A Fine-Grained Model for Language Identification

Harald Hammarstrøm
Department of IT
Chalmers University
S-412 96 Gothenburg
Sweden
harald2@cs.chalmers.se

ABSTRACT

Existing state-of-the-art techniques to identify the language of a written text most often use a 3-gram frequency table as basis for 'fingerprinting' a language. While this approach performs very well in practice (99%-ish accuracy) if the text to be classified is of size, say, 100 characters or more, it cannot be used reliably to classify even shorter input, nor can it detect if the input is a concatenation of text from several languages. The present paper describes a more fine-grained model which aims at reliable classification of input as short as one word. It is heavier than the classic classifiers in that it stores a large frequency dictionary as well as an affix table, but with significant gains in elegance since the classifier is entirely unsupervised. Classifying a short input query in multilingual information retrieval is the target application for which the method was developed, but also tools such as spell-checkers will benefit from recognising occasional interspersed foreign words. It is also acknowledged that a lot of practical applications do not need this fine level of granularity, and thus remain largely unbenefited by the new model. Not having access to real-world multi-lingual query data, we evaluate rigorously, using a 32-language parallel bible corpus, that accuracy is competitive on short input as well as multi-lingual input, and not only for a set of European languages with similar morphological typology.

1. INTRODUCTION

The language identification problem is to decide for a natural language text which language it is written in. The usual setting is to assume that one has access to training corpora beforehand for the languages to be considered. Some language fingerprint model is built from the training corpora and then classification of unseen text (belonging to one of the languages at hand) is performed through this model.

Existing state-of-the-art techniques rely on a surprisingly simple model, namely, a frequency table of character 3-grams for each language, read off directly from the training corpora. The corresponding 3-gram frequency table for the text to be classified is then compared to each stored language by some rank-frequency metric. In practice, this approach performs very well (99%-ish accuracy) if the text to be classified is of size, say, 100 characters or more [12]. Thus the language identification problem is a solved problem for most practical applications.

However, the crude 3-character gram method has a certain drawback (which may or may not be practical problem), in that it is not monotone. That is, if two texts s_1, s_2 are classified as l_1, l_2 respectively, then it is not certain that the concatenation of s_1 and s_2 is classified as either l_1 or l_2.

We will present an alternative model which aims at reliable classification of new text as short as one word. This model combines a frequency dictionary from each training corpus and a component that tries to recognize completely unseen words by looking at affixes (which would e.g. identify a word like *jihading* 'fighting the jihad' correctly as English). This latter component is crucial, not only for languages which make more use of affixes than English, but because there will always pop up completely novel words for any natural language no matter what size the training data. The affix detection technique implemented also builds from the same training corpora and requires no extra supervision or work by a human.

There are certainly practical applications which do require reliable classification of small segments and autodetection of language switches. These include spell checkers that wish to disregard interspersed foreign words, text-to-speech systems that make intermediate use of grapheme-to-phoneme conversion likewise wish to indentify interspersed foreign words, and multilingual information retrieval systems would benefit from knowing the language(s) of the words of a short query. For a lot of other practical applications, the granularity of the proposed new model is superfluous. For these applications, the only advantage of the proposed model is elegance and absolute lack of training supervision.

The resultant language identifier is evaluated using bible corpora for 32 languages, spanning the full range of morphological typology of languages of the world [7]. Both its ability to classify short segments into one language and to autodetect short segments that may be composed of several languages, are evaluated. However, we do not compare these figures to existing systems, because they were not designed for classifying short segments accurately (and thus perform very poorly)[1]. On longer segments, i.e. 100 char-

[1]There would also have been practical problems in doing justice as many descriptions of existing systems hide information on parameter tweaking. Online systems we have

acters, performance is near perfect, and it is presumed that the state-of-the-art systems would also perform near perfect if tested on the same set.

With the improved accuracy on short segments and wide typological testing range, we hope to have met the challenges for written language identification set out in a recent survey article by [11].

All the training corpora used in this paper are bible corpora, since they are the only sufficiently large corpora available for a reasonably varied set of languages.

2. PREVIOUS WORK

My full bibliography of works dealing narrowly with written language identification spans over 100 articles, a handful of technical reports and one PhD thesis [25] – it is therefore not possible to review them all here. Many pointers to older work and language identification of speech signals are given in [19, 2]. [22] is an excellent review and comparison of techniques used in early work.

For the language identification problem in the setting as in this paper, namely, written language identification trained on reference language data, two different feature models have been prevalent. One that looks at common words and one based on character n-grams [9, 3, 6, 8] – see [15, 13] for refinements of the n. The classification can then be done by comparing input text features to reference language features using rank-order statistics. More recent work in this direction has aimed at trimming overweight feature models [20, 23] or at combining n-gram and whole word features [21]. See, however [1] for a novel, completely different approach based on words clustered on sentence-co-occurrence. (The accuracy of this identifier is comparable to the older approaches, but it is not, as claimed therein, unsupervised, because there is a very large number of manually set parametres/thresholds and word-frequency statistics are gathered from curated corpora.) There is also more recent work targeting web pages specifically [24, 16, 14], that address the proper treatment of HTML tags.

Whereas the language identification problem has variously been labelled 'easy' and 'solved' [17], it depends on whether one sets the goal higher than distinguishing non-minimal noise-free samples of European languages. Some recent articles [18, 5, 4] identify practical problems where this is not so. For instance, as far as we can ascertain, the best systems in van Noord's Online Summary[2] minimally require some 20 characters of text to make a judgment at all. Nor are they capable of realizing that a sample text is a concatenation of two languages. For example, The Xerox MLTT Language Identifier[3] classifies the sentence 'good fish prefer their snake' correctly as English, the sentence 'fina fiskar sprattlar inte ofta' correctly as Swedish, but the concatenation of the two is classified as Norwegian (even though there is actually no legal Norwegian word in either sentence).

As indicated already, the present method seeks to tackle also smaller sample texts, which is crucial in order to be able to track whether a text is a composition of words from several languages. While the classic n-gram approaches have found that a good $n = 3$, i.e. that salient morphemes can be approximated as being exactly 3 characters, a more elegant alternative is to hold this variable, so that salient affixes can have any length in any language. Furthermore, we wish to extend the testing scope, as present published testing has been only on a rather small set of European languages.

found do not allow uploading the training/test set we use, which is crucial in order to assess language-dependentness.

[2] `http://odur.let.rug.nl/~vannoord/TextCat/competitors.html` accessed the 25th of May 2005.

[3] `http://www.xrce.xerox.com/competencies/content-analysis/tools/guesser` accessed 20 Jan 2007.

3. DEFINITIONS AND PRELIMINARIES

Start with a finite non-empty alphabet Σ. The following terminology and notation will be used.

word: a non-empty finite string over Σ. Thus the set of all possible words can be denoted Σ^+. Lowercase w with subscripts will be used for variables over words. A word will be enclosed in quotes if confusion could arise otherwise.

sentence: a finite non-empty tuple of words $\langle w_1, w_2, \ldots, w_n \rangle$. Commas and brackets will be omitted when no confusion can arise. However, variables that range over tuples, e.g. $\langle l \rangle$, will always be written with brackets.

S_Σ : let $S_\Sigma = \{\langle w_1 w_2 \ldots w_n \rangle | w_i \in \Sigma^+, n \in \mathbf{N}\}$ denote the set of all possible sentences.

language: a probability distribution over sentences $L : S_\Sigma \to [0, 1]$ such that $\sum_{\langle s \rangle} L(s) = 1$.

training corpus: a finite sequence of sentences. However, we will never make use of the order of sentences, or order or words in the sentences, so a training corpus may be equated with its bag of words. Thus, if T is a training corpus, let $f_T(w)$ denote the frequency of the word w in T. Also, use $W_T = \{w | f_T(w) \geq 1\}$ for the *set* of words in the training corpus.

names and variables: Unless we are talking about existing natural languages, e.g. English, natural numbers $1, 2, \ldots$ will be used for language names. $\Sigma_1, \Sigma_2, \ldots$ will be used for their corresponding alphabets, with $\Sigma = \bigcup_i \Sigma_i$ for the mother alphabet. $L_1, L_2, \ldots$ will be used for languages, i.e. probability distributions, and coindexed $T_1, T_2, \ldots$ for training corpora (where T_i is assumed to be sampled from L_i).

The idea is of course that sentences which are illegal or ill-formed in some natural language will have zero probability and legal sentences will have a non-zero probability corresponding to their relative frequency. A natural way to see how a natural language should correspond to such a formal probabilistic language is to consider ever increasing amounts of natural language text and let the probability of each sentence be its limiting relative frequency. This correspondence requires that this limit actually exists for all sentences. If there are natural languages that do not live up to this, or which cannot be modelled so with an acceptable level of discrepancy, they should not be thought of as languages in our terminology.

Our notion of language is a generalization of the more common formalization of natural language as a *set* of sentences. We actually need this greater flexibility in order for language identifiers to exploit the fact that some words (and thus some sentences) which are legal in several natural languages may be distinguished by their different levels of frequency. It also provides a framework for gracious treatment

of new words and proper names which are so ubiquitous in open domain natural language text (such as newspaper text) that they cannot be "abstracted away". With the probability model we have the power to say that any word is possible in any language, for example as a proper name, but it is more probable that an instance of e.g. 'the' is from English than in some other language where it may have occurred as a proper name.

4. A FINE-GRAINED MODEL OF LANGUAGE IDENTIFICATION

From the input of a training corpus, the proposed model characterizes a language using the following two components:

Frequency dictionary: Stores each seen word and its (relative) frequency. The frequency of seen words is a very powerful predictor of a language.

Unsupervised affix detection: Salient affixes are extracted (in an unsupervised manner), which form the basis for a probabilistic guessing of previously unseen words.

These two components are combined into a *word emission probability* distribution that aims to predict how likely a language is to have emitted a given word. In principle, a collection of such probability distributions are sufficient to make up a standard case of language identifier that always outputs exactly one language. However, we shall also use another component, a *language holdback bias*, to enable intuitively correct identification of text that is concatenated from several languages.

4.1 Word Emission Probability

A frequency dictionary FD_l is built simply as:

$$FD_l(w) = \frac{f_{T_l}(w)}{\sum_{w' \in \Sigma} f_{T_l}(w')}$$

Following [10] we use an unsupervised algorithm to gather information on the salient affixes for a given language. The algorithm uses W_l as its input and outputs a probability distribution on character strings that aims to say whether a given segment is likely to be a characteristic prefix or suffix for the language at hand. To be more precise, the probability distribution aims to capture the notion of morpheme probability that one arrives at if: 1. A linguist does a morphemic segmentation of the word types (not words tokens) occurring in a corpus, 2. The frequencies of the individual morphemes, in prefix or suffix position, are interpreted as probabilities. For example, *-qvj* would likely get zero probability in an English corpus. An example output, adapted from [10], is given in Table 1, sorted on highest probability. The outcome of the algorithm for languages which do not have any morphology at all is a fairly even spread of probability mass over initial and final characters of the words of the language in question. For reasons of space, the reader is referred to the said paper for a discussion of the inner workings and alternative algorithms.

As mentioned, the output from the affix extraction is a probability distribution over affixes. What we need is a probability distribution over words, in which any word ending in some salient suffix should have nonzero probability. One quite reasonable way to achieve this is to assign

Table 1: Comparative figures for prefix vs. suffix detection for three sample languages.

Swedish		English		Swahili	
för-	0.097	*-ed*	0.132	*-a*	0.100
-en	0.086	*-eth*	0.109	*wa-*	0.095
-na	0.036	*-iah*	0.099	*ali-*	0.065
-ade	0.035	*-ly*	0.090	*nita-*	0.059
-a	0.034	*-ings*	0.068	*aka-*	0.049
-ar	0.033	*-ing*	0.062	*ni-*	0.046
-er	0.033	*-ity*	0.059	*ku-*	0.044
-as	0.032	*-edst*	0.058	*ata-*	0.042
-s	0.031	*-ites*	0.046	*ha-*	0.032
-de	0.031	*-s'*	0.036	*a-*	0.031
...	...	...	...	...	...

Table 2: Some indications as to the widely differing identification cues for three languages; the polysynthetic Greenlandic versus the almost isolating Haitian creole.

Language	$\|T\|$	$\|W\|$	α	$argmax_w(FD(w))$	
Greenlandic	382188	107918	0.706	*taava* (then)	0.00857
Swedish	758773	26825	0.407	*och* (and)	0.05566
Haitian creole	904915	7796	0.335	*yo* (PL/they)	0.05531

geometrically decreasing probabilities for longer and longer words. Thinking in this way, we let all observed (in W_l) word lengths get the probability mass proportional to the number of observed words with such lengths, and unseen word lengths get geometrically decreasing probability. Thus, to get a well-defined probability distribution over words based on the affix probability distribution, we multiply together the word-length mass for w with the highest (not necessarily longest!) matching, if any, affix probability, for a given word w. The details aren't interesting, but use $A_l(w)$ to denote the just described affix-based probability distribution.

Putting the affix detection together with the frequency dictionary to make an emission probability involves a related kind of estimatate. How much probability mass should be assigned to seen vs. unseen words? There are probably many similar alternatives, but here we have simply guessed that unseen words are like hapax words, and assigned the probability mass proportions to be like the proportion of hapax words: $\alpha_l = \frac{|\{w \in W_l | f_{T_l}(w)=1\}|}{|W_l|}$.

We are now ready to define emission probability:

$$P_l(w) = \begin{cases} (1-\alpha_l) \cdot FD_l(w) & \text{if } w \in W_l \\ \alpha_l \cdot A_l(w) & \text{if } w \notin W_l \end{cases}$$

It can happen that there is more mass given to an unseen word than to a (rare) seen word, even within one particular language. In fact, proportions vary quite wildly between languages, as can be seen in Table 2 with figures computed on the translations of the same bible text.

4.2 Language Holdback Bias

If we have $L_1, \ldots, L_n$ languages, the previous section shows how to construct the corresponding $P_1, \ldots, P_n$ probability distributions over words. Next, we shall define a family of probability measures over *sequences of words*. There will be one probability distribution for each language tuple of the

same length as the sequence to be measured:

$$P_{l_1 l_2 \ldots l_m}(w_1 w_2 \ldots w_m) = \prod_i P_{l_i}(w_i)$$

Given a sequence of words we could then naïvely decide which language(s) it most probably belonged to by listing each tuple of the appropriate length and computing which tuple has the highest probability of having generated the sequence of words. However, for several reasons, such an approach is not advisable. First, with n languages there are n^m language tuples so it would not be tractable to enumerate them all. Second, the probability measures so defined, the output will be the concatenation of the most probable language for each word individually. This is probably not what we want since many words that are legal in several languages differ in frequency. Consider a sequence of a million words indisputably belonging to language L_1, and, interspersed inside, a word that is legal in both L_1 and L_2 but slightly more common in L_2. The naïve language identifier would yield L_2 disregarding the suggestive surrounding million words of L_1. While it is technically not impossible that it is a concatenation of the two languages, a human would never see it as that. Third, it's not clear how to see if an input sequence is non-trivially legal in more than one way (i.e. there are several satisfactory language tuples). Either we insert some kind of threshold which would be hard to know how to set, or we have to say that pretty much all tuples are satisfactory identification of the sequence only with some degree variation.

For the first problem, it is easy to see that not all tuples need to be enumerated to get the maximally probable one (if we want only this one, rather than the probabilities for all). As defined, the emission probabilities depend only on a particular word, not anything else in the sequence, so maximas can be computed locally in the sequence and glued together as in any standard application of dynamic programming. For the second and third problem, we shall propose a refinement of the strategy that obviates the need for any thresholds.

We propose that a machine language identifier like ours should have a *bias* towards minimizing the number of times we change languages in an identification sequence. To be more precise, the prior probability that a sequence should switch language c times should decrease exponentially in c. Also, other things being equal, the longer the sequence the stronger the bias should be, i.e. it should not be less likely that a million word sequence should switch language once somewhere within it, than that a two-word sequence should switch language (once) within it. This is the way to say that having seen a million words of language L_1 counts for more than having seen just one word of L_1. We do not see any basis for this to be a sequential property, e.g. that language switches are significantly more (or less) likely after or before certain words, wherefore a (H)MM-modeling technique offers no advantage.

Formally, let $C(l_1 l_2 \ldots l_m) = |\{i | l_i \neq l_{i+1}\}|$ denote the number of times a change in language occurs in a language sequence. Clearly, we have $0 \leq c \leq m-1$. Let $\langle l \rangle = l_1 l_2 \ldots l_m$ be an arbitrary language tuple under consideration and $c = C(\langle l \rangle)$ its number of switches. Now, for any language identifier parametrized on c and m, we wish the bias, regardless of the particular languages at hand, to ensure that:

$$\frac{P(c,m)}{P(c+k,m)} \geq 2^k \qquad \text{for all } k \geq 0, m$$

$$P(c,m) > P(c,m+k) \qquad \text{for all } k \geq 1, c$$

A simple fulfilment of these is the following **Language Holdback Bias** function $B(c,m)$:

$$B(c,m) = \frac{1}{m^c} \cdot \frac{1}{\sum_{0 \leq i \leq m-1} \frac{1}{m^i}}$$

There of course alternative bias functions that also fulfill the desiderata, but this is the simplest one. Now, with the bias function defined we are ready to present our full definition of the output of the now rather sophisticated language identifier.

$$ID(w_1 \ldots w_m) = \begin{array}{l} \text{the set of all tuples } \langle l \rangle = l_1 \ldots l_m \\ \text{such that for all } \langle l' \rangle \\ B(C(\langle l \rangle), m) \cdot P_{\langle l \rangle}(w_1 \ldots w_m) \geq \\ B(C(\langle l' \rangle), m) \cdot P_{\langle l' \rangle}(w_1 \ldots w_m) \end{array}$$

The formula conveys the following: look for tuples with as few cuts (i.e. minimal c) as possible, that are such that they have higher probability, the bias respected, than any other tuple with *more* cuts. This is the key feature which eliminates the need for a threshold. Thus, for example, a word sequence will be said to be of language L_l iff it has higher probability than any division of the sequence into two parts of different languages (or three parts etc). There may be several such languages, but hardly all, so the yield will be a strong prediction.

The following more procedural reformulation of the identification function may be easier to understand. It should also make it clear that language identification is still polynomial in the sequence length, since there are still no dependencies between the word-probabilities.

1. Find minimal c such that there exists a tuple $\langle l \rangle$ with $C(\langle l \rangle) = c$ and:

$$\begin{array}{l} B(c,m) \cdot P_{\langle l \rangle}(w_1 \ldots w_m) \geq \\ B(C(\langle l' \rangle), m) \cdot P_{\langle l' \rangle}(w_1 \ldots w_m) \\ \text{for all } \langle l' \rangle \text{ with } C(\langle l \rangle) > c \end{array}$$

2. Output all tuples $\langle l \rangle$ with $C(\langle l \rangle) = c$ and:

$$\begin{array}{l} B(c,m) \cdot P_{\langle l \rangle}(w_1 \ldots w_m) \geq \\ B(C(\langle l' \rangle), m) \cdot P_{\langle l' \rangle}(w_1 \ldots w_m) \\ \text{for all } \langle l' \rangle \text{ with } C(\langle l \rangle) > c \end{array}$$

4.3 Examples

4.3.1 Example 1: The kings hon walikusoma

Consider the sequence *the kings hon walikusoma* which consists of *the*, which is of course the English definite article; *kings* is the well-known English lexical item which does occur in the training corpus – it also happens to end in *-s* which is a very common Swedish inflectional ending (but there is no lexical item 'king' or 'kings' in Swedish); *hon* is a Swedish personal pronoun, abundantly occurring in the Swedish training corpus; and *walikusoma* is a well formed Swahili word whose individual morphemes all individually occur abundantly in the Swahili training corpus – but the

Table 3: Example 1: $P_l(w)$ for a set of languages and some interesting words, followed by a selection of the more interesting tuple-probabilities.

	'the'	'kings'	'hon'	'walikusoma'
English	0.051522	0.000286	0.000003	0.000004
Swedish	0.000002	0.000040	0.000916	0.000043
Swahili	0.000218	0.000000	0.000000	0.000317

All one-language tuples

$P_{eng,eng,eng,eng}$	1.350e-016
$P_{swe,swe,swe,swe}$	2.468e-018
$P_{swa,swa,swa,swa}$	1.878e-025

Some top one-switch tuples

$P_{eng,swe,swe,swe}$	2.034e-014
$P_{eng,eng,swe,swe}$	1.465e-013
$P_{eng,eng,eng,swa}$	3.008e-015

The top two-switch tuple

$P_{eng,eng,swe,swa}$	2.701e-013

Table 4: Example 2: $P_l(w)$ for a set of languages and some words that are very easy to classify, followed by examples to indicate that the dominance of a certain zero-switch tuple over some others.

	'the'	'kings'	'are'	'there'
English	0.051522	0.000286	0.002812	0.002065
Swedish	0.000002	0.000040	0.000006	0.000035
Swahili	0.000218	0.000000	0.000004	0.000006

$P_{eng,eng,eng,eng}$	8.5467629403443202e-011
$P_{swe,swe,swe,swe}$	1.2961894211016589e-020
$P_{swa,swa,swa,swa}$	2.5363460513704776e-023
...	

perfectly well-formed word 'walikusoma' does not occur in the training corpus (it would mean 'they read you').

The individual word-probabilities as well as a selection of the more interesting tuple-probabilities for the sequence as a whole, are shown in Table 3. As can be seen, the $P_{eng,eng,swe,swa}$ value beats all tuples with zero or one switches. It also happens to beat all tuples with three switches and it is the only such tuple. Therefore, in this case, the output will be exactly English, English, Swedish, Swahili.

4.3.2 Example 2: The kings are there

The complicated interaction seen in the previous example does not disturb the "normal" easy class of classifications. Table 4 shows the word-probabilities for the almost trivial sentence *the kings are there*. There is a certain zero-switch tuple which is way ahead of the others. As it also beats all one-switch tuples (and no other zero-switch tuple does), it will be the output of the identifier.

4.3.3 Example 3: De la

There are instances where there are several "winning" tuples, though informal tests show that this is not achieved very often. The sequence *de la* is very common to both Spanish and French. In English it is not common at all. In Swedish *de* is a personal pronoun so it enjoys a certain frequency, whereas *la* is not a word in (bible) Swedish. Similarly, *la* is a negator in Swahili and is therefore fairly frequent. Table 5 shows the relevant probabilities. The output

Table 5: Example 3: $P_l(w)$ for a set of languages and two words, followed by a selection of the more interesting tuple-probabilities.

	'de'	'la'
French	0.029172	0.016325
English	0.000000	0.000000
Swedish	0.008400	0.000001
Swahili	0.000000	0.001517
Spanish	0.033905	0.014280

$P_{fre,fre}$	0.0003174886
$P_{spa,spa}$	0.0003227756
$P_{spa,fre}$	0.0001844997
...	...

will be only the tuples spa, spa and fre, fre, because tuples like swe, swa and spa, fre lose out because of the bias, favouring few switches.

5. EVALUATION AND DISCUSSION

Three extensive tests were performed using a parallel corpus of the bible in 32 languages, which contains languages from the isolating Maori to the record holding polysynthetic Greenlandic [7]. In order to get a sufficiently cross-language comparable evaluation, size and randomness were equalized between languages the following way. A random verse from each chapter was selected (there are 1209 chapters in the bible). This was done once for the whole language set. Of course, these verses were removed from the training data. A random word from each selected verse was selected. This word-selection was done separately for each language. For each language, we thus get a set of randomly selected words E_l. Though 1209 word-selections were made for each language, many selections happened to select the same word. Thus the size of the E_l-sets varied from 350 (for Maori) to 974 (for Greenlandic). The descrepancy is not disturbing. Words are not entities of the same kind across languages, but our classifier operates on the granularity of words, and the desiderata is an evaluation of 'accuracy per (randomly selected) word'. An alternative, e.g. selecting 1000 unique words of each language would have made interpretation of the result difficult, because for Maori, it is likely that most of the 1000 words would have been *seen* words, occurring in other verses, whereas the opposite is the case for Greenlandic.

If E is a set of tuples (possibly one-word tuples), drawn for language l, we define the accuracy $R_E(l)$ of a language identifier ID:

$$R_E(l) = \frac{|\{\langle x\rangle | ID(\langle x\rangle) = l \text{ and } \langle x\rangle \in E\}|}{|E|}$$

One-word classification: The R_{E_l} was calculated for each of the 32 languages. Since the input sequence is of length 1, there will never be any cuts, so the language identifier was set to output the language with highest probability of having emitted the input word. The E_l-sets as defined above may contain words that are "impossible" predict where they were taken from, on the basis of the word alone. For example, let's say a word w is legal in two languages but much more common in l_1 than l_2. If it happened to be drawn from L_{l_2}, it is hard to see how this can be predicted. However,

we computed figures on the possible influence of this issue, and it turned out to be minor. Therefore, the results in Table 6 stand, but could be adjusted upwards by very small percentages.

Verse classification: To check how accurate the identifier was on longer segments, we chose to test on segments of roughly the length of a verse. Verses, in fact, happen to be around 100 characters long on average. From the 1209 verses selected (as above), those 100 verses thereof whose number of characters were closest to the average verse length of that language, were selected for testing. Denoting these 100-verse sets by V_l, the verse-classification accuracy R_{V_l} was calculated. This score, as well as data on average verse length, can be seen in Table 6.

4-tuple multilingual classification: A set of 1000 mixed language 4-tuples were built from $E_1, \ldots, E_{32}$ as follows.

1. Pick a random language l and pick two random words from that E_l.
2. Precede it with a random word from a random language $E_{l'}$.
3. Add a random word from a random language $E_{l''}$ at the end.

The results of this test was 193 (**19.3%**) fully correctly identified tuples and 204 (**20.4%**) with exactly one word misclassified.

Some figures are low, not surprisingly for languages with a lot of morphology, but overall we hold the results are very reasonable given the exceedingly difficult test problems of one-word and multi-language classification. It is very easy to make mistakes on single words when there are so many languages in the pool – the results are much higher if the number of competing languages is halved.

Unfortunately, we cannot contrast the verse-test with figures from competing state-of-the-art systems, as none of the systems known to us give enough details (on thresholds and such) to reconstruct a fair version of the classifier.

A matter requiring further commentary is the use of a bias function to do the job a scalar threshold value does in related work. (Human language identifiers, having the ability to assess syntactic and semantic coherence, need not use either.) Conceptually the bias function employed is nothing other than a complex system of thresholds, in terms of growth behaviour (exponential, linear etc.) rather than scalar values. Arguably, this is an elegance improvement, although it comes with the cost of being harder to understand, compute and analyse. Also, in the experiments reported above, the bias function approach experimentally outperforms a simple systems of scalar threshold values. For example, through supervised training we have tried tuning one single threshold value for all experiments, one threshold value individually for each language, different threshold values for different classification tasks (i.e. one for multi-language classification and one for single language classification) and so on, resulting in generally lower accuracy on the same test set (obviously, there is little room for presenting and discussing figures from these tests here). Nevertheless, it remains possible that some other, yet undiscovered, system of scalar thresholds is superior to the bias function.

Table 6: Accuracies for the one-word and verse tests plus average verse length in characters ($\overline{V}$).

Language	1-word	Verse	$\overline{V}$
Haitian Creole	0.839	1.00	101.79
Zarma	0.781	1.00	99.45
Kekchi	0.720	1.00	148.78
English	0.678	1.00	104.19
Maori	0.665	1.00	107.73
Hindi	0.607	1.00	119.50
Hausa	0.605	1.00	94.10
Afrikaans	0.594	1.00	103.34
Danish	0.580	1.00	89.30
Cebuano	0.573	1.00	129.48
Icelandic	0.550	1.00	95.58
Swedish	0.547	1.00	107.20
Adamawa Fulfulde	0.539	1.00	96.57
German	0.533	1.00	103.52
Albanian	0.523	1.00	114.80
Spanish	0.511	1.00	95.83
French	0.507	1.00	101.83
Swahili	0.494	1.00	105.03
Slovene	0.488	1.00	100.12
Polish	0.487	1.00	144.52
Portuguese	0.481	1.00	98.41
Esperanto	0.473	1.00	97.80
Italian	0.473	1.00	116.80
Catalan	0.450	1.00	109.70
Dutch	0.415	1.00	109.36
Lithuanian	0.396	1.00	104.99
Hungarian	0.386	1.00	102.10
Latin	0.366	0.99	112.54
Turkish	0.348	0.95	93.43
Finnish	0.345	0.99	107.88
Malayalam	0.276	0.88	128.65
Greenlandic	0.222	0.87	126.99

6. CONCLUSIONS

We have described a new model with considerable elegance for language identification on small, possibly mixed languages segments. We have also added significantly to the set of published evaluations of a language identification system with a balanced cross-language test. For larger input texts the new model has excellent accuracy, but it is bigger and slower in practice than the existing state-of-the-art systems.

7. REFERENCES

[1] C. Biemann and S. Teresniak. Disentangling from babylonian confusion - unsupervised language identification. In A. F. Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing, 6th International Conference, CICLing 2005, Mexico City, Mexico, February 13-19, 2005, Proceedings*, volume 3406 of *Lecture Notes in Computer Science*, pages 773–784. Springer, 2005.

[2] D. Caseiro. Automatic language identification bibliography. `http://www.phys.uni.torun.pl/kmk/projects/ali-bib.html` accessed the 25th of May 2005., 1999.

[3] W. B. Cavnar and J. M. Trenkle. N-gram-based text categorization. In *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175, Las Vegas, US, 1994.

[4] J. F. da Silva and G. P. Lopes. Identification of

document language is not yet a completely solved problem. In *CIMCA '06: Proceedings of the International Conference on Computational Inteligence for Modelling Control and Automation and International Conference on Intelligent Agents Web Technologies and International Commerce*, pages 212–219, Washington, DC, USA, 2006. IEEE Computer Society.

[5] J. F. da Silva and J. G. P. Lopes. Identification of document language in hard contexts. In *Proceedings of the SIGIR 2006 Workshop on New Directions in Multilingual Information Access, Seattle, USA*, 2006.

[6] M. Damashek. Gauging Similarity with n-Grams: Language-Independent Categorization of Text. *Science*, 267(5199):843–848, 1995.

[7] M. S. Dryer. Prefixing versus suffixing in inflectional morphology. In B. Comrie, M. S. Dryer, D. Gil, and M. Haspelmath, editors, *World Atlas of Language Structures*, pages 110–113. Oxford University Press, 2005.

[8] T. Dunning. Statistical identification of language. Technical report, Techical Report MCCS-94-273, Computing Research Lab (CRL), New Mexico State University, 1994.

[9] G. Grefenstette. Comparing two language identification schemes. In S. Bolasco, L. Lebart, and A. Salem, editors, *The proceedings of 3rd International Conference on Statistical Analysis of Textual Data (JADT 95), Rome, Italy, Dec. 1995*, 1995.

[10] H. Hammarström. A naive theory of morphology and an algorithm for extraction. In R. Wicentowski and G. Kondrak, editors, *SIGPHON 2006: Eighth Meeting of the Proceedings of the ACL Special Interest Group on Computational Phonology, 8 June 2006, New York City, USA*, pages 79–88. Association for Computational Linguistics, 2006. `http://www.cs.chalmers.se/~harald2/sigphon06.pdf`.

[11] B. Hughes, T. Baldwin, S. Bird, J. Nicholson, and A. MacKinlay. Reconsidering language identification for written language resources. In *Proceedings 5th International Conference on Language Resources and Evaluation (LREC2006)*, pages 485–488. Genoa, Italy, 2006.

[12] P. Juola. Language identification, automatic. In K. Brown, editor, *Encyclopedia of Language and Linguistics*, volume 6, pages 508–510. Elsevier, Amsterdam, 2 edition, 2006.

[13] C. Kruengkrai, V. Srichaivattana, P. andSornlertlamvanich, and H. Isahara. Language identification based on string kernels. In *IEEE International Symposium on Communications and Information Technology, 2005. ISCIT 2005*, volume 2, pages 926–929, 2005.

[14] R. D. Lins and P. Gonçalves, Jr. Automatic language identification of written texts. In *SAC '04: Proceedings of the 2004 ACM symposium on Applied computing*, pages 1128–1133, New York, NY, USA, 2004. ACM Press.

[15] T. Martin, B. Baker, E. Wong, and S. Sridharan. A syllable-scale framework for language identification. *Computer Speech & Language*, 20(2-3):276–302, 2006.

[16] B. Martins and M. J. Silva. Language identification in web pages. In *SAC '05: Proceedings of the 2005 ACM symposium on Applied computing*, pages 764–768, New York, NY, USA, 2005. ACM Press.

[17] P. McNamee. Language identification: a solved problem suitable for undergraduate instruction. *Journal of Computing Sciences in Colleges*, 20(3):94–101, 2005.

[18] K. N. Murthy and G. B. Kumar. Language identification from small text samples. *Journal of Quantitative Linguistics*, 13(1):57–80, 2006.

[19] Y. K. Muthusamy and L. A. Spitz. Automatic language identification. In R. A. Cole, editor, *Survey of the State of the Art in Human Language Technology*, chapter 8.7. Center for Spoken Language Understanding CSLU, Carnegie Mellon University, Pittsburgh, PA, 1997.

[20] A. Poutsma. Applying monte carlo techniques to language identification. In T. Mariët, A. Nijholt, and H. Hondorp, editors, *Computational Linguistics in the Netherlands 2001: Selected Papers from the Twelfth CLIN Meeting*, volume 45 of *Language and Computers - Studies in Practical Linguistics*, pages 179–189. Rodopi, Amsterdam/New York, NY, 2002.

[21] J. M. Prager. Linguini: Language identification for multilingual documents. *Journal of Management Information Systems*, 16(3):71–102, 2000.

[22] P. Sibun and J. C. Reynar. Language identification: Examining the issues. In *5th Symposium on Document Analysis and Information Retrieval*, pages 125–135, Las Vegas, Nevada, U.S.A., 1996.

[23] H. Takci and I. Sogukpinar. Centroid-based language identification using letter feature set. In A. Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing: 5th International Conference, CICLing 2004 Seoul, Korea, February 15-21, 2004 Proceedings*, volume 2945 of *Lecture Notes in Computer Science*, pages 640–648. Springer-Verlag, Berlin, 2004.

[24] A. Xafopoulos, C. Kotropoulos, G. Almpanidis, and I. Pitas. Language identification in web documents using discrete HMMs. *Pattern Recognition*, 37(3):583–594(12), 2004.

[25] D.-V. Ziegler. *The Automatic Identification of Languages Using Linguistic Recognition Signals*. PhD thesis, University of New York at Buffalo, 1991.