Feature Engineering for Entity Resolution with Arabic Names: Improving Estimates of Observed Casualties in the Syrian Civil War

Niccolò Dalmasso, Robin Mejia
Department of Statistics & Data Science
Carnegie Mellon University
Pittsburgh, PA
{ndalmass, rmejia}@andrew.cmu.edu

Jordan Rodu
Department of Statistics
University of Virginia
Charlottesville, VA
jsr6q@virginia.edu

Megan Price
HRDAG
San Francisco, CA
meganp@hrdag.org

Jared Murray
Department of Statistics & Data Science and
Department of Information, Risk, & Operations Management
McCombs School of Business, University of Texas at Austin
Austin, TX
Jared.Murray@mccombs.utexas.edu

Abstract

Entity resolution or record linkage is the task of identifying records referring to the same entity across multiple data sources. In the absence of a unique identifier entities must be resolved on the basis of possibly noisy and incomplete quasi-identifiers, such as names, ages, and addresses or geographic locations. Our goal is to improve estimates of the total observed casualty count in the ongoing Syrian civil war. Estimating the total victim tools in a conflict is an important element to understand its extend and magnitude, drive intervention policies and also to aid in bringing justice to perpetrators and mass murderers. Our data comprise multiple lists of casualties, compiled by the Human Rights Data Analysis Group. To arrive at an estimate of the number of unique casualties we first need to detect duplicate entries within and across lists. By focusing on Arabic names and their structure, we develop new features for comparing records and demonstrate meaningful improvements over existing classifiers (which have already seen significant engineering), empirically supporting the importance of language-specific analysis. We expect that these features will be useful in other contexts where it is necessary to measure the similarity between Arabic names.

1 Introduction

In early 2011, during the Arab Spring, protests against Syrian President President Bashar al-Assad quickly devolved into armed conflict. Since 2012 the Human Rights Data Analysis Group (HRDAG) has been working to provide an accurate count of the number of casualties incurred during the conflict. HRDAG accesses multiple lists of casualties compiled by different groups with the goal of resolving them into a list of unique casualties. As there is significant overlap between casualties reported by each group and no unique identifier for each individual (e.g., a social security number) entity resolution is necessary to de-duplicate these lists. Entity resolution proceeds in a few distinct stages, summarized in Figure 1: Since it is infeasible to compare all possible pairs of records, the first step is

1st Workshop on Artificial Intelligence for Humanitarian Assistance and Disaster Response (NeurIPS 2019), Vancouver, Canada.

to reduce the number of candidate pairs via *indexing* or *blocking* rules [1, 2]. In the second stage, record pairs that satisfy the blocking rules are assigned match probabilities or scores by a classifier. If such score is above a certain threshold the pair will be deemed a "match" (positive pairs) or, if not, a "non-match" (negative pairs). Once this step is completed, we can estimate the total victim tool by first clustering the records referring to the same individual and then using capture-recapture techniques [3] such as multiple system estimations (MSE, see e.g. [4]).

The role of estimating the total number of casualties plays an important part in humanitarian assistance in conflicts. Quoting Patrick Ball from HRDAG *"This is far more than an abstract intellectual exercise. Developing reliable statistics about mass violence is the first step in bringing justice to those guilty of mass violence."*[5]. For instance, the work in estimating total number of victims killed or disappeared between 1980 and 2000 in Peru has brought the attention of the political discussions to the reality of abuse in rural areas [6]. Similar work in Guatemala, started after a 36 years armed conflict ended in 1996, culminated in results which were brought to court in the trials of general José Efraín Ríos Monnt [7]. Statistical analysis for the conflict in Kosovo in 1999 pointed to a potential state-coordinated ethnical cleanse in the region at the time, and were used in the trail of the former Jugoslav president Slobodav Miloševíc [8]. In addition, investigating whether particular subgroups are dis-proportionally targeted in a conflict is also feasible leveraging further information about the victims. For instance, during the trial against the former Chad dictator Hissène Habré, analysis indicated a mortality rate among political prisoners to be 90 to 540 times higher than adult men in Chad during the time of the conflict [9].

The goal of our work is to engineer features from Arabic names in order to improve performance of a probabilistic classifier for entity resolution. The name represents the essential unit of a victim record and has arguably the most structure among other information available (e.g. age, gender, location and date of death). We provide insights on the Arabic naming structure and develop Arabic grammatical standardization rules (Section 2). In Sections 3 and 4 we provide features which shows empirical improvements and provide evidences of the value of language-specific work. Our code is publicly available on `Github`[1].

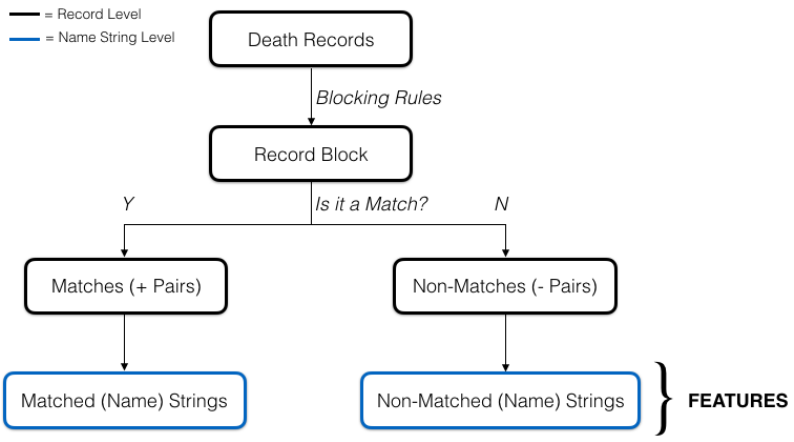

Figure 1: Diagram for matches and non-matches generation and terminology throughout the paper.

1.1 Data

The data are composed of $\sim 457,000$ death records, of which $\sim 122,000$ record pairs have been manually identified as matching by an expert for HRDAG. We assume such hand-labelled matches as ground truth. Blocking rules from [10] (listed in Supplementary Material A) were used in order to generate non-matches from the available death records. The data were made available in two batches, a larger one ($\sim 437,000$ records) spanning from March 2011 to August 2013 and a smaller one covering August 2013 to mid-2014 ($\sim 20,000$ records). Earlier records were used for training and testing (with a 2/3-1/3 split), while records from the last period were held out as a validation set.

[1] `github.com/Mr8ND/record-linkage-syrian`

1.2 Related Work

As noted above, probabilistic record linkage comprises several steps; see [2, 11] for reviews. Our work is most directly related to the linkage and analysis performed to estimate casualty counts in the Syrian conflict done by HRDAG [10, 12, 13, 14], which motivated our focus on Arabic names. [15] and [16] give a broad overview on the structure of Arabic names, with [17] providing a compendium of Arabic names useful for specific grammatical cases. [18] and [19] were pivotal in understanding the grammatical case of names in Arabic and the different possible alphabets for Arabic respectively. Regarding entity resolution in Arabic, [20, 21, 22] discuss different algorithm implementations and techniques applied to the Arabic language in general. Our work incorporates some of these developments, such as Arabic Soundex and n-grams, combining them with grammatical standardization techniques and data-driven features guided by the insights gleaned from records in our training set.

2 Arabic Name Structure Analysis

As Arabic names do not follow the traditional Western structure of "name-surname", we begin by describing the traditional structure of Arabic names and our development an automated parser (Section 2.1). We then consider different grammatical rules to account for cases in which the same name could be written in different characters, developing three standardization rules (Section 2.2).

2.1 Name Structure

An Arabic name may contain the following parts ([15], [16]):

1. *ism* (مسا): represents the person's proper name and it is the equivalent of the name in western languages. The words used in this case are usually related to religion and/or used also in regular language as adjectives and nouns[2];
2. *nasab* (بسذ): It represents a patronymic and it is a formal part of the name, usually appearing after the *ism* and/or *laqab* (the latter is defined below). It is used with the prefix "bin"/"ibn"/"ben" which translate as "son" and "bint" which translates as "daughter";
3. *kunya* (بسذ): Not given at birth, it technically represents a teknonym, i.e. a name referring to the child of the person in consideration.
4. *laqab* (بقذ: Not given at birth, it is an informal name describing a particular quality of the person - i.e. it is used for a description of the person;
5. *nisbah* (ةبسذ): Given at birth but less common than *ism* and *nasab* , this indicates either the profession or the geographical place and nation of origin of the person.

An Arabic name usually comprises one *ism* and by one or two *nasab* - the equivalent of a name and surname(s) in Latin or Anglo-Saxon languages. An example for that could be نبنامدس نب دمحم نمأ, which would translate as *Mohammed ibn Salman ibn Ameen*, *Muhammad* son of *Salman* son of *Ameen*, where *Salman* was the name of the father of this person and *Ameen* the name of the grandfather. Given the structure above, we implemented a novel tokenization method for Arabic names through the use of an automatic parser (see Supplementary Material B for more information).

2.2 Standardizing Arabic Names for Matching

We consider several standardization rules to adjust for minor name differences that are unlikely to be meaningful. These include slightly different spellings of the same name, different accents dependent on local dialects, and different grammatical cases. We defined three rules for standardizing names

Rule 1: Accent and Special Character Removal
Sometimes name strings can be written in characters that are different from the standard Arabic ones. This might be due to regional inflections - e.g. Arabic-Indic characters - or the use of older characters and annotations - e.g. older accents from the Quran. These are called "presentation forms" in Arabic, and can be converted to standard Arabic characters

[2] "Mohammed" means "praiseworthy", "Ali" means "exalted", for instance.

leveraging Unicode encoding as listed in [19]. Local dialects and annotations have different norms or rules for using accents, and can be sensitive to transcription errors (an extensive treatment can be found in [20]).

Rule 2: Removal of the "al" Particle
As shown in [23], the two letters ال – corresponding to the characters "al", translated in English as *the* – appear commonly in Arabic names. However, their presence or absence alone does not actually change the name, so we remove this particle from names before comparing them.

Rule 3: Removal of Prefixes and Suffixes by Stemming
Stripping the "al" particle is a special case of *stemming*, the removal of prefixes and suffixes from words. Stemming may also be useful for other cases in which the same name is written almost in the same exact way, apart from some characters at the beginning or at the end of the name. In Arabic this occurs due to the use of the plural rather than singular case, feminine rather than masculine form, the use of personal pronouns related to the noun in consideration or different grammatical cases. To add on the latter, in Arabic personal nouns are in fact usually "diptotes", i.e. they can have two different grammatical cases which change their suffix. We use the stemmer in the `Natural Language Toolkit` (NLTK) in Python, based on in [18] for Arabic stemming.

If we are to apply any name standardization at all, the removal of accents and special characters (Rule 1) is the most obvious place to start considering observations in [20]. Since stemming (Rule 3) removes *all* prefixes and suffixes, including the "al" particle from Rule 2, while also accounting for accents as well, we need only consider three combinations of these rules – Rule 1, Rule 1& 2, and Rules 1,2 & 3.

3 Feature Construction

Our goal is to identify features that can differentiate between matches and non-matches well. In the ideal case, the distribution of a feature within matches would not overlap the distribution of that feature within non-matches, providing perfect discrimination. We use the amount of separation as an exploratory tool to screen for promising features; once some degree of discrimination is shown, we rely on the classifier to perform feature selection and combination to optimize classification performance among all such features. For binary features we examine the difference in incidence between matches and non-matches. For continuous features we use distance metrics between the statistical distributions of matches and non-matches. We identified three sets of features that seem to exhibit good separation (Supplementary Material C provides more details about each set):

- **Binary Features**
  These include features indicating whether two names match *exactly*, match in their *soundex* representations, match after *removing spaces* between words, match when *ignoring the order* of words, or whether one name is *contained* within the other.
- **String-level Features**
  We consider the number of *aligned characters and words* between the two names - i.e. characters/words appearing in the exact same positions in both names. We also consider whether there is *exact agreement when sorting alphabetically* at a character level. We also included the percentage of characters in the same position in *prefixes*, *suffixes*, *ism* and *laqab /nisbah* .
- **N-Gram Features**
  We also generated features based on character n-grams, given their successful applications in other NLP problems (e.g. [24]). We included the *percentage of common n-grams from* $n = 2$ *to* $n = 8$ – as more than 99% of the words in names are made of 8 characters or less – both considering and ignoring their order of appearance in the name.

4 Results

We had three main questions when evaluating our results: (a) Are we improving over the baseline performance of existing HRDAG methods? (b) Are the Arabic standardization rules useful for

improving the discrimination ability of our features? (c) And finally, is the extra work involved in deriving language-specific standardization rules and features useful, or can similar performance be realized using language-agnostic features?

Following HRDAG [10], we use the F1 score as a summary measure of performance and gradient boosted trees (XGBoost, [25]) as our classifier, which exhibited the good empirical results in similar datasets. Due to the relatively severe class imbalance (non-matches represent around 94% of record pairs in the training set) we subsampled the number of non-matches to be the same as the number of matches when training XGBoost. Hyper-parameters to control over-fitting (maximum depth of trees, minimum node child weight and γ parameter) were selected using the test set (results below). All figures report the difference in F1 score with respect to the performance achieved by the baseline model from HRDAG over the same dataset; the test set was bootstrapped 25 times to assess uncertainty in the final estimate.

4.1 (a) Engineered features yield positive improvements over existing methods

As significant effort has already gone into engineering from HRDAG side, our first goal was to determine whether our features provide a further bump in performance. (Results achieved by HRDAG are already strong, with an F1 score of 94.7%.) Figure 2 (left) shows positive improvements in F1 score when adding our engineered features. The increase is small in magnitude, but it translates to a large number when considering the scale of the Syrian conflict. In the training dataset, the increase achieved when applying all the features engineered corresponds to ~ 130 more true positives and ~ 120 more true negatives correctly identified by the classifier.

4.2 (b) Standardization rules provide a significant contribution

Computing each of the features in Section 3 after applying the three different sets of standardization rules in Section 2.2 increases the total number of features by a factor of three. Figure 2 (Right) shows that Arabic rules achieve a positive increase in performance when compared to the same groups of features without any standardization, thus providing empirical evidence for the benefits of standardization. The cost of increasing the dimension of the classification problem by including the three different standardization rules is apparently offset by better performance.

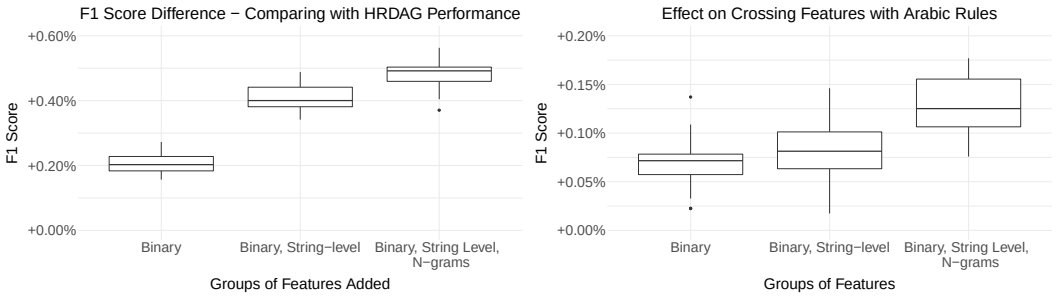

Figure 2: Left: F1 difference when adding our engineered features and Arabic standardization rule. Adding more features has a positive effect, statistically significant with respect to HRDAG features and across groups. Right: F1 difference when including Arabic rules in the classification process versus not using any standardization. Using Arabic rules has a significantly positive effect on the F1 score in classification.

4.3 (c) Language-specific features improve on language-agnostic features

Finally, we assessed whether adding language-specific features to HRDAG features would improve results further than adding only language-agnostic ones. As proxies, we consider as language-specific features the set of all our engineered features crossed with different standardization rules, while we use n-grams without any standardization as language-agnostic features. Figure 3 compares the relative increase in performance with respect to using only HRDAG features, providing evidences of a further significant improvement in performance when using language-specific features.

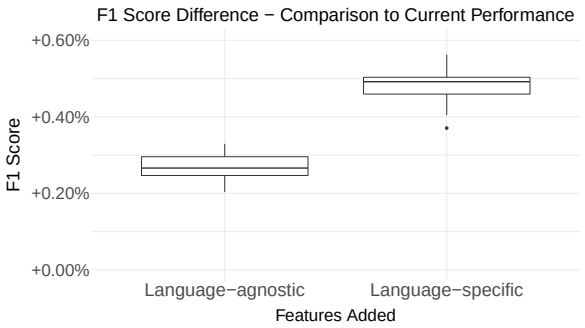

Figure 3: Relative difference in F1 score with respect to using HRDAG features only, when adding language-agnostic features versus when adding language-specific features, i.e. all engineered features.

5 Conclusions

In this paper we implemented a series of features based on classical string similarity metrics and Arabic grammatical structures within the framework of providing entity resolution for Arabic names, which resulted into a positive lift in performance in entity resolution. We expect these features to be relevant when it is necessary to measure similarities between Arabic names.

References

[1] Jared S. Murray. Probabilistic record linkage and deduplication after indexing, blocking, and filtering. *Journal of Privacy and Confidentiality*, 7(1), 2016.

[2] Peter Christen. *Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection*. Springer, 2012.

[3] C.J Peterson. The yearly immigration of young plaice into the limfjord from the german sea. *Report of the Danish Biological Station(1895)*, 6:5–84, 1896.

[4] Kristian Lum, Megan Emily Price, and David Banks. Applications of multiple systems estimation in human rights research. *The American Statistician*, 67(4):191–200, 2013.

[5] Patrick Ball. Why just counting the dead in syria won't bring them justice. *https://foreignpolicy.com/2016/10/19/why-only-counting-the-dead-in-syria-wont-bring-them-justice-hrdag-pattern-analysis/*, October 2016.

[6] Patrick Ball, Jana Asher, David Sulmont, and Daniel Manrique. How many peruvians have died?, 01 2003.

[7] Patrick Ball. Collecting and protecting human rights data in guatemala (1991-2013). https://hrdag.org/guatemala/.

[8] Patrick Ball. Kosovo. https://hrdag.org/kosovo/.

[9] Patrick Ball. Hrdag testifies in hissène habré trial. https://hrdag.org/2015/09/23/hrdag-testifies-in-hissene-habre-trial/.

[10] Megan Price, Anita Gohdes, and Patrick Ball. Technical memo for amnesty international report on deaths in detention. *https://hrdag.org/wp-content/uploads/2016/08/HRDAG-AI-memo-2.pdf*, 2016.

[11] David G. Brizan and Abdullah U. Tansel. A survey of entity resolution and record linkage methodologies. *Communications of the IIMA*, 6:41–50, 2006.

[12] Megan Price, Jeff Klingner, and Patrick Ball. Preliminary statistical analysis of documentation of killings in the syrian arab republic. *http://www.ohchr.org/Documents/Countries/SY/PreliminaryStatAnalysisKillingsInSyria.pdf*, 2013.

[13] Megan Price, Anita Gohdes, and Patrick Ball. Updated statistical analysis of documentation of killings in the syrian arab republic. *http://www.ohchr.org/Documents/Countries/SY/HRDAGUpdatedReportAug2014.pdf*, 2014.

[14] Peter Sadosky, Anshumali Shrivastava, Megan Price, and Rebecca C. Steorts. Blocking Methods Applied to Casualty Records from the Syrian Conflict. *arXiv e-prints*, page arXiv:1510.07714, Oct 2015.

[15] Da'ud ibn Auda (David B. Appleton). Period arabic names and naming practices. *http://heraldry.sca.org/names/arabic-naming2.html*, 2003.

[16] Beth Notzon and Gayle Neson. The arabic naming system. *Science Editor*, 2005.

[17] Pedra Mashiri. Terms of address in shona: A sociolinguistic approach. *Zambezia*, pages 93–110, 1999.

[18] Kazem Taghva, Rania Elkhoury, and Jeffrey Coombs. Arabic stemming without a root dictionary. *Information Science Research Institute - University of Las Vegas, Nevada*, 2005.

[19] Jonathan Kew. Notes on some unicode arabic characters: Recommendations for usage. 2005.

[20] S.U. Aqeel, S. Beitzel, E. Jensen, D. Grossman, and O. Frieder. On the development of name search techniques for arabic. *Journal of the American Society for Information Science and Technology*, 2006.

[21] Moawia E. Yahia, Mohamad E. Saeed, and Ashwag M. Salih. An intelligent algorithm for arabic soundex function using intuitionistic fuzzy logic. 2016.

[22] Tarek El-Shishtawy. Linking databases using matched arabic names. 19, 2014.

[23] Ahmed Yousef, Azza Higazy, Tarek El Tobely, and Amany Sarhan. Web-based arabic/english duplicate record detection with nested blocking technique. 11 2013.

[24] Ioannis Kanaris, Konstantinos Kanaris, Ioannis Houvardas, and Efstathios Stamatatos. Words versus character n-grams for anti-spam filtering. *International Journal on Artificial Intelligence Tools*, 16(6):1047–1067, 2007.

[25] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, 2016. ACM.

Supplementary Material - Feature Engineering for Entity Resolution with Arabic Names: Improving Estimates of Observed Casualties in the Syrian Civil War

A HRDAG Blocking Rules

In creating matches and non-matches we have used the following HRDAG blocking rules from [10]:

1. Last 5 letters of the name & First Soundex representation & same year and month of death;
2. First 5 letters of the name & Date of death & Last Soundex representation;
3. Gender & Same exact date of death & Death Location;
4. Death Governorate & Last 5 letters of the name & Sorted last 5 letters of the name;
5. Death Location & Sorted last 5 letters of the name & Year and month of death;
6. Gender & Same sorted name & First 5 letters of the name.

As a note, soundex representation has been used as proxy for phonetic representation and blocking rules requiring the English translation were not employed (as translations were not available).

B Arabic Name Parser

In Natural Language Processing (NLP) a parser is an algorithm which understands and separates a sentence according to its logical structure. For instance, an English parser would identify in the sentence *"The cat is on the table"* the subject being *"The cat"*, the verb being *"is"* and *"on the table"* being a prepositional phrase. With the same idea in mind, an Arabic name parser was developed according to the structure highlighted above, to identify the 5 possible name parts. The rationale behind it is as follow:

1. On top of the 5 categories listed before (*ism* , *laqab* , *nisbah* , *kunya* and *nasab*) a mixed category *laqab\nisbah* was created for indistinguishable cases. On top of these, an extra category "*other* " was created in order to include anything which was not identified;
2. The process starts with the identification of the *kunya* and *nasab* , looking for the linguistic particles that flag the genitive construct, either "bin/ben/ibn/bint" or "abu/umm/oum" respectively;
3. If a *kunya* or *nasab* was identified in the name:
   - The first word of the names stripped of *kunya* or *nasab* is identified as the *ism* ;
   - If there's another name between the *ism* and *kunya* or *nasab* , this is identified as the *laqab* ;
   - The first name after the *kunya* or *nasab* is identified as *nisbah* ;
   - The rest is assigned to the *other* category.
4. If a *kunya* or *nasab* was not identified:
   - The first word of the name is identified as the *ism* [3];
   - The next two words are categorized as the mixed category *laqab\nisbah*, unless any of them is recognized among the $\sim$ 90 pre-identified *laqab* or *nisbah* by [15] (manually added to the code);
   - Anything left is assigned to the *other* category.

For completeness sake, the *other* category does not necessarily contain only uncategorized parts of the name, but it could also contain either *laqab* or *nisbah* . It could not contain *kunya* or *nasab* – even in case of duplication.

Results of applying the Arabic Parser on matches and non-matches in the validation set are shown in Table 1 below. Due to the lack of the typical names for *nasab* and *kunya* in majority of the strings,

[3] As per Abu Adua [15] it is really rare to have two *ism* in the name

only a small percentage of the strings present a *nasab* or a *kunya* . This could be a consequence of the process of westernization of Arabic names, which sees the loss of the formal linguistic particle for the name of the father or the first child. It is unclear whether this favours the use of the father's name without a genitive particle or the use of *laqab* or *nisbah* more and we hence did encode any further layer into the parsing process. The vast majority have an *ism* and at least a couple of names afterward, which get categorized into the *laqab\nisbah* section. The *other* category is picked up considerably less with respect to the former two categories (only $\sim 11\%$). Therefore, $\sim 89\%$ of the strings available are categorized. It is important to remember that the *other* category might include either *laqab* or *nisbah* - to discriminate further would require understanding the meaning of the words included in that category. A categorical adjective would represent a *laqab* while a geographical place or job a *nisbah* .

	Validation Set	
Name Part	Matches	Non-Matches
ISM	99.65%	99.75%
LAQAB	0.58%	3.23%
NASAB	0.02%	0.52%
KUNYA	1.76%	5.15%
NISBAH	1.68%	3.94%
LAQAB/NISBAH	83.97%	93.65%
OTHER	10.71%	10.9%

Table 1: Incidence of name parts in both matches and non-matches in the validation set using the implemented Arabic Parser.

C Features Specification

3.1 Binary Features.

Initially, we focus on finding a set of binary features that exhibit good separation before expanding our feature set to continuous features. We identify the following binary features:

- **Exact Agreement**
  Name strings agree if they are exactly the same;
- **Soundex Agreement**
  Name strings agree if they have identical phonetic representations through Soundex;
- **No Spaces Agreement**
  Name strings agree if they are identical after removing spaces;
- **Shuffled Agreement**
  Name strings agree if they are identical up to a permutation of word order;
- **Completely Contained Agreement**
  Name strings agree if one of the two strings is contained into the other one, up to a permutation of word order.

These initial set of features could be useful for any language[4]. Table 2 shows the percent of matches that agree (i.e. score a 1) on these five features, under the three Arabic standardization regimes outlined in Section 2.2: just removing accents (1), removing accents and removing the "AL" particle (1+2), and removing accents, the "AL" particle, and prefixes and suffixes (1+2+3). The row "Non Agreeing Strings" indicates the number of matched strings missed by all features. We note that while exact agreement implies agreement on all features, the features are not nested in general. Our analyses show that "Completely Contained Match" correctly identifies the largest number of matched pairs for both the unprocessed name strings and pre-processing under all three regimes. For all features, removal of accents (pre-processing regime (1)) produces an increase in the number of matches identified, with the exception of "Soundex Agreement", as accents are already accounted for in the sound representation. Further, it might seem counter-intuitive that the "Non Agreeing Strings" grows

[4]Assuming the soundex algorithm could be implemented for any language.

Features	No Standardization	(1)	(1+2)	(1+2+3)
Exact Agreement	76.16%	79.98%	81.1%	81.56%
Soundex Agreement	81.48%	81.48%	82.46%	82.88%
No Spaces Agreement	81.00%	85.07%	86.24%	82.45%
Shuffled Agreement	83.11%	87.20%	88.50%	88.86%
Comp. Contained Agreement	85.67%	89.86%	91.27%	92.00%
Non Agreeing Strings	4.08% (504)	3.46% (427)	2.22% (274)	6.06% (749)

Table 2: Matched strings which agree under initial binary features (rows) considering name strings as they and after having applied the three combinations of Arabic rules (columns), in both percentage and absolute numbers

from pre-processing regimes (1+2) to (1+2+3) (or, alternatively, "No Spaces Agreement" performs worse). This is because names like *Abdul* (عبد) may or may not be separated by a space from the next name. If that name has a pronoun, this will not be removed during standardization, unless there is a space. So for instance, the name *Abu Abdul Karim* could be written either as مبد يبع مبركلا or as مبركلاعبد يبع - the first three letters of the third name, *Karim*, would be removed in the first case, but not in the second, and thus the "No Spaces Agreement" feature would fail to indicate a match, while under regime (1+2) it would have. Table 3 reports the degree of separation for the binary features, i.e. the mean value of each features on matches over their value over non matches across different standardization regimes. Exact Agreement under no standardization achieves the highest degree of separation. Generally, degree of separation decreases with additional standardization steps, with the exception of "Soundex Agreement", which only decreases from (1+2) to (1+2+3).

	"Degree of Separation" between Matches and Non-Matches in Validation Set			
Features	Perc. Ratio	Perc. Ratio (1)	Perc. Ratio (1+2)	Perc. Ratio (1+2+3)
Exact Agreement	18.87	19.09	18.91	18.33
Soundex Agreement	18.01	18.01	18.06	17.22
No Spaces Agreement	18.73	18.93	18.76	18.19
Shuffled Agreement	16.95	17.09	17.01	16.75
Comp. Contained Agreement	14.88	14.98	14.46	13.26

Table 3: "Degree of Separation" of features when leaving name strings as are and applying the three combinations of Arabic rules. The value is obtained by taking the ratio between respective percentages for matched string and non matched strings. A value of 12, for instance, would mean that the the percentage of matched strings agreeing under such features is 12 times the one of non matched strings. In this sense, matched strings are 12 times more likely to agree under that specific feature.

3.2 String Features

We next consider features defined over character-level and word-level agreement. The features are:

- **Number of aligned characters from beginning**
  Number of consecutively aligned characters from the beginning of the name strings;
- **Number of aligned characters from end**
  Number of consecutively aligned characters from the end of the name strings;
- **Percentage of aligned characters**
  Percentage of characters in the shorter string that match the characters in the corresponding positions in the longer string.
- **Number of aligned words from beginning**
  Number of words which appears in the exact same position starting from the beginning of the names;
- **Number of aligned words from end**
  Number of words which appears in the exact same position starting from the end of the names;

- **Percentage of aligned words**
  Percentage of words in the shorter string that match the words in the corresponding positions in the longer string.
- **Percentage of different words**
  Percentage of the words in the shorter name that do not appear in the longer name;
- **Exact agreement when sorted**
  A binary feature that is 1 if the names agree exactly when the characters are sorted alphabetically;
- **Percentage agreeing prefixes**
  Percentage of word prefixes in the shorter name that appear in the longer name;
- **Percentage agreeing suffixes**
  Percentage of word suffixes in the shorter name that appear in the longer name.

3.2.1 Arabic Parser Based Features

Two features apply to specific parts of the name, as generated by an Arabic parser. Because name parts that occur the least frequency tend to have a high percentage of exact agreement, we focus on features that yield higher separability in the *ism* and *laqab \nisbah* name parts. The features are:

- **Percentage of aligned characters in *ism***
  Percentage of characters which appears in the exact same position in the *ism* name part of the two Arabic names - calculated over the shortest *ism* . Set to 0 if one of the two names do not have an *ism* category;
- **Percentage of aligned characters in *laqab \nisbah***
  Percentage of characters which appears in the exact same position in the *laqab \nisbah* name part of the two Arabic names - calculated over the shortest one. Set to 0 if one of the two names do not have a *laqab \nisbah* category.

In general, having more than 50% of aligned characters in the *ism* and *laqab \nisbah* is 6 and 9 times more prominent in matches.

3.3 Character N-grams and Further Work

We have explored the number of identical n-grams between the pair of names, both with and without order of appearance. We included n-grams from $n = 2$ to $n = 8$, as more than 99% of the words in names are made of 8 characters or less. We have decided to include these as potential features for their ability to capture previously undetected structures in the Arabic names. In addition, we explored some similarity metrics which were not included due to the Jaccard index and locality similarity hashing on Arabic names already in use by HRDAG (section D).

D HRDAG Features

Keeping in mind that the interest in features generation was to improve on the performances already provided by the features currently in use by HRDAG, we have listed them below according to [10, 14]. These features consider, on top of the name fields, also location (in Arabic) and date of death.

1. **Number of days between two dates of death**
   Absolute value of the difference;
2. **Agreement on year and month of death**;
3. **Agreement on first 5 letters of the name**;
4. **Agreement on last 5 letters of the name**;
5. **Jaccard index on Arabic name strings**
   Jaccard index is the ratio between the number of common words over the total number of words in the two names;
6. **Jaccard index on location**;

7. **Substring sharing with locality sensitive hashing**
   This is equivalent to the Jaccard index over all the n-grams between the two name strings;
8. **Jaro Winkler distance on alphabetically sorted names**
   Jaro Winkler is an edit distance, it measures the number of insertions and deletion required to change the first name string into the other.

As a note, we have excluded features which relied on the English translation of record as translations were not available.