, i.e., the best of all reported results. In particular, this combination exhibits both the high recall of phonetic blocking based on the NYSIIS algorithm and the high precision of Random Forests.

**Execution Time.** Our implementation takes around 20 h to process the complete set of the data (for 10M signatures, on a 16 cores machine with 32GB of RAM). Related work [15] reports execution times around 24 h to cluster 4M signatures. Note also that shorter execution times can be achieved, at the expense of worse results, by reducing the set of the features used.

## 6    Conclusions

In this work, we have revisited and validated the general author disambiguation pipeline introduced in previous independent research work. The generic approach is composed of three components, whose design and tuning are all critical

---

[4] This holds for the data we extracted, but may in the future, with the rise of non-Western researchers, be an underestimate of the ambiguous cases.

to good performance: (i) a blocking function for pre-clustering signatures and reducing computational complexity, (ii) a linkage function for identifying signatures with coreferring authors and (iii) the agglomerative clustering of signatures. Making use of a distinctively large dataset of more than 1 million crowdsourced annotations, we experimentally study all three components and propose further improvements. With regards to blocking, we suggest to use phonetization of author names to increase recall while maintaining low computational complexity. For the linkage function, we introduce ethnicity-sensitive features for the automatic tailoring of disambiguation to non-Western author names whenever necessary. Finally, we explore semi-supervised cut-off threshold strategies for agglomerative clustering. For all three components, experiments show that our refinements all yield significantly better author disambiguation accuracy. In general, the results encourage further improvements and research. For blocking, one of the challenges is to manage signatures with inconsistent surnames or first given names (cases 4 and 5, as described in Sect. 3.1) while maintaining blocks to a tractable size. As phonetic algorithms are not yet perfect, another direction for further work is the design of better phonetization functions, tailored for author disambiguation. For the linkage function, the good results of the proposed features pave the way for further research in ethnicity-sensitivity. The automatic fitting of the pipeline to cultures and ethnic groups for which standard author disambiguation is known to be less efficient (e.g., Chinese authors with many homonyms) indeed constitutes a direction of research with great potential benefits for the concerned scientific communities. Exploring other name-to-ethnicity datasets with deeper coverage of names is another future work worth considering.

Moreover, the techniques presented in this work can be easily adapted to the broader problem of named entity disambiguation and thus might significantly improve accuracy of semantic search algorithms.

As part of this study, we also publicly release the annotated data extracted from the *INSPIRE* platform, on which our experiments are based. To the best of our knowledge, data of this size and coverage is the first to be available in author disambiguation research. By releasing the data publicly, we hope to provide the basis for further research on author disambiguation and related topics.

# References

1. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001)
2. 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., Varoquaux, G.: API design for machine learning software: experiences from the scikit-learn project. CoRR, abs/1309.0238 (2013)
3. 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.: Effective string processing and matching for author disambiguation. J. Mach. Learn. Res. **15**(1), 3037–3064 (2014)
4. Culotta, A., Kanani, P., Hall, R., Wick, M., McCallum, A.: Author disambiguation using error-driven machine learning with a ranking loss function. In: 6th International Workshop on Information Integration on the Web (IIWeb-2007), Vancouver, Canada (2007)

5. Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., Lin, C.-J.: Liblinear: a library for large linear classification. J. Mach. Learn. Res. **9**, 1871–1874 (2008)
6. Fellegi, I.P., Sunter, A.B.: A theory for record linkage. J. Am. Stat. Assoc. **64**, 1183–1210 (1969)
7. Ferreira, A.A., Gonçalves, M.A., Laender, A.H.: A brief survey of automatic methods for author name disambiguation. ACM SIGMOD Rec. **41**(2), 15–26 (2012)
8. Ferreira, A.A., Veloso, A., Gonçalves, M.A., Laender, A.H.: Effective self-training author name disambiguation in scholarly digital libraries. In: Proceedings of 10th Annual Joint Conference on Digital Libraries, pp. 39–48. ACM (2010)
9. Friedman, J.H.: Greedy function approximation: a gradient boosting machine. Ann. Stat. **29**, 1189–1232 (2001)
10. Gentil-Beccot, A., Mele, S., Holtkamp, A., O'Connell, H.B., Brooks, T.C.: Information resources in high-energy physics: Surveying the present landscape and charting the future course. J. Am. Soc. Inf. Sci. Technol. **60**(1), 150–160 (2009)
11. Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene selection for cancer classification using support vector machines. Mach. Learn. **46**(1–3), 389–422 (2002)
12. Han, H., Giles, L., Zha, H., Li, C., Tsioutsiouliklis, K.: Two supervised learning approaches for name disambiguation in author citations. In: Proceedings of 2004 Joint ACM/IEEE Conference on Digital Libraries, pp. 296–305. IEEE (2004)
13. Huang, J., Ertekin, S., Giles, C.L.: Efficient name disambiguation for large-scale databases. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) PKDD 2006. LNCS (LNAI), vol. 4213, pp. 536–544. Springer, Heidelberg (2006)
14. Kang, I.-S., Na, S.-H., Lee, S., Jung, H., Kim, P., Sung, W.-K., Lee, J.-H.: On co-authorship for author disambiguation. Inf. Process. Manag. **45**(1), 84–97 (2009)
15. Khabsa, M., Treeratpituk, P., Giles, C.L.: Large scale author name disambiguation in digital libraries. In: 2014 IEEE International Conference on Big Data (Big Data), pp. 41–42. IEEE (2014)
16. Lange, D., Naumann, F.: Frequency-aware similarity measures: why Arnold Schwarzenegger is always a duplicate. In: Proceedings of 20th ACM International Conference on Information and Knowledge Management, pp. 243–248. ACM (2011)
17. Levin, M., Krawczyk, S., Bethard, S., Jurafsky, D.: Citation-based bootstrapping for large-scale author disambiguation. J. Am. Soc. Inf. Sci. Technol. **63**(5), 1030–1047 (2012)
18. Liu, W., Islamaj Doğan, R., Kim, S., Comeau, D.C., Kim, W., Yeganova, L., Wilbur, W.J.: Author name disambiguation for pubmed. J. Assoc. Inf. Sci. Technol. **65**(4), 765–781 (2014)
19. Louppe, G., Wehenkel, L., Sutera, A., Geurts, P.: Understanding variable importances in forests of randomized trees. In: Advances in Neural Information Processing Systems, pp. 431–439 (2013)
20. Malin, B.: Unsupervised name disambiguation via social network similarity. In: Workshop on Link Analysis, Counterterrorism, and Security, vol. 1401, pp. 93–102 (2005)
21. Newman, M.E.: The structure of scientific collaboration networks. Proc. Natl. Acad. Sci. **98**(2), 404–409 (2001)
22. 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., Duchesnay, E.: Scikit-learn: machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
23. Philips, L.: The double metaphone search algorithm. C/C++ Users J. **18**(6), 38–43 (2000)

24. Ruggles, S., Sobek, M., Fitch, C.A., Hall, P.K., Ronnander, C.: Integrated Public Use Microdata Series. Historical Census Projects, Department of History, University of Minnesota (2008)
25. Schulz, C., Mazloumian, A., Petersen, A.M., Penner, O., Helbing, D.: Exploiting citation networks for large-scale author name disambiguation. EPJ Data Sci. **3**(1), 1–14 (2014)
26. Smalheiser, N.R., Torvik, V.I.: Author name disambiguation. Ann. Rev. Inf. Sci. Technol. **43**(1), 1–43 (2009)
27. Song, Y., Huang, J., Councill, I.G., Li, J., Giles, C.L.: Efficient topic-based unsupervised name disambiguation. In: Proceedings of 7th ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 342–351. ACM (2007)
28. Strotmann, A., Zhao, D.: Author name disambiguation: what difference does it make in author-based citation analysis? J. Am. Soc. Inf. Sci. Technol. **63**(9), 1820–1833 (2012)
29. Taft, R.L.: Name search techniques. Technical report Special Report No. 1, New York State Identification and Intelligence System, Albany, NY, February 1970
30. The National Archives. The soundex indexing system, May 2007
31. Torvik, V.I., Smalheiser, N.R.: Author name disambiguation in medline. ACM Trans. Knowl. Disc. Data (TKDD) **3**(3), 11 (2009)
32. Tran, H.N., Huynh, T., Do, T.: Author name disambiguation by using deep neural network. In: Nguyen, N.T., Attachoo, B., Trawiński, B., Somboonviwat, K. (eds.) ACIIDS 2014, Part I. LNCS, vol. 8397, pp. 123–132. Springer, Heidelberg (2014)
33. Treeratpituk, P., Giles, C.L.: Disambiguating authors in Academic Publications using random forests. In: Proceedings of 9th ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 39–48. ACM (2009)
34. Treeratpituk, P., Giles, C.L.: Name-ethnicity classification and ethnicity-sensitive name matching. In: AAAI, Citeseer (2012)
35. Ward Jr., J.H.: Hierarchical grouping to optimize an objective function. J. Am. Stat. Assoc. **58**(301), 236–244 (1963)