Name-Ethnicity Classification and Ethnicity-Sensitive Name Matching

Article *in* Proceedings of the AAAI Conference on Artificial Intelligence · January 2012

DOI: 10.1609/aaai.v26i1.8324

2 authors, including:

Pucktada Treeratpituk
Pennsylvania State University
24 PUBLICATIONS **525** CITATIONS

SEE PROFILE

Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence

Name-Ethnicity Classification and Ethnicity-Sensitive Name Matching

Pucktada Treeratpituk
Information Sciences and Technology
Pennsylvania State University
University Park, PA, 16802, USA
pxt162@ist.psu.edu

C. Lee Giles
Information Sciences and Technology
Computer Science and Engineering
Pennsylvania State University
University Park, PA, 16802, USA
giles@ist.psu.edu

Abstract

Personal names are important and common information in many data sources, ranging from social networks and news articles to patient records and scientific documents. They are often used as queries for retrieving records and also as key information for linking documents from multiple sources. Matching personal names can be challenging due to variations in spelling and various formatting of names. While many approximated name matching techniques have been proposed, most are generic string-matching algorithms. Unlike other types of proper names, personal names are highly cultural. Many ethnicities have their own unique naming systems and identifiable characteristics. In this paper we explore such relationships between ethnicities and personal names to improve the name matching performance. First, we propose a name-ethnicity classifier based on the multinomial logistic regression. Our model can effectively identify name-ethnicity from personal names in Wikipedia, which we use to define name-ethnicity, to within 85% accuracy. Next, we propose a novel alignment-based name matching algorithm, based on Smith–Waterman algorithm and logistic regression. Different name matching models are then trained for different name-ethnicity groups. Our preliminary experimental result on DBLP's disambiguated author dataset yields a performance of 99% precision and 89% recall. Surprisingly, textual features carry more weight than phonetic ones in name-ethnicity classification.

Introduction

Personal name matching is very crucial in many applications. Personal names are often used as queries for retrieving documents; searching for scientific papers by a particular author, or for news articles on public figures. Because of its prevalence, personal names are also often used as a key field to match records from multiple sources.

Personal names are very different from other types of names such as names of products or organizations. First, unlike product names, many person names have multiple valid spelling variations, for instance '*Arafat*' and '*Arafaat*' are valid variations of the same name. Individuals also frequently use nicknames in their daily life, for example '*Bill*' instead of the more formal '*William*.' Personal names also may change over time. For instance, after marriage individuals might adopt their spouses's last names or append that name to their maiden names. Some even change how their names are written, when moving to another country. Moreover, names are often recorded in different formats in different data sources; some with the full names, some with just last names and the first initials. These issues and others make matching of personal names challenging.

Table 1: Example of different types of name variations for different ethnicities

Name	Variation
Fatimah bint Tariq bin Khalid al-Fulan	Fatimah al-Fulan
Pedro Juan López Rodríguez	Pedro López
Mao Zedong	Mao Tse-tung
Heung-Yeung Shum	Harry Shum
Li Wei Gang	Weigang Li

One of the more unique aspects of personal names is that they are highly cultural. Not only are certain names more common in different ethnic groups, but many cultures also have their own unique naming conventions. Spanish names can consist of a composite first name and two family names, e.g. '*Pedro Juan López Rodríguez*.' Arabic names often reference ancestral names. An example is the Arabic name '*Fatimah bint Tariq bin Khalid al-Fulan*,' which means *Fatimah*, daughter of *Tariq*, son of *Khalid*, of the *Fulan* family.

Additionally, differences in ethnicity can also determine possible variations of a name thus impact the name matching performance. Different languages have different ways to transliterate their personal names not in Latin into Latin alphabets. Some such as Chinese even have multiple transliteration standards ('*Mao Zedong*' ⇔ '*Mao Tse-tung*'). Name-ethnicity also determines which name variations are valid. For example, for English names if the middle names or the initials conflict, the names do not match ('*Jim M Brown*' ≠ '*Jim E Brown*'). However, for Arabic names, this is not always the case. '*Khalid Bin Hasan Bin Ahmad Fazul*' could match with '*Khalid Bin Hasan Fazul*' but not with '*Khalid Bin Ahmad Fazul*' because the first '*Bin Ahmad*' refers to one's grandfather name while the latter '*Bin Ahmad*' refers to one's father name. Certain errors or variations are also

more common in some name ethnicities than others. Chinese names are often mistakenly reversed ('*Lee Wang*' ⇔ '*Wang Lee*'), for a variety of reasons. It is more common to drop the last name in Spanish names (out of the two lastnames), than in English names ('*Pedro Juan López Rodríguez*' ⇔ '*Pedro López*').

While various personal name matching methods have been proposed (Christen 2006), most are generic and culture or ethnicity independent. Others are too specific and are specially designed to work with specific name ethnicities. In this paper we explore the relationships between ethnicities and personal names to improve the name matching performance. For this work, we consider a name-ethnicity class as a nationality category or a collection of nationality categories given to that name in Wikipedia. For more detail see the Name-Ethnicity Classification section.

This work has three main contributions. First, we present a novel name-ethnicity classifier based on the multinomial logistic regression. Our classifier identifies ethnicity of each name based on its sequences of alphabets and sequences of phonetics sound. Second, we extend the Smith-Waterman alignment algorithm to take into account various characteristics found in personal name matching. Third, we propose an ethnicity-sensitive name matching method, by combining our name-ethnicity classifier with our name alignment algorithm, where different costs are placed on different types of misalignments depending on the ethnicity of the names being compared.

Related Work

Name-Ethnicity Classification The ethnicity of a person is an important demographic indicator used in many applications including target advertising, public policy, and scientific behavioral studies. However, unlike names, ethnic information is often unavailable due to practical, political or legal reasons. Thus, especially in biomedical research, name-based ethnicity classification has gathered much interest (Coldman, Braun, and Gallagher 1988; Fiscella and Fremont 2006; Mateos 2007; Gill, Bhopal, and Kai 2005; Burchard et al. 2003). The primary used method in ethnicity classification is to compare to existing name lists. (Coldman, Braun, and Gallagher 1988) use a simple probabilistic method based on full name lists to identify people with Chinese ethnicity. (Gill, Bhopal, and Kai 2005) combine surname analysis with location information to better infer ethnicity from names. The drawback of such dictionary approaches is that it cannot classify names which do not appear in the training data and constructing such a dictionary is often difficult. Furthermore, existing dictionaries often do not have desired granularity of ethnic groups. For instance, US Census data only contains six broad ethnicity categories: Caucasian, African American, Asian/Pacific Islander, American Indian/Alaskan Native, Hispanic, and 'Two or more races'. More recently, (Chang et al. 2010) train a graphical model based on US Census names to infer ethnicities of Facebook users from names and studied the interactions between ethnic groups. (Ambekar et al. 2009) use Hidden Markov Model and decision trees to classify names into 13 ethnic groups. This work is the most similar to our name-ethnicity classifier. Both their and our approach break down each name into smaller units of character sequences, which allow the models to make ethnicity prediction on names that they have not seen before. However, their approach only models sequences of characters in different name-ethnicities, while ours utilizes both phonetic and character sequences.

Name Matching Much work has been done on name matching in the domain of information integration, record linkage and information retrieval (Bilenko et al. 2003; Christen 2006). Most of the previously proposed techniques can be categorized into two categories: phonetic-based and edit distance-based. The phonetic-based approaches convert each name string into a code according to its pronunciation, which is then used for comparison (Raghavan and Allan 2004). The edit distance approaches define a small number of edit operations (e.g. insertion, deletion and substitution), each with an associated cost. The distance between two names is then defined to be the total cost of edit operations required to change one name into another. Jaro measures the similarity based on number and order of characters shared between names. Jaro-Winkler then improves on Jaro distance by emphasizing matches of the first few characters. Other notable flavors of edit distance used in name matching include Levenshtein distance and Smith-Waterman distance (Freeman, Condon, and Ackerman 2006). An excellent review of commonly used name matching methods can be found in (Christen 2006). Recently, (Gong, Wang, and Oard 2009) propose a transformation based approach, where they compute the best transformation path between names based on three types of operation: abbreviation, omission and sequence changing. SVM is then used to learn the final decision rule. Their approach is somewhat related to our name matching method. Both attempt to find a mapping between two names; for them, it is the best transformation path using a graph-based algorithm, and for us, it is the optimal alignment through an alignment algorithm. However, their approach assumes universal cost for each type of transformations, while our cost function depends on the ethnicity of the names.

Name-Ethnicity Classification

In this section, we describe the process we use to collect a list of names and the features used in our name-ethnicity classifier.

Extracting Names from Wikipedia

Inspired by (Ambekar et al. 2009), we take advantage of Wikipedia and their categories as the source for collecting personal names of different ethnicities. To cultivate a list of personal names for a given nationality, we use the following procedure. First, for each target nationality N, we pick the Wikipedia category N_people as the root node. We then employ BFS to transverse all subcategories and pages reachable from the root node up the the depth of 4. Simple heuristics is used to restrict the link transversals. For example, we only include subcategories whose titles contain the word '*people*' or '*s of*' (eg. '*Members of the Institut de France*') or

Table 2: Name-Ethnicity data gathered from Wikipedia

Ethnicity	#Names	Wikipedia Categories
MEA	10,559	Egyptian, Iraqi, Iranian, Lebanese, Syrian, Tunisian
IND	21,271	Indian
ENG	28,624	British
FRN	29,271	French
GER	35,101	German
ITA	23,328	Italian
SPA	15,154	Columbian, Spanish, Venezuelan
RUS	19,580	Russian
CHI	10,385	Chinese
JAP	17,790	Japanese
KOR	3,750	Korean
VIE	859	Vietnamese
Total	215,672	

end with plural nouns (eg. '*entertainers*'). An example of a valid path is shown below:

French people → *French people by occupation*
→ *French entertainers*
→ *French magicians*
→ *Alexander Herrmann*

The leaf nodes resulting from such transversal are then collected as personal names for that nationality. Note that neither our heuristics nor Wikipedia categories are perfect. For instance, names under 'British people of Indian descent,' which are of Indian ethnicity, will be filed under English names. Non-personal names could also be included. For instance, musical group names such as '*Spice Girls*' is included because it is under '*British musicians*'. Thus, we also manually curate the resulting name lists, removing any such obvious missassignments as best as we could.

In the end, we gather a total of 215,672 personal names (after curation) from 19 nationalities, which are then grouped into 12 ethnic groups as shown in Table 2. Personal names of Egyptian, Iraqi, Iranian, Lebanese, Syrian and Tunisian are grouped together as Middle Eastern names (MEA) and names of Spanish, Columbian, Venezuelan are grouped together as Spanish names (SPA).

Name-Ethnicity Classifier and Features

We then train a multinomial logistic regression classifier to identify ethnicity of different names based on characters and phonetic sequences. The intuition is that names of different ethnicity have identifiable sequences of alphabets and phonetics. The multinomial logistic regression is a logistic regression that is generalized to more than two discrete outcomes. In a multinomial logistic regression, the conditional probabilities are calculated as follows:

$$Pr(Y_i = y_K|X_i) = \frac{1}{1+\sum_{l=1}^{K-1} exp(\beta_{l,0}+\beta_l^T \cdot X_i)}$$

$$Pr(Y_i = y_k|X_i) = \frac{exp(\beta_{k,0}+\beta_k^T \cdot X_i)}{1+\sum_{l=1}^{K-1} exp(\beta_{l,0}+\beta_l^T \cdot X_i)} \qquad k=1,...,K-1$$

Table 3: Example of features used in Name-Ethnicity classifier

Feat Type	String	⟶	Features
nonASCII	Pedro López	⟶	ó
charNgram	$pedro$ +lopez+	⟶	$p, pe, ed, ... (2gram) $pe, ped, ... (3gram)
dmpNgram	PTR +LPS+	⟶	P_m, PT$_m$, ... PT_m, PTR$_m$, ...
soundex	P360 L120	⟶	P360$_s$, L120$_s$

For the name-ethnicity classification, Y_i is the random variable for the ethnicity of the name i and X_i represents the corresponding feature vector. The set $\{y_1, ..., y_K\}$ is the set of K ethnicity classes. The set of coefficients $\{\beta_{l,0}, \beta_l\}_{k=1...K}$ are estimated by a maximum a posteriori probability (MAP) through iterative process. The normal distribution is used as prior on the coefficients.

Our classifier utilizes four types of features: (1) *nonASCII*, (2) *charNgram* – character ngrams, (3) *dmpNgram* – Double Metaphone ngrams and (4) *Soundex*. The *nonASCII* features consist of non-ASCII characters (such as 'ä', 'é') present in the name. The *charNgram* features are character bigrams, trigrams, and four-grams of the name after normalization, where all non-ASCII characters are mapped to their ASCII equivalences ('ä' → 'a'). The phonetic characteristics of the name are represented by *dmpNgram* and *soundex* features. To compute *dmpNgram* features, each name token is first encoded with the Double Metaphone algorithm. The bigrams, trigrams and four-grams are then generated from the resulting encoding. Lastly, *soundex* features are simply Soundex encodings of each token in the name. The Soundex and Double Metaphone algorithms are popular phonetic algorithms for encoding a word according to its phonetic sound. The Soundex algorithm encodes each word as a code of four characters: the first character of the word and a three digit code representing its phonetic sound (Knuth 1973). For instance, Both '*Steven*' and '*Stephen*' are encoded as 'S315'. Double Metaphone is a more advance phonetic algorithm than Soundex, designed to better handle non-English words (Phillips 2000). It can also return multiple codes in the case of words with phonetic ambiguity. Double Metaphone encodes each word into a code of 16 consonant symbols, e.g '*Stephen*' is encoded as 'STFN.'

Before the ngram generation, both for *charNgram* and *dmpNgram*, we pad the beginning and the ending of each token with '*$*.' To differentiate last names from other tokens, the last tokens are padded with '+' instead of '*$*.' For example, '*Pedro López*' would be padded to '*$pedro$ +lopez+*,' resulting in character ngrams '$p', 'pe', 'ed', ... , 'opez', and 'pez+.' The full example of features extracted for the name '*Pedro López*' is shown in Table 3.

Ethnicity-Sensitive Name Matching

In this section, we present the overview of the ethnicity-sensitive name matching algorithm. To calculate a probability that a pair of names match, we first tokenize each name into name tokens. A dynamic programming based on

Smith–Waterman algorithm is applied to find the optimal alignment between the two tokenized names. Features are then extracted from the optimal alignment. A logistic regression model is then applied to convert features into the match probability. Using the name-ethnicity classifier to separate names into ethnic groups, we train separate logistic regression models for each name-ethnicity. Now we discuss each step in more detail.

Computing Optimal Alignment

We use a modified version of Smith–Waterman algorithm to find the optimal alignment between two names. The Smith–Waterman algorithm is a popular alignment algorithm for identifying common molecular subsequences (Smith and Waterman 1981). However, the standard Smith–Waterman operates on sequences of characters (e.g. aligning characters in 'ACAT' with 'AGCA') and only allows exact match between characters. We, on the other hand, are interested in aligning sequences of name tokens, not characters. In addition, name tokens can align without being an exact match (such as '*Kathy*' and '*Katharine*'). Therefore, we extend the Smith–Waterman algorithm to token alignment and replace its exact match comparison with an approximate matching function.

For a given name pair $p = (p_0, ..., p_m)$ and $q = (q_0, ..., q_n)$, where p_i, q_j are word tokens in p and q respectively, we define the token similarity between the token p_i and q_j, denoted by $S(p_i, q_j)$, as follows:

$$S(p_i,q_j)=\begin{cases} \textit{Jaro-Winkler}(p_i,q_j) & \text{if } |p_i|\geq 2 \text{ and } |q_j|\geq 2 \\ 0.95 & \text{if } (|p_i|=1 \text{ or } |q_j|=1) \text{ and } init(p_i)=init(q_j) \\ 0 & \text{otherwise} \end{cases}$$

where $init(a)$ = the first character of a. Basically, we measure the similarity between tokens with the *Jaro-Winkler* similarity, which works well for short strings. But in the case where one of the token is just an initial, the similarity is 0.95 if their initials are compatible, such as between p_i = '*E*' and q_j = '*Edward*,' and 0 otherwise. The value 0.95 is arbitrary chosen, because we want the similarity of the initial-match cases to be sufficiently high, but still less than those of exact matches.

We then define the scoring matrix, H, where $H(i, j)$ represents the maximum similarity score between $(p_0, ..., p_i)$ and $(q_0, ..., q_j)$, in term of a similarity function S as follows:

$$H(i,0)=0, \quad 0\leq i\leq m$$

$$H(0,j)=0, \quad 0\leq j\leq n$$

$$H(i+1,j+1)=$$

$$max\begin{cases} H(i,j)+S(p_i,q_j) & \text{if } S(p_i,q_j)>0.8 & //p_i \textit{ matches } q_j \\ H(i,j-1)+1 & \text{if } p_i=q_{j-1}q_j & //p_i \textit{ spans 2 tokens} \\ H(i-1,j)+1 & \text{if } p_{i-1}p_i=q_j & //q_j \textit{ spans 2 tokens} \\ H(i+1,j) & & // \textit{ skip } q_j \\ H(i,j+1) & & // \textit{ skip } p_i \end{cases}$$

$$\text{for} \quad 0\leq i\leq m-1, \quad 0\leq j\leq n-1$$

The matrix H can be computed using dynamic programming. Unlike in the standard Smith–Waterman, where each cell can align with at most one other cell, our name token can align with upto two tokens in the corresponding name. This allows our alignment algorithm to account for a common problem in name matching, where multiple name segmentations exist for instance ('*DeFélice*' ⇔ '*De Félice*') and ('*Min Seo Kim*' ⇔ '*Minseo Kim*'), without any special preprocessing steps. After the matrix H is computed, the traceback procedure similar to the one in Smith–Waterman algorithm, can be used to find the optimal alignment. Figure 1 shows an example of the matrix H for aligning '*Marcelino B Santos*' and '*Marcelo Bicho Dos Santos*' and the resulting optimal alignment are shown in Figure 4a.

	$-$	$Marcelo$	$Bicho$	Dos	$Santos$
$-$	0.0	0.0	0.0	0.0	0.0
$Marcelino$	0.0	0.96	0.96	0.96	0.96
B	0.0	0.96	1.91	1.91	1.91
$Santos$	0.0	0.96	1.91	1.91	2.91

Figure 1: Scoring matrix H for aligning '*Marcelo Bicho Dos Santos*' and '*Marcelino B Santos*.' The arrows show the path of the optimal alignment generated by the traceback procedure.

Table 4: Examples of the optimal alignments between (a) '*Marcelo Bicho Dos Santos*' and '*Marcelino B Santos*.' (b) '*Juan Ginés Sánchez Moreno*' and '*G Lopez Moreno*.'

(a)	p	Marcelo	Bicho	Dos	Santos
	q	Marcelino	B	-	Santos
	$S(p_i, q_j)$	0.96	0.95	$<skip>$	1.0
(b)	p	Juan	Ginés	Sánchez	Moreno
	q	-	G	Lopez	Moreno
	$S(p_i, q_j)$	$<skip>$	0.95	$<con>$	1.0

Since name field reversal such as ('*Min Seo Kim*' ⇒ '*Kim Min Seo*') is very common, especially with East Asian names, we introduce a shift operator α such that $\alpha_t(p)$ shifts each token in p, t positions to the right for $t > 0$ and to the left for $t < 0$. So $\alpha_1(p) = (p_m, p_0, ..., p_{m-1})$ and $\alpha_{-1}(p) = (p_1, ..., p_m, p_0)$. Let $\Omega(p, q)$ denotes the optimal alignment between p and q, and $\Omega_s(p, q)$ denotes the optimal alignment with the shift operation. Then

$$\Omega_s(p,q) = \underset{\Omega(\alpha_t(p),q), t\in\{-1,0,1\}}{\arg\max} \mathcal{M}(\Omega(\alpha_t(p),q)))$$

where $\mathcal{M}(\Omega(p, q))$ is the similarity score for the alignment $\Omega(p, q)$ in the matrix H. So $\Omega_s(p, q)$ is the best alignment out of all alignments with various shifts.

Name Matching Probability

From the resulting optimal alignment $\Omega_s(p, q)$, we then categorize each match/mismatch based on its location. For example, we differentiate unmatched tokens at the end of a name from unmatched tokens at the beginning of a name. The probability P, that a name p matches a name q is then computed based on the number of each types of match/mismatch in their optimal alignment $\Omega_s(p, q)$. More

specifically, the probability P depends on the feature vector $f = (x_1, ..., x_7)$ where

$x_1 =$ # of skip tokens before the first aligned token pair

$x_2 =$ # of skip tokens after the last aligned token pair

$x_3 =$ # of skip tokens in the middle that are initials

$x_4 =$ # of skip tokens in the middle that are non-initials

$x_5 =$ # conflicting tokens

$x_6 =$ # of shift operation in the optimal alignment $= |t|$

$x_7 =$ products of similarity scores of aligned token pairs

$$= \prod_{\substack{u,v \\ p_u \text{ align with } q_v \text{ in } \Omega_s(p,q)}} S(p_u, q_v)$$

Two tokens p_i, q_j are considered to be conflicting ($<con>$) if they both are unaligned and are between the same aligned token pairs, for example '*Sánchez*' and '*Lopez*' in Table 4b. Unaligned tokens that do not conflict with any tokens are considered skip tokens ($<skip>$), for example '*Dos*' in Table 4a and '*Juan*' in Table 4b.

To illustrate, consider the two optimal alignments shown in Table 4. In (a), there are one skip token in the middle and three aligned tokens pairs. So $x_3 = 1$ and $x_7 = 0.96 \times 0.95 \times 1$ thus $f = (0, 0, 1, 0, 0, 0, .91)$. In (b), there are two aligned token pairs, two conflicting tokens and one skip token. So $x_1 = 1, x_5 = 2$ and $x_7 = 0.95 \times 1.0$.

To compute P, we assume that the odds ratio of P is directly proportion to

$$\frac{P}{1-P} \propto {D_1}^{x_1} {D_2}^{x_2} ... {D_6}^{x_6} {D_7}^{log(x_7)}$$

where $D_1, ..., D_6, D_7$ are discounting factors for different types of alignment/misalignment $(x_1, ..., x_6, x_7)$. The odds ratio of P can be rewritten as:

$$\frac{P}{1-P} = D_0 {D_1}^{x_1} ... {D_6}^{x_6} {D_7}^{log(x_7)} \quad (1)$$

$$log(\frac{P}{1-P}) = \beta_0 + \beta_1 x_1 + ... + \beta_7 log(x_7) \quad (2)$$

$$logit(P) = \beta_0 + \beta_1 x_1 + ... + \beta_7 log(x_7) \quad (3)$$

The equation (3) above is simply a logistic regression model. Thus the optimal values for the coefficient $\beta_1, ..., \beta_7$ with respect to a dataset can be easily estimated.

In the training phase, we use the name-ethnicity classifier to classify names in the training data according to their ethnicities. Separate logistic regression models are then built for each name-ethnicity, e.g. one for Spanish names, one for Middle Eastern names and so on. In addition, a back-off model is train over all the training data. In the evaluation phase, if both names to be compared are classified as of the same ethnicity, the ethnicity specific regression model is used, otherwise the default model is used.

Currently, the token similarity function $S(pi, qj)$ is ethnicity independent. However, in the future, different token similarity functions can be used for different ethnicity names. For instance, if the system detects that the names being compared are Chinese, a special similarity function that includes the mapping between Hanyu-Pinyin and Wade-Giles (two different transliteration systems for Mandarin Chinese) can be used instead. Additionally, in a real system, one can also improve the similarity function $S(pi, qj)$ by incorporating nickname dictionaries for different ethnic groups; for example, mapping 'Bartholomew' with 'Bart,' or 'Meus' for English names, and mapping the Chinese first name 'Jian' to its Western nickname 'Jerry.'

Table 5: The precision, recall and F1 measure of the name-ethnicity classifier for each ethnicity.

Ethnicity	Precision	Recall	F1
MEA	0.79	0.78	0.79
IND	0.89	0.86	0.87
ENG	0.79	0.85	0.82
FRN	0.80	0.80	0.80
GER	0.84	0.85	0.85
ITA	0.85	0.86	0.85
SPA	0.82	0.79	0.81
RUS	0.90	0.85	0.81
CHI	0.92	0.90	0.91
JAP	0.97	0.95	0.96
KOR	0.93	0.92	0.92
VIE	0.93	0.83	0.88
Accuracy	0.85		

Experiments

Name-Ethnicity Classification on Wikipedia

To assess the performance of our name-ethnicity classifier, we randomly split the name list collected from Wikipidia into 70% training data and 30% test data. Table 5 shows the precision, recall and F1 measure of the classifier for each ethnicity. The overall classification accuracy is 0.85, with Japanese as the most identifiable name-ethnicity (F1=0.96), followed by Korean (F1=0.92). In general, the classifier does well identifying East Asian names (CHI, JAP, KOR, and VIE) with over 90% precision and recall with just one exception, VIE's recall. The most problematic class is Middle Eastern names (MEA) with 0.79 F1, followed by French with 0.80 F1. We also ran another experiment without any diacritic features; every name is normalized to its ASCII equivalence. With just ASCII features, the classification accuracy drops down slightly to 0.83. This suggests that even without diacritics, each name-ethnicity still has identifiable characteristics that can be used for classification.

The confusion matrix between different name-ethnicities is shown in Figure 2. We observe that Middle Eastern names are mostly confused with Indian names. This is not unexpected, since India has a large population of Muslims. Most of the confusion for European names are with other European names, especially between English, French, and German. The majority of confusion for Russian names are with German names.

We also examine the coefficient of each feature learned by the classifier. The most predictive features for each non-English ethnic group, are listed in Table 6 according to their coefficients in the logistic regression. For instance, the 'bh'

Table 6: Most predictive features, excluding *nonASCII*, for each name-ethnicity according to the ethnicity classifier. '$' represents the word boundary and '+' indicates the lastname boundary.

Ethnicity	Feature	Example
MEA	ou	Jassem Khallou*ou*fi
IND	bh	*Bh*airavi Desai
FRN	ien$	Henri Chreti*ien*
GER	sch	Gregor *Sch*neider
ITA	one+	Giovanni Falc*one*
SPA	de	Jesus *de* Polanco
RUS	v+	Viktor Yakushe*v*
CHI	ng$	Jin*g* Huan*g*
JAP	tsu	Yutakayama Hiromi*tsu*
KOR	ae	Song T*ae* Kon, Nam Yun B*ae*
VIE	nh	Nguyen Di*nh*, Li*nh* Chi

character sequences (as in '*Bhairavi Desai*') are highly predictive of Indians names. The 'sch' character sequences is more unique in German names, while 'v+' (the lastname ending with 'v') is a good indicator for names of Russian ethnicity. An example of notable phonetic features picked up by our classifier is Soundex S400 for Middle Eastern names (corresponding to '*EL*' as in '*Hatem El Mekki*'). It is interesting to note that the majority of the top features are character features. This suggests that the written form of a name is a better indicator of its ethnicity than its phonetic. In general, the top features learned by our classifier comply with our expectations.

Overall, our name-ethnicity classifier can infer ethnicity from names with fairly high accuracy. Our result is overall comparable to the previously reported attempt in (Ambekar et al. 2009), and is significantly better at identifying some ethnic groups such as German, and East Asian names. Using smaller units of names, instead of the full names, as features not only makes the classifier generalize better to unseen names, but should also help reduce the number of training data needed in order to include a new ethnic class. One interesting direction for future work would be to compare the effect of the size of the training data on the performance between our model and theirs.

Name Matching Evaluation

We evaluate our name matching algorithm on the DBLP author dataset (Dataset 2) (Lange and Naumann 2011). The dataset contains 10,000 pairs of ambiguous author records from DBLP, that were manually cleaned due to author requests, or by fine-tuning heuristics. The dataset was created to be a challenging testbed for author record disambiguation. We use the subset of the data (2500 record pairs), which comprises only of author records from authors with multiple name aliases.

As the baselines, we implement four standard string matching techniques, Jaro-Winkler and Levenstein editing distance, together with their recursive versions (L2 Jaro-Winkler and L2 Levenstein) using the SecondString library[1].

[1]http://secondstring.sourceforge.net/

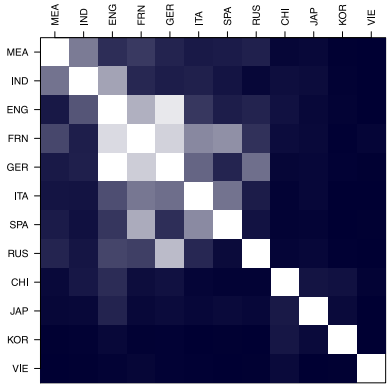

Figure 2: The confusion matrix of name-ethnicity classification on Wikipedia data. The rows are the true ethnicities, and the columns are the predictions. Darker tiles indicates less confusion between ethnic pairs, while lighter tiles represent more confusion.

We evaluate the performance of all baselines with various thresholds, and plot their precision and recall in Figure 3. The maximal F1 for all possible threshold values is 0.75 for Jaro-Winkler, 0.70 for Levenstein distance, 0.80 for L2 Jaro-Winkler, and 0.77 for L2 Levenstein.

We create three ethnicity-dependent name matching models; one for Middle Eastern names (MEA), one for Spanish names (SPA), one for East Asian names (CHI, JAP, KOR, VIE), and one default model, which is trained on all ethnicity names. The three ethnic groups are chosen since they have quite distinct naming conventions compared to typical Western names. We train and evaluate the ethnicity-sensitive name matching models using 10-folds cross validation. Our model achieves 0.99 precision, 0.89 recall and 0.94 F1 measure. This is 14% improvement in F1 over the best standard similarity measures (L2 Jaro-Winkler). Our algorithm scores extremely high precision. While the recall is at 0.89, which is slightly lower, it is still quite high. When examining the names that their matches were not correctly recalled, we observe that some of them would be difficult even for human without using other information. For instance, '*Hedvig Sidenbladh*' and '*Hedvig Kjellström*' are marked as the same author in the DBLP dataset, so as '*Maria-Florina Balcan*' and '*Maria-Florina Popa*.' These are examples of authors that changed their last names. One drawback of training separate models for each ethnicity is that we effectively make the training data for each model smaller. Thus we would like to test our model with other larger dataset in the future and to investigate the impact on performance.

Conclusion

We propose a novel name-ethnicity classifier based on the multinomial logistic regression. Our name-ethnicity classifier is trained and evaluated on Wikipedia data, achieving around 85% accuracy. We observe that name strings were usually more important than phonetic sequences for classifi-

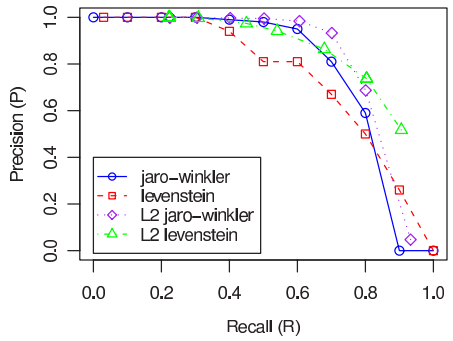

Figure 3: The recall-precision curves of the baseline edit-distance measures on DBLP data. The optimal F1 is 0.75 for Jaro-Winkler (R=0.7, P=0.81), 0.70 for Levenstein distance (R=0.6, P=0.81), 0.80 for L2 Jaro-Winkler (R=0.7, P=0.93), and 0.77 for L2 Levenstein (R=0.8, P=0.74).

cation. In addition, we also propose a novel name-matching method that is ethnicity-sensitive. The name-matching algorithm is trained and evaluated on the standard DBLP author dataset, achieving 14% improvement in F1 over the standard name similarity measures. Both our ethnicity classifier and our ethnicity-sensitive name matching also can be easily extended to include new ethnicities. Future work would be to extend this to other and possibly finer definitions of ethnicity and to larger datasets; for instance, to differentiate between French names in France and French names in Quebec, Canada.

Acknowledgment

This project has been funded, in part by, the DTRA grant (HDTRA1-09-1-0054) and the National Science Foundation. We gratefully thank Prasenjit Mitra, Madian Khabsa, Pradeep Teregowda, Sujatha Das, and Jose San Pedro for their useful discussions.

References

Ambekar, A.; Ward, C.; Mohammed, J.; Male, S.; and Skiena, S. 2009. Name-ethnicity classification from open sources. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge Discovery and Data Mining*, 49–58.

Bilenko, M.; Mooney, R.; Cohen, W.; Ravikumar, P.; and Fienberg, S. 2003. Adaptive name matching in information integration. *Intelligent Systems* 18(5):16–23.

Burchard, E. G.; Ziv, E.; Coyle, N.; Gomez, S. L.; Tang, H.; Karter, A. J.; Mountain, J. L.; Pérez-Stable, E. J.; Sheppard, D.; and Risch, N. 2003. The Importance of Race and Ethnic Background in Biomedical Research and Clinical Practice. *The New England Journal of Medicine* 348:1170–1175.

Chang, J.; Rosenn, I.; Backstrom, L.; and Marlow, C. 2010. epluribus: Ethnicity on social networks. In *Proceedings of the International Conference in Weblogs and Social Media (ICWSM)*, 18–25.

Christen, P. 2006. A comparison of personal name matching: Techniques and practical issues. *Workshop on Mining Complex Data (MCD '06)*.

Coldman, A. J.; Braun, T.; and Gallagher, R. P. 1988. The classification of ethnic status using name information. *Journal of Epidemiology and Community Health* 42:390–395.

Fiscella, K., and Fremont, A. M. 2006. Use of geocoding and surname analysis to estimate race and ethnicity. *Health Service Research* 41:1482–1500.

Freeman, A. T.; Condon, D. S. L.; and Ackerman, C. M. 2006. Cross linguistic name matching in English and Arabic: a one to many mapping extension of the Levenshtein edit distance algorithm. *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*.

Gill, P. S.; Bhopal, R.; and Kai, J. 2005. Limitations and potential of country of birth as proxy for ethnic group. *British Medical Journal* 330:196.

Gong, J.; Wang, L.; and Oard, D. 2009. Matching person names through name transformation. In *Proceeding of the 18th ACM Conference on Information and Knowledge Management*, 1875–1878.

Knuth, D. 1973. Fundamental Algorithms: The Art of Computer Programming.

Lange, D., and Naumann, F. 2011. Frequency-aware Similarity Measures: Why Arnold Schwarzenegger is Always a Duplicate. *In Proceedings of the 20th ACM CIKM Conference on Information and Knowledge Management* 24–28.

Mateos, P. 2007. A review of name-based ethnicity classification methods and their potential in population studies. *Population Space and Place*.

Phillips, L. 2000. The double metaphone search algorithm. *C/C++ Users Journal* 18(6).

Raghavan, H., and Allan, J. 2004. Using Soundex Codes for Indexing Names in ASR documents. *In Proceedings of the HLT-NAACL 2004 Workshop on Interdisciplinary Approaches to Speech Indexing and Retrieval*.

Smith, T., and Waterman, M. 1981. Identification of common molecular subsequences. *The Journal of Molecular Biology*.