Language Identification: The Long and the Short of the Matter

Timothy Baldwin and **Marco Lui**

Dept of Computer Science and Software Engineering
University of Melbourne, VIC 3010 Australia
tb@ldwin.net, saffsd@gmail.com

Abstract

Language identification is the task of identifying the language a given document is written in. This paper describes a detailed examination of what models perform best under different conditions, based on experiments across three separate datasets and a range of tokenisation strategies. We demonstrate that the task becomes increasingly difficult as we increase the number of languages, reduce the amount of training data and reduce the length of documents. We also show that it is possible to perform language identification without having to perform explicit character encoding detection.

1 Introduction

With the growth of the worldwide web, ever-increasing numbers of documents have become available, in more and more languages. This growth has been a double-edged sword, however, in that content in a given language has become more prevalent but increasingly hard to find, due to the web's sheer size and diversity of content. While the majority of (X)HTML documents declare their character encoding, only a tiny minority specify what language they are written in, despite support for language declaration existing in the various (X)HTML standards.[1] Additionally, a single encoding can generally be used to render a large number of languages such that the document encoding at best filters out a subset of languages which are incompatible with the given encoding, rather than disambiguates the source language. Given this, the need for automatic means to determine the source language of web documents is crucial for web aggregators of various types.

There is widespread misconception of language identification being a "solved task", generally as a result of isolated experiments over homogeneous datasets with small numbers of languages (Hughes et al., 2006; Xia et al., 2009). Part of the motivation for this paper is to draw attention to the fact that, as a field, we are still a long way off perfect language identification of web documents, as evaluated under realistic conditions.

In this paper we describe experiments on language identification of web documents, focusing on the broad question of what combination of tokenisation strategy and classification model achieves the best overall performance. We additionally evaluate the impact of the volume of training data and the test document length on the accuracy of language identification, and investigate the interaction between character encoding detection and language identification.

One assumption we make in this research, following standard assumptions made in the field, is that all documents are monolingual. This is clearly an unrealistic assumption when dealing with general web documents (Hughes et al., 2006), and we plan to return to investigate language identification over multilingual documents in future work.

Our contributions in this paper are: the demonstration that language identification is: (a) trivial over datasets with smaller numbers of languages and approximately even amounts of training data per language, but (b) considerably harder over datasets with larger numbers of languages with more skew in the amount of training data per language; byte-based tokenisation without character encoding detection is superior to codepoint-based tokenisation

[1] http://dev.opera.com/articles/view/mama-head-structure/

Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the ACL, pages 229–237,
Los Angeles, California, June 2010.

with character encoding detection; and simple cosine similarity-based nearest neighbour classification is equal to or better than models including support vector machines and naive Bayes over the language identification task. We also develop datasets to facilitate standardised evaluation of language identification.

2 Background Research

Language identification was arguably established as a task by Gold (1967), who construed it as a closed class problem: given data in each of a predefined set of possible languages, human subjects were asked to classify the language of a given test document. It wasn't until the 1990s, however, that the task was popularised as a text categorisation task.

The text categorisation approach to language identification applies a standard supervised classification framework to the task. Perhaps the best-known such model is that of Cavnar and Trenkle (1994), as popularised in the `textcat` tool.[2] The method uses a per-language character frequency model, and classifies documents via their relative "out of place" distance from each language (see Section 5.1). Variants on this basic method include Bayesian models for character sequence prediction (Dunning, 1994), dot products of word frequency vectors (Darnashek, 1995) and information-theoretic measures of document similarity (Aslam and Frost, 2003; Martins and Silva, 2005). More recently, support vector machines (SVMs) and kernel methods have been applied to the task of language identification task with success (Teytaud and Jalam, 2001; Lodhi et al., 2002; Kruengkrai et al., 2005), and Markov logic has been used for joint inferencing in contexts where there are multiple evidence sources (Xia et al., 2009).

Language identification has also been carried out via linguistically motivated models. Johnson (1993) used a list of stop words from different languages to identify the language of a given document, choosing the language with the highest stop word overlap with the document. Grefenstette (1995) used word and part of speech (POS) correlation to determine if two text samples were from the same or different languages. Giguet (1995) developed a cross-language tokenisation model and used it to identify the language of a given document based on its tokenisation similarity with training data. Dueire Lins and Gonçalves (2004) considered the use of syntactically-derived closed grammatical-class models, matching syntactic structure rather than words or character sequences.

The observant reader will have noticed that some of the above approaches make use of notions such as "word", typically based on the naive assumption that the language uses white space to delimit words. These approaches are appropriate in contexts where there is a guarantee of a document being in one of a select set of languages where words are space-delimited, or where manual segmentation has been performed (e.g. interlinear glossed text). However, we are interested in language identification of web documents, which can be in any language, including languages that do not overtly mark word boundaries, such as Japanese, Chinese and Thai; while relatively few languages fall into this categories, they are among the most populous web languages and therefore an important consideration. Therefore, approaches that assume a language is space-delimited are clearly not suitable for our purposes. Equally, approaches which make assumptions about the availability of particular resources for each language to be identified (e.g. POS taggers, or the existence of precompiled stop word lists) cannot be used.

Language identification has been applied in a number of contexts, the most immediate application being in multilingual text retrieval, where retrieval results are generally superior if the language of the query is known, and the search is restricted to only those documents predicted to be in that language (McNamee and Mayfield, 2004). It can also be used to "word spot" foreign language terms in multilingual documents, e.g. to improve parsing performance (Alex et al., 2007), or for linguistic corpus creation purposes (Baldwin et al., 2006; Xia et al., 2009; Xia and Lewis, 2009).

3 Datasets

In the experiments reported in this paper, we employ three novel datasets, with differing properties relevant to language identification research:

[2] `http://www.let.rug.nl/vannoord/TextCat/`

Corpus	Documents	Languages	Encodings	Document Length (bytes)
EUROGOV	1500	10	1	17460.5±39353.4
TCL	3174	60	12	2623.2±3751.9
WIKIPEDIA	4963	67	1	1480.8±4063.9

Table 1: Summary of the three language identification datasets

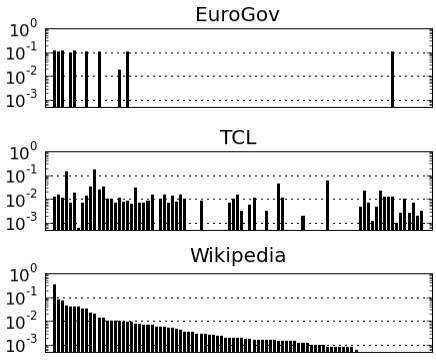

Figure 1: Distribution of languages in the three datasets (vector of languages vs. the proportion of documents in that language)

EUROGOV: longer documents, all in a single encoding, spread evenly across a relatively small number (10) of Western European languages; this dataset is comparable to the datasets conventionally used in language identification research. As the name would suggest, the documents were sourced from the EuroGOV document collection, as used in the 2005 WebCLEF task.

TCL: a larger number of languages (60) across a wider range of language families, with shorter documents and a range of character encodings (12). The collection was manually sourced by the Thai Computational Linguistics Laboratory (TCL) in 2005 from online news sources.

WIKIPEDIA: a slightly larger number of languages again (67), a single encoding, and shorter documents; the distribution of languages is intended to approximate that of the actual web. This collection was automatically constructed by taking the dumps of all versions of Wikipedia with 1000 or more documents in non-constructed languages, and randomly selecting documents from them in a bias-preserving manner (i.e. preserving the document distribution in the full collection); this is intended to represent the document language bias observed on the web. All three corpora are available on request.

We outline the characteristics of the three datasets in Table 1. We further detail the language distribution in Figure 1, using a constant vector of languages for all three datasets, based on the order of languages in the WIKIPEDIA dataset (in descending order of documents per language). Of note are the contrasting language distributions between the three datasets, in terms of both the languages represented and the relative skew of documents per language. In the following sections, we provide details of the corpus compilation and document sampling method for each dataset.

4 Document Representation

As we are interested in performing language identification over arbitrary web documents, we require a language-neutral document representation which does not make artificial assumptions about the source language of the document. Separately, there is the question of whether it is necessary to determine the character encoding of the document in order to extract out character sequences, or whether the raw byte stream is sufficient. To explore this question, we experiment with two document representations: (1) byte n-grams, and (2) codepoint n-grams. In both cases, a document is represented as a feature vector of token counts.

Byte n-grams can be extracted directly without explicit encoding detection. Codepoint n-grams, on the other hand, require that we know the character encoding of the document in order to perform tokenisation. Additionally, they should be based on a common encoding to prevent: (a) over-fragmenting the feature space (e.g. ending up with discrete feature spaces for `euc-jp`, `s-jis` and `utf-8` in the case of Japanese); and (b) spurious matches between encodings (e.g. Japanese hiragana and Korean hangul mapping onto the same codepoint in `euc-jp` and `euc-kr`, respectively). We use uni-

code as the common encoding for all documents.

In practice, character encoding detection is an issue only for TCL, as the other two datasets are in a single encoding. Where a character encoding was provided for a document in TCL and it was possible to transcode the document to unicode based on that encoding, we used the encoding information. In cases where a unique encoding was not provided, we used an encoding detection library based on the Mozilla browser.[3] Having disambiguated the encoding for each document, we transcoded it into unicode.

5 Models

In our experiments we use a number of different language identification models, as outlined below. We first describe the nearest-neighbour and nearest-prototype models, and a selection of distance and similarity metrics combined with each. We then present three standalone text categorisation models.

5.1 Nearest-Neighbour and Nearest-Prototype Models

The 1-nearest-neighbour (1NN) model is a common classification technique, whereby a test document D is classified based on the language of the closest training document D_i (with language $l(D_i)$), as determined by a given distance or similarity metric.

In nearest-neighbour models, each training document is represented as a single instance, meaning that the computational cost of classifying a test document is proportional to the number of training documents. A related model which aims to reduce this cost is nearest-prototype (AM), where each language is represented as a single instance, by merging all of the training instances for that language into a single centroid via the arithmetic mean.

For both nearest-neighbour and nearest-prototype methods, we experimented with three similarity and distance measures in this research:

Cosine similarity (COS): the cosine of the angle between two feature vectors, as measured by the dot product of the two vectors, normalised to unit length.

Skew divergence (SKEW): a variant of Kullback-Leibler divergence, whereby the second distribution (y) is smoothed by linear interpolation with the first (x) using a smoothing factor α (Lee, 2001):

$$s_\alpha(x, y) = D(x \mid\mid \alpha y + (1 - \alpha)x)$$

where:

$$D(x \mid\mid y) = \sum_i x_i(\log_2 x_i - \log_2 y_i)$$

In all our experiments, we set α to 0.99.

Out-of-place (OOP): a ranklist-based distance metric, where the distance between two documents is calculated as (Cavnar and Trenkle, 1994):

$$oop(D_x, D_y) = \sum_{t \in D_x \vee D_y} abs(R_{D_x}(t) - R_{D_y}(t))$$

$R_D(t)$ is the rank of term t in document D, based on the descending order of frequency in document D; terms not occurring in document D are conventionally given the rank $1 + \max_i R_D(t_i)$.

5.2 Naive Bayes (NB)

Naive Bayes is a popular text classification model, due to it being lightweight, robust and easy to update. The language of test document D is predicted by:

$$\hat{l}(D) = \arg\max_{l_i \in L} P(l_i) \prod_{j=1}^{|\mathscr{V}|} \frac{P(t_j|l_i)^{N_{D,t_j}}}{N_{D,t_j}!}$$

where L is the set of languages in the training data, N_{D,t_j} is the frequency of the jth term in D, $\mathscr{V}$ is the set of all terms, and:

$$P(t|l_i) = \frac{1 + \sum_{k=1}^{|\mathscr{D}|} N_{k,t} P(l_i|D_k)}{|\mathscr{V}| + \sum_{j=1}^{|\mathscr{V}|} \sum_{k=1}^{|\mathscr{D}|} N_{k,t_j} P(l_i|D_k)}$$

In this research, we use the `rainbow` implementation of multinominal naive Bayes (McCallum, 1996).

5.3 Support Vector Machines (SVM)

Support vector machines (SVMs) are one of the most popular methods for text classification, largely because they can automatically weight large numbers of features, capturing feature interactions in the process (Joachims, 1998; Manning et al., 2008). The basic principle underlying SVMs is to maximize the

[3] `http://chardet.feedparser.org/`

margin between training instances and the calculated decision boundary based on structural risk minimisation (Vapnik, 1995).

In this work, we have made use of `bsvm`,[4] an implementation of SVMs with multiclass classification support (Hsu et al., 2008). We only report results for multi-class bound-constrained support vector machines with linear kernels, as they were found to perform best over our data.

6 Experimental Methodology

We carry out experiments over the cross-product of the following options, as described above:

model (×7): nearest-neighbour ($\text{COS}_{1\text{NN}}$, $\text{SKEW}_{1\text{NN}}$, $\text{OOP}_{1\text{NN}}$), nearest-prototype (COS_{AM}, SKEW_{AM}),[5] NB, SVM

tokenisation (×2): byte, codepoint

n-gram (×3): 1-gram, 2-gram, 3-gram

for a total of 42 distinct classifiers. Each classifier is run across the 3 datasets (EUROGOV, TCL and WIKIPEDIA) based on 10-fold stratified cross-validation.

We evaluate the models using micro-averaged precision ($\mathcal{P}_\mu$), recall ($\mathcal{R}_\mu$) and F-score ($\mathcal{F}_\mu$), as well as macro-averaged precision ($\mathcal{P}_M$), recall ($\mathcal{R}_M$) and F-score ($\mathcal{F}_M$). The micro-averaged scores indicate the average performance *per document*; as we always make a unique prediction per document, the micro-averaged precision, recall and F-score are always identical (as is the classification accuracy). The macro-averaged scores, on the other hand, indicate the average performance *per language*. In each case, we average the precision, recall and F-score across the 10 folds of cross validation.[6]

As a baseline, we use a majority class, or ZeroR, classifier (ZEROR), which assigns the language with highest prior in the training data to each of the test documents.

[4] `http://www.csie.ntu.edu.tw/~cjlin/bsvm/`

[5] We do not include the results for nearest-prototype classifiers with the OOP distance metric as the results were considerably lower than the other methods.

[6] Note that this means that the averaged $\mathcal{F}_M$ is not necessarily the harmonic mean of the averaged $\mathcal{P}_M$ and $\mathcal{R}_M$.

Model	Token	$\mathcal{P}_M$	$\mathcal{R}_M$	$\mathcal{F}_M$	$\mathcal{P}_\mu/\mathcal{R}_\mu/\mathcal{F}_\mu$
ZEROR	—	.020	.084	.032	.100
$\text{COS}_{1\text{NN}}$	byte	.975	.978	.976	.975
$\text{COS}_{1\text{NN}}$	codepoint	.968	.973	.970	.971
COS_{AM}	byte	.922	.938	.926	.937
COS_{AM}	codepoint	.908	.930	.913	.931
$\text{SKEW}_{1\text{NN}}$	byte	.979	.979	.979	.977
$\text{SKEW}_{1\text{NN}}$	codepoint	.978	.978	.978	.976
SKEW_{AM}	byte	.974	.972	.972	.969
SKEW_{AM}	codepoint	.974	.972	.973	.970
$\text{OOP}_{1\text{NN}}$	byte	.953	.952	.953	.949
$\text{OOP}_{1\text{NN}}$	codepoint	.961	.960	.960	.957
NB	byte	.975	.973	.974	.971
NB	codepoint	.975	.973	.974	.971
SVM	byte	**.989**	**.985**	**.987**	**.987**
SVM	codepoint	.988	.985	.986	.987

Table 2: Results for byte vs. codepoint (bigram) tokenisation over EUROGOV

7 Results

In our experiments, we first compare the different models for fixed n-gram order, then come back to vary the n-gram order. Subsequently, we examine the relative performance of the different models on test documents of differing lengths, and finally look into the impact of the amount of training data for a given language on the performance for that language.

7.1 Results for the Different Models and Tokenisation Strategies

First, we present the results for each of the classifiers in Tables 2–4, based on byte or codepoint tokenisation and bigrams. In each case, we present the best result in each column in **bold**.

The relative performance over EUROGOV and TCL is roughly comparable for all methods barring $\text{SKEW}_{1\text{NN}}$, with near-perfect scores over all 6 evaluation metrics. $\text{SKEW}_{1\text{NN}}$ is near-perfect over EUROGOV and TCL, but drops to baseline levels over WIKIPEDIA; we return to discuss this effect in Section 7.2.

In the case of EUROGOV, the near-perfect results are in line with our expectations for the dataset, based on its characteristics and results reported for comparable datasets. The results for WIKIPEDIA, however, fall off considerably, with the best model achieving an $\mathcal{F}_M$ of .671 and $\mathcal{F}_\mu$ of .869, due to

Model	Token	$\mathcal{P}_M$	$\mathcal{R}_M$	$\mathcal{F}_M$	$\mathcal{P}_\mu/\mathcal{R}_\mu/\mathcal{F}_\mu$
ZEROR	—	.003	.017	.005	.173
COS_{1NN}	byte	.981	.975	.975	.982
COS_{1NN}	codepoint	.931	.930	.925	.961
COS_{AM}	byte	.967	.975	.965	.965
COS_{AM}	codepoint	.979	**.977**	.974	.964
SKEW_{1NN}	byte	**.984**	.974	.976	**.987**
SKEW_{1NN}	codepoint	.910	.210	.320	.337
SKEW_{AM}	byte	.962	.959	.950	.972
SKEW_{AM}	codepoint	.968	.961	.957	.967
OOP_{1NN}	byte	.964	.945	.951	.974
OOP_{1NN}	codepoint	.901	.892	.893	.933
NB	byte	.905	.905	.896	.969
NB	codepoint	.722	.711	.696	.845
SVM	byte	.981	.973	**.977**	.984
SVM	codepoint	.979	.970	.974	.980

Table 3: Results for byte vs. codepoint (bigram) tokenisation over TCL

Model	Token	$\mathcal{P}_M$	$\mathcal{R}_M$	$\mathcal{F}_M$	$\mathcal{P}_\mu/\mathcal{R}_\mu/\mathcal{F}_\mu$
ZEROR	—	.004	.013	.007	.328
COS_{1NN}	byte	**.740**	**.646**	**.671**	**.869**
COS_{1NN}	codepoint	.685	.604	.625	.835
COS_{AM}	byte	.587	.634	.573	.776
COS_{AM}	codepoint	.486	.556	.483	.725
SKEW_{1NN}	byte	.005	.013	.008	.304
SKEW_{1NN}	codepoint	.006	.013	.007	.241
SKEW_{AM}	byte	.605	.617	.588	.844
SKEW_{AM}	codepoint	.552	.575	.532	.807
OOP_{1NN}	byte	.619	.518	.548	.831
OOP_{1NN}	codepoint	.598	.486	.520	.807
NB	byte	.496	.454	.442	.851
NB	codepoint	.426	.349	.360	.798
SVM	byte	.667	.545	.577	.845
SVM	codepoint	.634	.494	.536	.818

Table 4: Results for byte vs. codepoint (bigram) tokenisation over WIKIPEDIA

the larger number of languages, smaller documents, and skew in the amounts of training data per language. All models are roughly balanced in the relative scores they attain for $\mathcal{P}_M$, $\mathcal{R}_M$ and $\mathcal{F}_M$ (i.e. there are no models that have notably higher $\mathcal{P}_M$ relative to $\mathcal{R}_M$, for example).

The nearest-neighbour models outperform the corresponding nearest-prototype models to varying degrees, with the one exception of SKEW_{1NN} over WIKIPEDIA. The nearest-prototype classifiers were certainly faster than the nearest-neighbour classifiers, by roughly an order of 10, but this is more than outweighed by the drop in classification performance. With the exception of SKEW_{1NN} over WIKIPEDIA, all methods were well above the baselines for all three datasets.

The two methods which perform consistently well at this point are COS_{1NN} and SVM, with COS_{1NN} holding up particularly well under micro-averaged F-score while NB drops away over WIKIPEDIA, the most skewed dataset; this is due to the biasing effect of the prior in NB.

Looking to the impact of byte- vs. codepoint-tokenisation on classifier performance over the three datasets, we find that overall, bytes outperform codepoints. This is most notable for TCL and WIKIPEDIA, and the SKEW_{1NN} and NB models. Given this result, we present only results for byte-based tokenisation in the remainder of this paper.

The results for byte tokenisation of TCL are particularly noteworthy. The transcoding into unicode and use of codepoints, if anything, hurts performance, suggesting that implicit character encoding detection based on byte tokenisation is the best approach: it is both more accurate and simplifies the system, in removing the need to perform encoding detection prior to language identification.

7.2 Results for Differing n-gram Sizes

We present results with byte unigrams, bigrams and trigrams in Table 5 for WIKIPEDIA.[7] We omit results for the other two datasets, as the overall trend is the same as for WIKIPEDIA, with lessened relative differences between n-gram orders due to the relative simplicity of the respective classification tasks.

SKEW_{1NN} is markedly different to the other methods in achieving the best performance with unigrams, moving from the worst-performing method by far to one of the best-performing methods. This is the result of the interaction between data sparseness and heavy-handed smoothing with the α constant. Rather than using a constant α value for all n-gram orders, it may be better to parameterise it using an exponential scale such as $\alpha = 1 - \beta^n$ (with

[7] The results for OOP_{1NN} over byte trigrams are missing due to the computational cost associated with the method, and our experiment hence not having run to completion at the time of writing. Extrapolating from the results for the other two datasets, we predict similar results to bigrams.

Model	n-gram	$\mathcal{P}_M$	$\mathcal{R}_M$	$\mathcal{F}_M$	$\mathcal{P}_\mu/\mathcal{R}_\mu/\mathcal{F}_\mu$
ZEROR	—	.004	.013	.007	.328
COS_{1NN}	1	.644	.579	.599	.816
COS_{1NN}	2	.740	.646	.671	**.869**
COS_{1NN}	3	**.744**	**.656**	**.680**	.862
COS_{AM}	1	.526	.543	.487	.654
COS_{AM}	2	.587	.634	.573	.776
COS_{AM}	3	.553	.632	.545	.761
SKEW_{1NN}	1	.691	.598	.625	.848
SKEW_{1NN}	2	.005	.013	.008	.304
SKEW_{1NN}	3	.005	.013	.004	.100
SKEW_{AM}	1	.552	.569	.532	.740
SKEW_{AM}	2	.605	.617	.588	.844
SKEW_{AM}	3	.551	.631	.554	.825
OOP_{1NN}	1	.519	.446	.468	.747
OOP_{1NN}	2	.619	.518	.548	.831
NB	1	.576	.578	.555	.778
NB	2	.496	.454	.442	.851
NB	3	.493	.435	.432	.863
SVM	1	.585	.505	.523	.812
SVM	2	.667	.545	.577	.845
SVM	3	.717	.547	.594	.840

Table 5: Results for different n-gram orders over WIKIPEDIA

$\beta = 0.01$, e.g.), based on the n-gram order. We leave this for future research.

For most methods, bigrams and trigrams are better than unigrams, with the one notable exception of SKEW_{1NN}. In general, there is little separating bigrams and trigrams, although the best result for is achieved slightly more often for bigrams than for trigrams.

For direct comparability with Cavnar and Trenkle (1994), we additionally carried out a preliminary experiment with hybrid byte n-grams (all of 1- to 5-grams), combined with simple frequency-based feature selection of the top-1000 features for each n-gram order. The significance of this setting is that it is the strategy adopted by `textcat`, based on the original paper of Cavnar and Trenkle (1994) (with the one exception that we use 1000 features rather than 300, as all methods other than OOP_{1NN} benefitted from more features). The results are shown in Table 6.

Compared to the results in Table 5, SKEW_{1NN} and SKEW_{AM} both increase markedly to achieve the best overall results. OOP_{1NN}, on the other hand, rises slightly, while the remaining three methods actually drop back slightly. Clearly, there is considerably more experimentation to be done here with mixed n-gram models and different feature selection methods, but the results indicate that some methods certainly benefit from n-gram hybridisation and feature selection, and also that we have been able to surpass the results of Cavnar and Trenkle (1994) with SKEW_{1NN} in an otherwise identical framework.

Model	$\mathcal{P}_M$	$\mathcal{R}_M$	$\mathcal{F}_M$	$\mathcal{P}_\mu/\mathcal{R}_\mu/\mathcal{F}_\mu$
ZEROR	.004	.013	.007	.328
COS_{1NN}	.735	.664	.682	.865
COS_{AM}	.592	.626	.580	.766
SKEW_{1NN}	**.789**	.708	**.729**	**.902**
SKEW_{AM}	.681	**.718**	.680	.870
OOP_{1NN}	.697	.595	.626	.864
SVM	.669	.500	.544	.832

Table 6: Results for mixed n-grams (1–5) and feature selection over WIKIPEDIA (a lá Cavnar and Trenkle (1994))

7.3 Breakdown Across Test Document Length

To better understand the impact of test document size on classification accuracy, we divided the test documents into 5 equal-size bins according to their length, measured by the number of tokens. We then computed $\mathcal{F}_\mu$ individually for each bin across the 10 folds of cross validation. We present the breakdown of results for WIKIPEDIA in Figure 2.

WIKIPEDIA shows a pseudo-logarithmic growth in $\mathcal{F}_\mu$ (= $\mathcal{P}_\mu$ = $\mathcal{R}_\mu$) as the test document size increases. This fits with our intuition, as the model has progressively more evidence to base the classification on. It also suggests that performance over shorter documents appears to be the dominating factor in the overall ranking of the different methods. In particular, COS_{1NN} and SVM appear to be able to classify shorter documents most reliably, leading to the overall result of them being the best-performing methods.

While we do not show the graph for reasons of space, the equivalent graph for EUROGOV displays a curious effect: $\mathcal{F}_\mu$ drops off as the test documents get longer. Error analysis of the data indicates that this is due to longer documents being more likely to be "contaminated" with either data from a second language or extra-linguistic data, such as large tables of numbers or chemical names. This suggests that all the models are brittle when the assump-

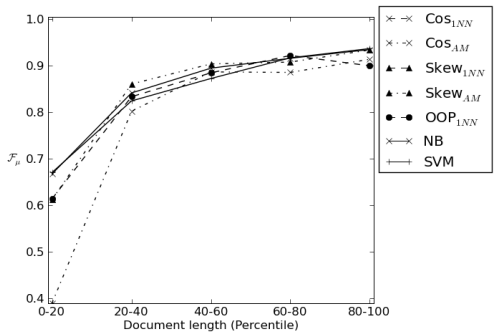

Figure 2: Breakdown of $\mathcal{F}_\mu$ over WIKIPEDIA for test documents of increasing length

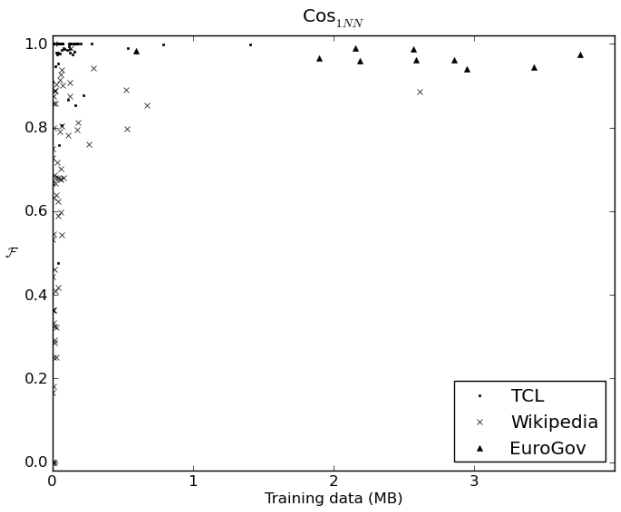

Figure 3: Per-language $\mathcal{F}_M$ for $\text{COS}_{1\text{NN}}$, relative to the training data size (in MB) for that language

tion of strict monolingualism is broken, or when the document is dominated by extra-linguistic data. Clearly, this underlines our assumption of monolingual documents, and suggests multilingual language identification is a fertile research area even in terms of optimising performance over our "monolingual" datasets.

7.4 Performance Relative to Training Data Size

As a final data point in our analysis, we calculated the $\mathcal{F}_M$ for each language relative to the amount of training data available for that language, and present the results in the form of a combined scatter plot for the three datasets in Figure 3. The differing distributions of the three datasets are self-evident, with most languages in EUROGOV (the squares) both having reasonably large amounts of training data and achieving high $\mathcal{F}_M$ values, but the majority of languages in WIKIPEDIA (the crosses) having very little data (including a number of languages with no training data, as there is a singleton document in that language in the dataset). As an overall trend, we can observe that the greater the volume of training data, the higher the $\mathcal{F}_M$ across all three datasets, but there is considerable variation between the languages in terms of their $\mathcal{F}_M$ for a given training data size (the column of crosses for WIKIPEDIA to the left of the graph is particularly striking).

8 Conclusions

We have carried out a thorough (re)examination of the task of language identification, that is predicting the language that a given document is written in, focusing on monolingual documents at present. We experimented with a total of 7 models, and tested each over two tokenisation strategies (bigrams vs. codepoints) and three token n-gram orders (unigrams, bigrams and trigrams). At the same time as reproducing results from earlier research on how easy the task can be over small numbers of languages with longer documents, we demonstrated that the task becomes much harder for larger numbers of languages, shorter documents and greater class skew. We also found that explicit character encoding detection is not necessary in language detection, and that the most consistent model overall is either a simple 1-NN model with cosine similarity, or an SVM with a linear kernel, using a byte bigram or trigram document representation. We also confirmed that longer documents tend to be easier to classify, but also that multilingual documents cause problems for the standard model of language identification.

Acknowledgements

This research was supported by a Google Research Award.

References

Beatrice Alex, Amit Dubey, and Frank Keller. 2007. Using foreign inclusion detection to improve parsing performance. In *Proceedings of the Joint Conference*

on Empirical Methods in Natural Language Processing and Computational Natural Language Learning 2007 (EMNLP-CoNLL 2007), pages 151–160, Prague, Czech Republic.

Javed A. Aslam and Meredith Frost. 2003. An information-theoretic measure for document similarity. In *Proceedings of 26th International ACM-SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2003)*, pages 449–450, Toronto, Canada.

Timothy Baldwin, Steven Bird, and Baden Hughes. 2006. Collecting low-density language materials on the web. In *Proceedings of the 12th Australasian Web Conference (AusWeb06)*. `http://www.ausweb.scu.edu.au/ausweb06/edited/hughes/`.

William B. Cavnar and John M. Trenkle. 1994. N-gram-based text categorization. In *Proceedings of the Third Symposium on Document Analysis and Information Retrieval*, Las Vegas, USA.

Marc Darnashek. 1995. Gauging similarity with n-grams: Language-independent categorization of text. *Science*, 267:843–848.

Rafael Dueire Lins and Paulo Gonçalves. 2004. Automatic language identification of written texts. In *Proceedings of the 2004 ACM Symposium on Applied Computing (SAC 2004)*, pages 1128–1133, Nicosia, Cyprus.

Ted Dunning. 1994. Statistical identification of language. Technical Report MCCS 940-273, Computing Research Laboratory, New Mexico State University.

Emmanuel Giguet. 1995. Categorization according to language: A step toward combining linguistic knowledge and statistic learning. In *Proceedings of the 4th International Workshop on Parsing Technologies (IWPT-1995)*, Prague, Czech Republic.

E. Mark Gold. 1967. Language identification in the limit. *Information and Control*, 5:447–474.

Gregory Grefenstette. 1995. Comparing two language identification schemes. In *Proceedings of Analisi Statistica dei Dati Testuali (JADT)*, pages 263–268.

Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin. 2008. A practical guide to support vector classification. Technical report, Department of Computer Science National Taiwan University.

Baden Hughes, Timothy Baldwin, Steven Bird, Jeremy Nicholson, and Andrew MacKinlay. 2006. Reconsidering language identification for written language resources. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006)*, pages 485–488, Genoa, Italy.

Thorsten Joachims. 1998. Text categorization with support vector machines: learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning*, pages 137–142, Chemnitz, Germany.

Stephen Johnson. 1993. Solving the problem of language recognition. Technical report, School of Computer Studies, University of Leeds.

Canasai Kruengkrai, Prapass Srichaivattana, Virach Sornlertlamvanich, and Hitoshi Isahara. 2005. Language identification based on string kernels. In *Proceedings of the 5th International Symposium on Communications and Information Technologies (ISCIT-2005)*, pages 896–899, Beijing, China.

Lillian Lee. 2001. On the effectiveness of the skew divergence for statistical language analysis. In *Proceedings of Artificial Intelligence and Statistics 2001 (AISTATS 2001)*, pages 65–72, Key West, USA.

Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, and Chris Watkins. 2002. Text classification using string kernels. *Journal of Machine Learning Research*, 2:419–444.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, Cambridge, UK.

Bruno Martins and Mário J. Silva. 2005. Language identification in web pages. In *Proceedings of the 2005 ACM symposium on Applied computing*, pages 764–768, Santa Fe, USA.

Andrew Kachites McCallum. 1996. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. `http://www.cs.cmu.edu/~mccallum/bow`.

Paul McNamee and James Mayfield. 2004. Character N-gram Tokenization for European Language Text Retrieval. *Information Retrieval*, 7(1–2):73–97.

Olivier Teytaud and Radwan Jalam. 2001. Kernel-based text categorization. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN'2001)*, Washington DC, USA.

Vladimir N. Vapnik. 1995. *The Nature of Statistical Learning Theory*. Springer-Verlag, Berlin, Germany.

Fei Xia and William Lewis. 2009. Applying NLP technologies to the collection and enrichment of language data on the web to aid linguistic research. In *Proceedings of the EACL 2009 Workshop on Language Technology and Resources for Cultural Heritage, Social Sciences, Humanities, and Education (LaTeCH – SHELT&R 2009)*, pages 51–59, Athens, Greece.

Fei Xia, William Lewis, and Hoifung Poon. 2009. Language ID in the context of harvesting language data off the web. In *Proceedings of the 12th Conference of the EACL (EACL 2009)*, pages 870–878, Athens, Greece.