2012 10th IAPR International Workshop on Document Analysis Systems

Evaluation of Features for Author Name Disambiguation Using Linear Support Vector Machines

Piotr Jan Dendek, Łukasz Bolikowski, Michał Łukasik
Interdisciplinary Centre for Mathematical
and Computational Modelling, Univ. of Warsaw
Prosta 69, 00-838 Warsaw, Poland
Email: {p.dendek,l.bolikowski,m.lukasik}@icm.edu.pl

***Abstract*—Author name disambiguation allows to distinguish between two or more authors sharing the same name. In a previous paper, we have proposed a name disambiguation framework in which for each author name in each article we build a context consisting of classification codes, bibliographic references, co-authors, etc. Then, by pairwise comparison of contexts, we have been grouping contributions likely referring to the same people. In this paper we examine which elements of the context are most effective in author name disambiguation. We employ linear Support Vector Machines (SVM) to find the most influential features.**

***Keywords*-Name Disambiguation, Support Vector Machines, classification algorithms, supervised learning, context, bibliographies, linearity, information retrieval, computer science**

I. Introduction

A. Motivation

Author name disambiguation is an important activity (as defined by Gonçalves *et al.* [1]) in a digital library. The ability to distinguish automatically between several authors named Mark Smith is advantageous both for user interface and for bibliometrics. For example, knowing the identity of an author of a document, the author's name appearing among the document's metadata may become a hyperlink to a page listing other documents written by the same person. This knowledge can also be leveraged in analyzing author collaboration network, or identifying influential persons.

The problem of ambiguous names is far from uncommon: Torvik and Smalheiser [2] have found that about two-thirds of authors in MEDLINE cannot be unambiguously identified based solely on a surname and initials.

There were many approaches to the problem. Levin and Heuser [3] reached 78.5% accuracy in author name disambiguation by using genetic programming. Han *et al.* [4] provided a comparison between Support Vector Machines (SVM) [5] and Naive Bayes approach using a few basic features. Our goal is to explore more features and evaluate their impact on the effectiveness of author name disambiguation.

B. Framework

In our previous paper [6] we have presented a flexible, scalable author name disambiguation framework for a digital library. In the paper we have established a vocabulary briefly summarized here. A **contribution** represents a fact that a certain **document** was co-authored by a certain **person**. Therefore, each contribution refers to exactly one document and exactly one person. A single document refers to as many contributions as there are co-authors of the document. A person refers to as many contributions as there are documents co-authored by the person. The links between documents and contributions are known to us, i.e., are input to the problem. What we do not know and want to recreate are the links between contributions and persons. In other words we want to find the most likely clustering of contributions, where each cluster represents a person.

Our workflow is laid out as follows. First, for each input document we generate a list of associated contributions. Next, for each contribution we calculate a hash function of the contributor's name (for example, the hash function may lowercase a surname and filter out all diacritic marks). The preceding actions correspond to the "map" phase of the map-reduce paradigm. All the contributions with the same value of the hash are referred to as a **shard**. In the "reduce" phase, all the contributions within a shard are clustered into persons. In order to do that, for each pair of contributions within a shard, a number of **features** are calculated. Having sufficient information, a feature is a function returning a positive value if the two contributions are likely to refer to the same person, and a negative value otherwise. In case of lacking information for either of the inspected contributions, e.g. absence of e-mail address, a feature outcome is a `null` value. Next, a weighted sum of all the feature values is calculated. The higher the sum, the more likely it is that the two contributions are made by the same person. Having calculated all the sums, we run a clustering algorithm which groups together contributions belonging to the same persons.

To summarize, we have proposed a workflow compatible with the map-reduce paradigm. Three methods in the workflow are customizable: the hash function, the set of feature functions with associated weights, and the clustering algorithm.

978-0-7695-4661-2/12 $26.00 © 2012 IEEE
DOI 10.1109/DAS.2012.36

C. Evaluation plan

In this paper we fix a hash function and a clustering algorithm, define a number of feature functions, and calculate feature weights yielding the highest quality of results. We analyze the resulting weights in order to find the most influential feature functions. Results of this effort can be used to fine-tune the name disambiguation framework. Also, the results may be insightful to other researchers constructing author name disambiguation solutions.

II. Features

Recall that a **feature** is a function which takes two contributions and returns a real value from the set $[-1, 1]$ indicating whether the contributions are likely to refer to the same person. A feature typically refers to a single aspect of the contributions' contexts, such as year of publication or list of keyword phrases. We divide feature building into two stages. First, let us propose a set of **crude features**: a feature-like functions, which return arbitrarily large integer values. Next, in order to produce features from crude features, in each case we need to decide on a mapping from the set of integers to the set $[-1, 1]$. We have investigated the following crude features:

1) **CS** – number of common co-author surnames. Example: one contribution by a M. Smith was co-authored by H. Black and A. Johnson and T. White, while another contribution by a M. Smith was co-authored by R. Brown and A. Johnson and F. White. The two contributions have two co-author surnames in common (Johnson and White), therefore CS for these contributions will yield 2.
2) **EM** – number of common co-author e-mails. Similar to CS, but this time common e-mails are counted.
3) **EL** – number of common e-mail local parts in co-author e-mails. For example: co-authors of the first contribution have the following e-mails: `ljb@abc.edu`, `tsmith@def.edu`, while co-authors of the second contribution have: `cjones@abc.edu`, `ljb@ghi.edu`. There is one e-mail local part in common (`ljb`), thus EL for these contributions will yield 1.
4) **CC** – number of common classification codes. Due to the nature of our test set, we have compared Mathematics Subject Classification (MSC) codes. Each MSC code is an alphanumerical tag representing a subject mentioned in an article, e.g. the code 14A25 represents "Elementary questions in the algebraic geometry", where "15" represents "Algebraic geometry", "A" corresponds to "Foundations" and "25" symbolizes "Elementary questions". In this article we compare complete 5-character codes.
5) **KP** – number of common keyword phrases on the list of keywords. A keyword phrase (usually comma- or semicolon-delimited) may contain more than one word, for example: "name disambiguation".
6) **KW** – number of common words on the list of keywords. Here, unlike in KP, each word is counted separately.
7) **RF** – number of common bibliographic references.
8) **YR** – number of years between publications of the two documents.
9) **CI** – one if one of the documents references the other, zero otherwise.
10) **IS** – one if ISSNs of the two contributions match, zero otherwise.

Recall that in our framework the features are calculated within shards. Since each shard contains contributions with the same surname (up to diacritics), there was no need to consider a "same contributor surname" crude feature.

Ultimately, we were interested in scaling the crude features to the $\{-1, 1\}$ set. In order to achieve that, in each case we have chosen a positive/negative threshold which had maximized accuracy and sensitivity of a mapping. Most of the values attained by the crude features were 0, 1, or 2, and the thresholds were either 0 or 1.

III. Feature Selection

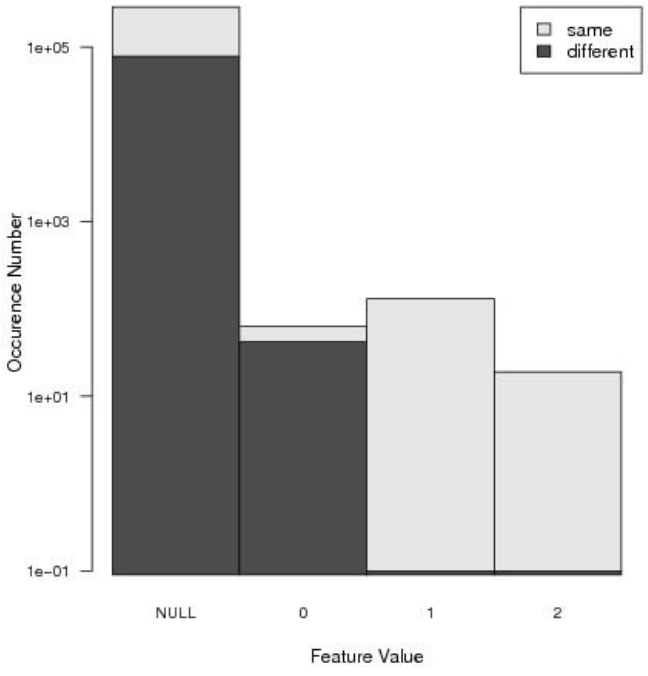

Figure 1. Distribution of values of the crude feature EM (common author e-mails), broke down into contribution pairs originating from same persons and contribution pairs originating from different persons.

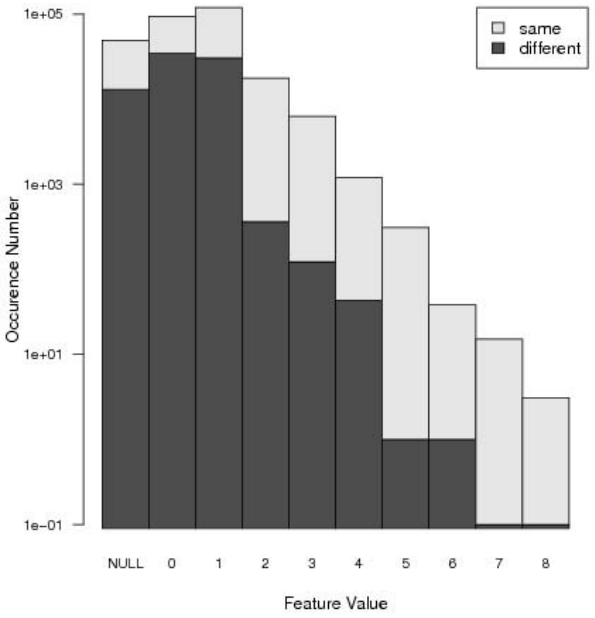

Figure 2. Distribution of values of the crude feature CS (common co-author surnames), broke down into contribution pairs originating from same persons and contribution pairs originating from different persons.

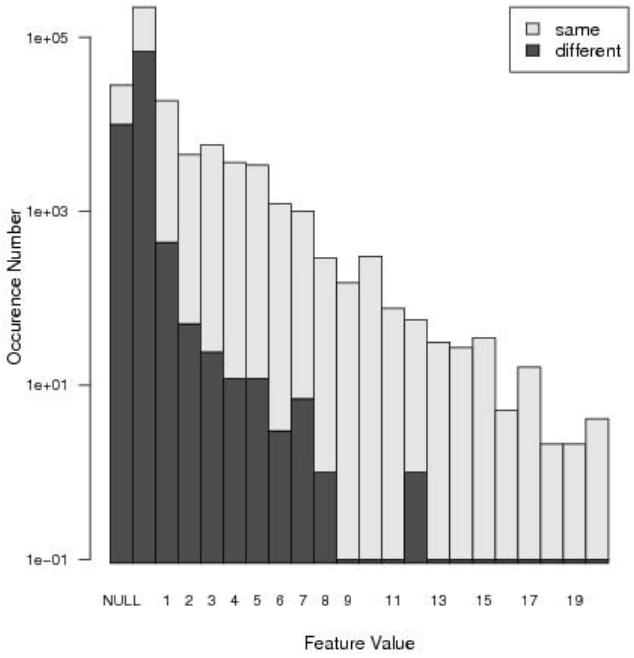

Figure 3. Distribution of values of the crude feature KP (common keyword phrases), broke down into contribution pairs originating from same persons and contribution pairs originating from different persons.

In the further phases only those features could be used, which returned few `null` values. This forced us to discard a couple of good features, most notably EM (common co-author emails) and EL (common e-mail local parts in co-author e-mails), since less than one in a thousand contribution pairs had e-mail addresses listed in both documents. This was unfortunate, since the e-mail-based features, when present, were the most effective. One or more common e-mail address meant, in 100% cases, that the two contributions originated from the same person (cf. Figure 1). Similarly, EL was almost as good an indicator, reaching true positive value of 96%.

We also had to drop RF (common bibliographic references) and YR (number of years between publications of the two documents) crude features, which occurred in 1% of all the contribution pairs. Again, RF was a good indicator when present: 9 or more common bibliographic references gives certainty that the contributions belonged to the same person. See Table I for detailed statistics of frequency of `null` values, as well as Figures 2 and 3 for distributions of values of CS and KP respectively.

Ultimately, the following features remained:

- number of common co-author surnames (CS)
- number of common classification codes (CC)
- number of common keyword phrases on the list of keywords (KP)
- number of common words on the list of keywords (KW)
- matching ISSNs (IS)

Table I
FREQUENCY OF `null` VALUES FOR CRUDE FEATURES.

Feature	Non-`null` values		`null` values	
	Number	Percent	Number	Percent
EM	213	0.07	288796	99.93
EL	213	0.07	288796	99.93
CS	239753	82.95	49256	17.04
CC	289009	100.00	0	0.00
KP	260879	90.26	28130	9.73
KW	260879	90.26	28130	9.73
RF	1556	0.53	287453	99.46
YR	4744	1.64	284265	98.36
CI	289009	100.00	0	0.00
IS	265861	91.99	23148	8.01

IV. DATA PREPARATION

We have been granted access to Zentralblatt MATH authority file, which we combined with metadata from the European Digital Mathematics Library. From this we have generated a set of contributions for further processing.

As it was stated in the previous section, features are computed for pairs of contributions. Each contributor in Zentralblatt MATH database has a unique personality identifier assigned, hence by taking a pair of contributions we can check if two contributions belong to the same person. In this manner we switch from the problem of clustering to the one of classification, where each record (or observation) represents a distinct pair of contributions, containing a vector of feature values and the final classification: "same" or "different". This approach of mapping clustering problem to classification was used by Wang *et al.* [7] as a measure (called salient degree) of a successful clustering. To reduce the number of created observations, we create them only from contributions from the same shard. We checked on our test set that the problem of contributions belonging to the same person, but having different surnames affects only 1 shard in a thousand (given the total number of about 3000 shards this problem is marginal). Hence we may safely assume that all the contributions of a single person are put in the same shard.

We have prepared a set of almost 300 thousand observations. After choosing only the records in which all the fields were non-`null` we have got approx. 200 thousand observation. Since "same" observations outnumbered "different" ("same" accounted for nearly 75% of observations), which was highly undesirable, because it could affect the final classification. Because of this, we have randomly chosen every third "same" observation thus creating the final, balanced set of observations.

To sum up, we propose the following mapping from our problem to the classification one:

- each pair of contributors is being treated as an object
- the feature vector consists of values of the features described above
- the label equals to -1 if the contributions in the pair belong to different persons, and 1 if to the same person

V. Weight Assignment

In this chapter we describe our approach to the problem of automatically assigning weights to the features. Basing on the concept of mapping to the classification problem, we explain the restriction to linear classifiers and choose linear SVM to choose weight values.

If we think about the problem of discriminating the pairs of authors that are the same person from the pairs that aren't, we can use one of many classification algorithms that have been designed. However, if we want to have weights assigned to the features, our options are narrowed. We are interested in getting a classifier of the form:

$$f(x_1, x_2, \ldots, x_n) = sign(\sum_{i=1}^{n} a_i\, x_i + bias)$$

Such a function corresponds to the labels we described in the mapping.

Table II
Feature Weights

Feature number	Weight Assigned			
	1-fold	2-fold	3-fold	Average
CS	$3.411\dot{0}^{-13}$	$-4.991\dot{0}^{-05}$	$1.131\dot{0}^{-13}$	$-1.661\dot{0}^{-05}$
CC	0.99	0.99	0.99	0.99
KP	1.00	0.99	0.99	0.99
KW	$3.551\dot{0}^{-14}$	$4.291\dot{0}^{-05}$	$6.781\dot{0}^{-05}$	$3.691\dot{0}^{-05}$
IS	$1.421\dot{0}^{-13}$	$-2.941\dot{0}^{-05}$	$4.061\dot{0}^{-05}$	$3.751\dot{0}^{-06}$

In the function, a_i elements are the weights we search for. Bias appears, so that it may consider hyperplanes that do not contain the vector $\mathbf{0} = (0_1, 0_2, \ldots, 0_n)$. We can think of the bias as the weight of the 0-th feature, whose value is always 1 (for each pair of contributors).

SVM is a classifier that finds a hyperplane separating vectors belonging to different classes. It tries to maximize the margin between the hyperplane and the vectors closest to the separating plane. Therefore, SVM returns parameters to the function as described. It is an efficient algorithm when considering the restriction to linear classifiers (when considering linear kernel as in our approach).

VI. Evaluation

We have built a classifier using the features described earlier, it turned out they have an average error of 20.54% on 3-fold cross-validation, and bias equal to 1.

As we have remarked earlier, since we compare contributions within a single shard, there is an implicit feature "same contributor surnames". Table II presents feature weights (parameters of the best hyperplanes). It turns out that the most influential features are common classification codes and common keyword phrases. This result signifies that if one uses features examined in this article, then documents from any given branch of science will most likely be attributed to the same person, but identities of authors having several radically different scientific interests may not be correctly recovered. Generally, the lower is the threshold of needed similar codes in CC Feature, the more connections to the same personalities can be obtained. Assuming it is common for authors to gradually change a topic of interest, another, more coarse-grained approach is additional top-down inspection of all three parts of classification code. It has to be stressed that the purpose of automatic author name disambiguation is to relieve an expert from a tedious, manual process. It is not the ambition of the authors to propose an approach that would outperform a human expert. Therefore, if an author is active in two substantially different areas and there are no clues linking the two "identities", then neither a human expert nor our automated approach will be able to correctly restore the author's identity based on available metadata.

In order to explain the rejection of common co-author surnames let us note that neither does a common co-author

surname entail that a pair of contributions belong to the same person, nor does lack of common co-author surname entail the opposite.

We performed a number of "sanity checks" in order to check how stable the results are. The following has been done:

1) what happens when we set to 0 weights that are already close to 0: we observed that the error doesn't change.
2) what happens when all the features are randomly changed by a value from a specific interval: the error for such a new table of objects doesn't change until the noise is substantial (e.g. drawn uniformly from $(-1, 1)$)
3) how the weights change when all the features are randomly changed by a value from a specific interval: again, the error has to be taken from the interval close to $(-1, 1)$ in terms of its size, in order to change the weights substantially (e.g. the 2 features have values close to 1 and others close to 0 until we don't introduce such a big noise)

VII. Future works

A. Better hash function

Given Zentralblatt MATH authority file, we may look closer at the relation between surnames and person identities. Such observations may lead us to a better hash function for the framework. It would be especially desired to cope with ambiguous phonetic surname notation.

B. New features

Thanks to combining EuDML dataset (containing e-mail information) and Zentralblatt MATH dataset (containing unique person identifiers), we were able to prove importance of some features. It may be beneficial to merge more datasets from EuDML and other sources, which would give opportunity to construct and check more features.

C. Using `null` values

Our next approach should be able to cope with frequent appearance of `null` values. Possible, but less hoped solution would be indirect, based on accuracy/sensitivity comparison between major-null features and those proceeded by the module described in this article. Better solution would use some preprocessing to deal with null values.

VIII. Summary

In this article we have proposed a weight assignment process based on linear SVM. Thanks to information from Zentralblatt MATH authority file we were able to switch from a clustering problem to a classification problem. This gave us an opportunity to use optimal, well-tested methods, thanks to which we have obtained a set of weights for the chosen set of features.

Acknowledgment

The work is supported by the National Centre for Research and Development (NCBiR) under Grant No. SP/I/1/77065/10 by the Strategic scientific research and experimental development program: "Interdisciplinary System for Interactive Scientific and Scientific-Technical Information".

The authors would like to thank Zentralblatt MATH for providing their authority file for the purpose training and evaluation of our name disambiguation module.

We would also like to thank the anonymous reviewers for their insightful comments.

References

[1] M. A. Gonçalves, E. A. Fox, L. T. Watson, and N. A. Kipp, "Societies (5S): A Formal Model for Digital Libraries," *ACM Transactions on Information Systems*, vol. 22, no. 2, pp. 270–312, 2004.

[2] V. I. Torvik and N. R. Smalheiser, "Author name disambiguation in MEDLINE," *ACM Transactions on Knowledge Discovery from Data*, vol. 3, no. 3, pp. 1–29, Jul. 2009. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1552303.1552304

[3] F. H. Levin and C. A. Heuser, "Using Genetic Programming to Evaluate the Impact of Social Network Analysis in Author Name Disambiguation," in *Proceedings of the 4th Alberto Mendelzon International Workshop on Foundations of Data Management Buenos Aires Argentina May 1720 2010*, ser. CEUR Workshop Proceedings, A. H. F. Laender and L. V. S. Lakshmanan, Eds., vol. 619. Citeseer, 2010. [Online]. Available: http://ceur-ws.org/Vol-619/paper2.pdf

[4] H. Han, L. Giles, H. Zha, C. Li, and K. Tsioutsiouliklis, "Two supervised learning approaches for name disambiguation in author citations," *Proceedings of the 2004 joint ACM/IEEE conference on Digital libraries - JCDL '04*, p. 296, 2004. [Online]. Available: http://portal.acm.org/citation.cfm?doid=996350.996419

[5] V. N. Vapnik, *The Nature of Statistical Learning Theory*, ser. Statistics for Engineering and Information Science. Springer, 1995, vol. 8, no. 6. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/18255760

[6] L. Bolikowski and P. J. Dendek, "Towards a Flexible Author Name Disambiguation Framework," in *Towards a Digital Mathematics Library*, P. Sojka and T. Bouche, Eds. Masaryk University Press, 2011, pp. 27–37.

[7] J. Wang, S. Wu, H. Vu, and G. Li, "Text document clustering with metric learning," in *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2010, pp. 783–784. [Online]. Available: http://59.108.48.12/proceedings/sigir/sigir2010/docs/p783.pdf