Using Place Name Data to Train Language Identification Models

Stanley F. Chen, Benoît Maison

IBM T.J. Watson Research Center
P.O. Box 218, Yorktown Heights, NY 10598
{stanchen,bmaison}@us.ibm.com

Abstract

The language of origin of a name affects its pronunciation, so language identification is an important technology for speech synthesis and recognition. Previous work on this task has typically used training sets that are proprietary or limited in coverage. In this work, we investigate the use of a publically-available geographic database for training language ID models. We automatically cluster place names by language, and show that models trained from place name data are effective for language ID on person names. In addition, we compare several source-channel and direct models for language ID, and achieve a 24% reduction in error rate over a source-channel letter trigram model on a 26-way language ID task.

1. Introduction

The language of origin of a name affects its pronunciation; for example, the name *Jean* is pronounced differently in the names *Billie Jean King* and *Jean-Paul Sartre*. Thus, automatic language identification has application in speech synthesis and recognition tasks, *e.g.*, spelling-to-sound conversion [1, 2]. Language identification can be framed as the following task: given a word or sequence of words W, find the language L^* that maximizes $\Pr(L|W)$:

$$L^* = \arg\max_L \Pr(L|W) = \arg\max_L \Pr(W|L)\Pr(L) \quad (1)$$

Past work in language ID has included both *source-channel* approaches, where one estimates the two distributions $\Pr(W|L)$ and $\Pr(L)$; and *direct* (or *channel*) models which estimate $\Pr(L|W)$ directly. Common to both of these approaches is the need for training data consisting of word or name lists for each language under consideration. However, rich, wide-coverage name lists are difficult to acquire, and past work on language ID has generally used data that is proprietary and/or limited in extent and coverage. Complicating the task is that most applications require data in a single alphabet (*e.g.*, the Latin alphabet).

In this work, we investigate the use of the publically-available GEOnet Names Server database provided by the National Imagery and Mapping Agency (USA) in training language ID models. This database contains over 5 million place names rendered in the Latin alphabet, and covers most nations in the world. To use this data for language ID, we need to categorize place names by language of origin. We describe a novel algorithm for clustering place names by language; we then manually assign language identities to each cluster. To evaluate this clustering, we compare place name models against models trained on (non-name) word lists on a small person name language ID task, and find that place models outperform the word models by a wide margin.

In the latter part of this paper, we compare several models for language ID, including both source-channel and direct models. We implement a baseline source-channel letter n-gram model, as well as a maximum entropy direct model and extensions of models by Vitale [1] and Yvon [3]. We evaluate these models on a wide-coverage language ID task, and find that source-channel n-gram models perform well. Due to space limitations, many salient details have been omitted in this document; a full description of this work can be found in [4].

2. Processing place names for language ID

In this section, we describe how we processed the raw place name data to yield the language-specific training sets we used in our experiments. Starting with data downloaded from the GEOnet Names Server,[1] we tokenized place names by spacing, converted names to uppercase, and discarded all tokens contained in the *Enable* Scrabble dictionary[2] as many foreign place names contain English terms. The GEOnet Names Server does not have place names for the U.S.; for U.S. data, we downloaded place names from the United States Geological Survey web site.[3] To come up with an initial pool of country data, we took all countries whose processed data was above a certain size (approximately 5000 unique tokens); this resulted in 90 countries not including the U.S.

2.1. Clustering country data

Consider the following model of generating place names W given the country C they are located in:

$$\Pr(W|C) \approx \sum_L p(L|C)p(W|L) \quad (2)$$

where L is a hidden variable encoding a word's language of origin, $p(L|C)$ estimates what fraction of a country's place names comes from each language, and $P(W|L)$ is a model of place spellings given the language. It seems plausible that if we simply do maximum likelihood (ML) estimation of the preceding models, we may achieve reasonable results. (Given the above models, we can label the language of a place W as $\arg\max_L p(L|W,C) = \arg\max_L p(L|C)p(W|L)$.)

However, past work indicates that unsupervised ML estimation does not always yield satisfactory classification performance. Instead, we constrain the learning task by assuming that for most languages, there exists a country whose place names are primarily derived from that language; *i.e.*, for most languages L, $p(W|L) \approx p(W|C)$ for some country C. Then, we need not perform the unconstrained training of $p(W|L)$

[1] `http://www.nima.mil/gns/html/`
[2] `ftp://puzzlers.org/pub/wordlists/enable1.txt`
[3] `http://geonames.usgs.gov/`

Ⓐ *germany*: germany (0.930)
Ⓑ *austria*: germany (0.879)
Ⓒ *belgium*: neth. (0.513), france (0.275), germany (0.060)
Ⓓ *brazil*: brazil (0.779)
Ⓔ *mexico*: mexico (0.743), peru (0.077), brazil (0.052)
Ⓕ *peru*: peru (0.807), mexico (0.090)
Ⓖ *portugal*: brazil (0.731), mexico (0.070)
Ⓗ *cuba*: mexico (0.552), brazil (0.143), peru (0.094)

Table 1: Examples of country clustering. On left is country C; on right is list of countries C' and values $p(C'|C)$ (see eq. (3)).

(for hidden languages L), and can instead build fixed models $p(W|C)$ (for observed countries C). Instead of eq. (2), we have

$$\Pr(W|C) \approx \sum_{C' \in \mathcal{C}} p(C'|C) p(W|C') \tag{3}$$

The optimization problem can then be framed as finding the minimal set of countries $\mathcal{C}$ that covers all relevant languages; we can manually assign languages to countries in $\mathcal{C}$ after training.

To assure that $\mathcal{C}$ covers all relevant languages, ML estimation may be inappropriate since this only makes guarantees about the *global* likelihood. Instead, we take our objective function to be the maximum log likelihood per token loss in any *single* country C, where the loss is calculated relative to the likelihood of that country's data according to the model $p(W|C)$ trained solely on that country's place names.

We began training with $\mathcal{C}$ containing all 90 of our original (non-U.S.) countries. We repeatedly removed the member of $\mathcal{C}$ that resulted in the lowest loss in the objective function, until $\mathcal{C}$ contained a single country. We manually inspected all intermediate sets $\mathcal{C}$ and chose the $\mathcal{C}$ containing 48 countries to be the final clustering $\mathcal{C}^*$. To calculate the loss associated with removing a country, we reestimate the distribution $p(C'|C)$ using the EM algorithm before calculating the data likelihood. The model $p(W|C)$ was chosen to be a letter 5-gram model. The objective function was computed on 1000 tokens from each country held out from the data used to train the n-gram models.

In Table 1, we list examples of $p(C'|C)$ for the final clustering for several countries C; the table only includes C' for which $p(C'|C) > 0.05$. Case Ⓐ is an example of a country, Germany, included in the final set $\mathcal{C}^*$ of representative countries; case Ⓑ is an example of a country *not* included in $\mathcal{C}^*$ but which is fairly pure in a single language. Case Ⓒ is an example of a country not included in $\mathcal{C}^*$ that has place names originating from several languages; indeed, the official languages of Belgium are Dutch, French, and German. Finally, cases Ⓔ through Ⓗ show how Spanish and Portugese place names were handled. The model decided that Brazil, Mexico, and Peru were "basis" countries, and expressed all other Spanish/Portugese-speaking countries as mixtures of these countries.

2.2. Assigning place names to languages

While the preceding process produces a set of *countries* that may each correspond to a different language, ultimately we want to create data labeled by *language*. Here, we define each country in $\mathcal{C}^*$ to be its own *pseudolanguage*. For an application, we can do language ID in terms of pseudolanguages, and then manually map from pseudolanguage ID's to the "true" language ID's in a manner appropriate for the application. Then, we can share pseudolanguage models across applications while only changing the pseudolanguage-to-language map.

To select the place names to be included in the training data for each pseudolanguage, we chose a conservative strategy to emphasize quality of data over quantity. For each country/pseudolanguage $C' \in \mathcal{C}^*$, we took its training data to be all place names from countries C such that $p(C'|C) > 0.7$. That is, we only used data from "pure" countries; place names in countries that are mixtures of several pseudolanguages were not included in any training set. This process resulted in 41 pseudolanguages containing data from a single country, five containing data from two countries,[4] and two containing data from three.[5] While it may be possible to salvage data from impure countries, we leave this matter for future investigation.

3. Validating the use of place name data

In this section, we evaluate the quality of our place name data sets against non-name data sets for a five-way language ID task on person names, to assess whether place name data are effective for non-place language ID. For the baseline non-name data sets, we used the Moby word lists[6] for French, German, Italian, and Spanish; and the *Enable* Scrabble dictionary for English. We extracted the test set from players ranked by the World Chess Federation[7]; this list contains about 45,000 people and their countries of origin. We created a random test set of 500 players each from France, Germany, Italy, Spain, and England; for each name, we assume that the country of origin accurately reflects the language of origin.

In the test set, we removed abbreviations, changed non-alphabetic characters to spaces, removed single-character tokens, and changed all characters to uppercase. Similarly, diacritic marks were stripped from the baseline word lists and tokens with nonalphabetic characters were discarded. The resulting training sets were 131k, 159k, 60k, 86k, and 173k words, respectively, for French, German, Italian, Spanish, and English. For the place name data sets, for each country C we selected the cluster corresponding to the country C' maximizing $p(C'|C)$. The resulting country clusters were France, Germany/Austria, Italy, Mexico, and the United Kingdom, with training set sizes of 62k, 129k, 18k, 21k, and 17k words, respectively.

To perform language ID, we used a source-channel letter n-gram model [5] (see Section 4.1). The probability $p(W|L)$ was calculated using a letter 5-gram model trained with modified Kneser-Ney smoothing [6]. Since we have a balanced test set, we used a uniform prior over languages $p(L)$.

training set	test accuracy
word lists	62.2%
place names	78.8%

As seen above, the place name models outperformed the non-name models by a wide margin, despite the place name training sets being substantially smaller and despite the mismatch in country data used; *e.g.*, Mexican place name data were used to identify Spanish players. We conclude that place name data can be effective for person name ID, and may be much more suitable for this task than non-name word lists.

4. Models for language identification

In this section, we discuss several models for language ID; results for these models are presented in Section 5.

[4]Braz./Port., China/Taiw., Germ./Austr., S. Kor./N. Kor., Mor./Alg.
[5]Egypt/Saudi Arabia/Yemen, Bosnia and Herz./Croatia/Yugo.
[6]`http://www.dcs.shef.ac.uk/research/ilash/Moby/`
[7]`http://www.fide.com/`

4.1. Source-channel n-gram models

In this approach, the distribution $p(W|L)$ in eq. (1) is estimated using a letter n-gram model:

$$p(W|L) = \prod_{i=1}^{k} p_L(l_i|l_{i-n+1} \cdots l_{i-1}) \tag{4}$$

where a word W is composed of the letters $l_1 \cdots l_k$. As smoothing can significantly affect classification performance, we consider several smoothing algorithms: Witten-Bell smoothing [7]; modified Kneser-Ney smoothing [6]; and maximum entropy (ME) n-gram models smoothed with a Gaussian prior [8].

We also consider a novel method for smoothing ME n-gram models, *cross-model* ME smoothing. Typically, one smooths a probability $p_L(l_i|l_{i-n+1}^{i-1})$ towards the probability for a shorter n-gram $p_L(l_i|l_{i-n+2}^{i-1})$. In language ID, we have an additional source of information: the data from other languages; *i.e.,* we can smooth the language-dependent probability $p_L(l_i|l_{i-n+1}^{i-1})$ with a language-independent model $p(l_i|l_{i-n+1}^{i-1})$. We can achieve this behavior with an ME model by adding language-independent n-gram features; the Gaussian prior can then be interpreted as smoothing the language-dependent models toward the model defined solely by the language-independent features.

4.2. Direct maximum entropy classifier

It has been argued that direct models may be more appropriate than source-channel models for classification tasks as the natural objective function, conditional likelihood, is more closely related to classification performance than the natural objective function for source-channel models, joint likelihood. Here, we describe a model that uses n-grams like the models in Section 4.1, but which is direct rather than source-channel.

We use a maximum entropy model [9]:

$$p(L|W) = \frac{\exp(\sum_i \lambda_i f_i(W,L))}{\sum_{L'} \exp(\sum_i \lambda_i f_i(W,L'))} \tag{5}$$

We selected the set of features $f_i(\cdot)$ to be

$$f_{\vec{l},L'}(W,L) = \begin{cases} 1 & \text{if } W \text{ contains } \vec{l} \text{ and } L = L' \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

for all n-grams $\vec{l}$ up to a certain length (*e.g.,* 5) and languages L such that $f_{\vec{l},L'}(\cdot)$ is active in the training set. The λ_i were trained to optimize the conditional likelihood of L given W for the training set modulo smoothing with a Gaussian prior [8].

4.3. Other direct models

We consider several ways of expressing $p(L|W)$ as a function of the probabilities $p(L|\vec{l})$ for the n-grams $\vec{l}$ present in W. In [1], Vitale estimates $p(L|W)$ by the arithmetic average of the probabilities $p(L|\vec{l})$ for all trigrams present; $p(L|\vec{l})$ is estimated on training data as $\frac{c_{\vec{l},L}/u_L}{\sum_{L'} c_{\vec{l},L'}/u_{L'}}$, where $c_{\vec{l},L}$ is the count of the trigram $\vec{l}$ in language L and u_L is the number of distinct n-grams found in L. Our experiments suggest the two following modifications to this technique: normalizing $c_{\vec{l},L}$ by the total number of n-grams in language L rather than the number of unique n-grams, and increasing the n-gram length to 5.

Another model we consider is a stochastic version of the "overlapping chunks" model proposed by Yvon [3] for the spelling-to-sound problem. A word W can be represented as a sequence $\mathcal{S}$ of *overlapping chunks* of up to n letters. The probability $p(L|W)$ is estimated as the maximum of the product of chunk probabilities over all possible sequences $\mathcal{S}$: $\max_{\mathcal{S}} \prod_{\vec{l} \in \mathcal{S}} p(L|\vec{l})$, where $p(L|\vec{l})$ is estimated as above.

Finally, we also consider a non-stochastic counterpart to the previous algorithm (inspired by [3]) that assigns to each language L a score $S(L,W)$ defined as the maximum average chunk length $\max_{\mathcal{S}}(\sum_{\vec{l_L} \in \mathcal{S}} |\vec{l_L}|)/|\mathcal{S}|$, where the $\vec{l_L}$ are taken from the inventory of chunks collected from the training set.

smoothing	n	accuracy
Witten-Bell	3	62.7%
Witten-Bell	5	67.5%
modified KN	5	69.1%
ME	5	69.9%
cross-model ME	5	70.0%

model	n	accuracy
Vitale	3	44.3%
modified Vitale	5	62.7%
overlapping chunks	$\leq$6	66.8%
avg. chunk length	$\leq$7	61.3%

Table 2: Accuracy on 48-way place name test set for source-channel n-gram models (top) and other models (bottom).

5. Results

5.1. Source-channel n-gram model results

In this section, we compare various smoothing algorithms and n-gram orders on the data sets described in Section 2.2. We created a matched test set consisting of 1000 words from each of the 48 language clusters held out from the training sets, and a development test set of half the size. We take our baseline algorithm to be a trigram model with Witten-Bell smoothing, as this is similar to models from the literature [5, 2]. We present selected results at the top of Table 2.

We see that higher-order n-gram models can outperform trigram models; on the development set, we found 5-gram models gave the best performance. Furthermore, smoothing can significantly affect performance, with the maximum entropy n-gram models performing best. Cross-model ME smoothing gave almost no additional gain over plain ME smoothing. For the ME models, smoothing was parameterized by a single global variance value that was optimized on the development set.

5.2. Direct ME model results

To examine the direct ME model described in Section 4.2, we began with a small balanced training set covering five languages. We used the same data sets as used in Section 5.1, except restricted to the five languages used in Section 3 and with training sets truncated to 15000 place names. Optimizing a single global variance parameter for each model on the development set, we achieved the following test set accuracies:

direct ME	85.6%
source-channel ME	86.7%
source-channel ME, cross-model	86.8%

Due to the absence of any gain over the source-channel models and the resource requirements of training the direct model, we decided not to pursue this approach further.

5.3. Results for other direct models

We compared the different models described in Section 4.3 on the 48-way test set mentioned above; results are given in the bottom of Table 2. The best n-gram length for each model was chosen on the development set. The stochastic version of the "overlapping chunks" technique clearly outperforms its non-stochastic counterpart (*i.e.*, average chunk length), as well as Vitale's fixed-length n-gram model. We hypothesize that letting the model use n-grams of variable size has a smoothing effect that explains its better performance.

5.4. A wide-coverage person name task

In this section, we describe experiments on a 26-way person name language ID task, to demonstrate performance on a realistic, wide-coverage task. From the World Chess Federation player lists, we extracted players from all countries containing a minimum number of players and which speak primarily a single language. Players were also collected from Go player rankings[8] to boost the representation of China, Korea and Japan. Multiple countries speaking the same or very similar languages were grouped into the same cluster in the test set. Finally, names with a last name common to multiple clusters were discarded. As before, we assume the country of origin correctly identifies the language of origin of each name. This resulted in a test set containing 21349 names in 26 language clusters, ranging from 3891 players for Russia/Ukraine to 110 players for Kazakhstan; we held out a development set half the size of the test set. Results for the source-channel cross-model ME n-gram model and the overlapping chunks model were as follows:

accuracy	cross-model ME n-gram	overlapping chunks
uniform prior	65.1%	65.1%
true prior	71.7%	70.6%
(true prior)^a	73.4%	72.2%
trained prior	74.7%	73.7%

The "prior" is the prior distribution of languages $p(L)$ in eq. (1); the "true" prior is the distribution of languages calculated from the development set. For the third line in the table, we raised the true prior to a power in eq. (1), optimizing this value on the development set. Finally, the last line refers to the unconstrained training of $p(L)$ on the development set to optimize accuracy. For reference, the analogous number to the last line for the baseline Witten-Bell trigram model is 66.8%; our best model, the cross-model ME n-gram, achieves a 24% reduction in error rate over this baseline.

6. Related work

Past work in language ID for names/words has included both source-channel and direct approaches. Church [10] uses a model similar to a source-channel letter trigram model and Vitale [1] combines n-gram-based filtering rules with the direct model described in Section 4.3; both present only anecdotal results. Riis *et al.* [11] use a direct neural network model and report an accuracy of 86.4% on 4-way classification for words. Häkkinen and Tian [5] compare source-channel n-gram models with a direct decision-tree model and report that n-gram models are superior, achieving an accuracy of 71.8% on a 4-way language ID task for person names with an "extended" bigram model. Llitjós [2] uses a source-channel letter trigram model similar to our baseline model and reports a 73% accuracy on 5-way language ID for proper names.

[8] `http://www.gobase.org/information/players/`

To provide an indirect comparison with Riis *et al.* [11] and Häkkinen and Tian [5], we constructed training sets of the same size over the same language set, but using place name data instead. On a place name test set, we achieve accuracies of 90.9% (same training set size as [5]) and 92.8% (same training size as [11]) using a Kneser-Ney 5-gram model, though little can be concluded from this due to the different data sets used.

7. Discussion

In this work, we have demonstrated that public place name data can be used effectively for training wide-coverage language ID models, and can be substantially more effective than non-name word lists. Using a novel clustering algorithm, we produced a 48-way clustering of place names by language, yielding a set of language distinctions which should be sufficiently fine-grained for most language ID applications, but which still offers good inter-class discriminability: we can perform 48-way place name language ID over this set with an accuracy of 70.0%.

In addition, we compared several algorithms for language identification. Despite arguments for the suitability of direct models for classification, we found that the source-channel models we examined performed as well as or better than our direct models. Our best model, a source-channel ME n-gram model with cross-model smoothing, achieved an accuracy of 74.7% on a 26-way person name ID task, a 24% relative improvement over a baseline source-channel trigram model.

8. References

[1] T. Vitale, "An algorithm for high accuracy name pronunciation by parametric speech synthesizer," *Comp. Ling.*, vol. 17, no. 3, pp. 257–276, 1991.

[2] A. F. Llitjós, "Improving pronunciation accuracy of proper names with language origin classes," in *Proc. of the Seventh ESSLLI Student Session*, Trento, Italy, 2002.

[3] F. Yvon, "Grapheme-to-phoneme conversion using multiple unbounded overlapping chunks," in *Proc. of NeMLaP*, Ankara, Turkey, 1996, pp. 218–228.

[4] S. F. Chen and B. Maison, "A study of automatic grapheme-to-phoneme conversion," IBM Research, Tech. Rep., 2003, in preparation.

[5] J. Häkkinen and J. Tian, "N-gram and decision tree based language identification for written words," in *Proc. of ASRU*, Trento, Italy, December 2001.

[6] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," *Computer Speech and Language*, vol. 13, no. 4, pp. 359–393, 1999.

[7] T. C. Bell, J. G. Cleary, and I. H. Witten, *Text Compression*. Englewood Cliffs, N.J.: Prentice Hall, 1990.

[8] S. F. Chen and R. Rosenfeld, "A survey of smoothing techniques for maximum entropy models," *IEEE Trans. on Speech and Audio Proc.*, vol. 8, no. 1, pp. 37–50, 2000.

[9] A. Berger, S. Della Pietra, and V. Della Pietra, "A maximum entropy approach to natural language processing," *Comp. Ling.*, vol. 22, no. 1, pp. 39–71, 1996.

[10] K. Church, "Stress assignment in letter to sound rules for speech synth.," in *Proc. of ACL*, July 1985, pp. 246–253.

[11] S. K. Riis, M. W. Pederson, and K. Jensen, "Multilingual text-to-phoneme mapping," in *Proc. of Eurospeech*, 2001.