Cross-domain Feature Selection for Language Identification

Marco Lui and Timothy Baldwin
NICTA VRL
Department of Computer Science and Software Engineering
University of Melbourne, VIC 3010, Australia
saffsd@gmail.com, tb@ldwin.net

Abstract

We show that transductive (cross-domain) learning is an important consideration in building a general-purpose language identification system, and develop a feature selection method that generalizes across domains. Our results demonstrate that our method provides improvements in transductive transfer learning for language identification. We provide an implementation of the method and show that our system is faster than popular standalone language identification systems, while maintaining competitive accuracy.

1 Introduction

Language identification (LangID) is the task of determining the language(s) that a text is written in. It is considered by some researchers to be a solved task, because previous research has reported near-perfect accuracy (Cavnar and Trenkle, 1994). Hughes et al. (2006) elaborated a number of simplifying assumptions that have made this the case, and Baldwin and Lui (2010a) showed that when some of these assumptions are relaxed to make the task closer to the actuality of open-web LangID, it becomes considerably harder. In this paper, we demonstrate that the style of evaluation used by Baldwin and Lui (2010a) performs well in-domain but badly cross-domain, and develop a novel method for preserving high in-domain accuracy, while significantly boosting cross-domain accuracy. Similarly to Baldwin and Lui (2010a), we make the simplifying assumption that all documents are monolingual, despite recent work on multilingual LangID (Baldwin and Lui, 2010b).

LangID is usually formulated as a supervised machine learning problem and evaluated in an offline setting (Cavnar and Trenkle, 1994; Baldwin and Lui, 2010a). However, its primary use is online without any additional configuration, optimized for maximal cross-domain accuracy. A number of such standalone LangID systems are available, notable among which is `TextCat` (van Noord, 1997). `TextCat` has been the LangID solution of choice in research, and is the basis of language identification/filtering in the ClueWeb09 Dataset (Callan and Hoy, 2009) and Corpus-Builder (Ghani et al., 2004). Elsewhere, Google provides LangID as a web service via its Google Language Detect API (`GoogleAPI`). While it has much higher accuracy than `TextCat` (as we show in Section 6.1), research applications contravene the service's terms of use, and moreover the service is rate-limited.

Our ideal system should offer the same advantages as `TextCat` in terms of licensing and run-time speed, while matching the open-domain accuracy and ease-of-use of an API such as `GoogleAPI`. To this end, we release the optimized final LangID system described in this paper, and benchmark it against `TextCat` and `GoogleAPI`. Our major contributions are: (1) we show that negative transfer occurs when training a classifier over a combined set of LangID datasets; (2) we show that language models learned in a particular domain do not always generalize to other domains; (3) we develop a method for extracting features for LangID that are not tied to a particular domain, and show that our method mitigates negative transfer and provides improvements in cross-domain LangID; (4) we show that our method can incorporate additional languages without a penalty in performance on existing languages; and (5) we provide an implementation of our method that is faster than state-of-the-art LangID systems while maintaining competitive accuracy.

2 Background

LangID as a computational task is usually attributed to Gold (1967), who sought to investigate

Proceedings of the 5th International Joint Conference on Natural Language Processing, pages 553–561,
Chiang Mai, Thailand, November 8 – 13, 2011. ©2011 AFNLP

language learnability from a language theory perspective. However, its current form is much more recognizable in the work of Cavnar and Trenkle (1994), where the authors classified documents according to rank order statistics over byte n-grams between a document and a global language profile. Since the 1990s, LangID has been formulated as a supervised machine learning task, and has been greatly influenced by text categorization in general. Monolingual LangID of a test document D_i takes the form of a mapping onto a unique language from a closed set of languages C, i.e. $\mathcal{I} : D_i \rightarrow c_i \in C$.

Statistical approaches applied to LangID include the use of Markov models over n-gram frequencies (Dunning, 1994) and dot products of word frequency vectors (Darnashek, 1995). Kernel methods have also been applied to the task of LangID (Kruengkrai et al., 2005). Linguistically motivated models for LangID have also been proposed, such as stop word list overlap (Johnson, 1993), where a document is classified according to its overlap with lists for different languages. There has also been work on word and part of speech correlation (Grefenstette, 1995), cross-language tokenisation (Giguet, 1995) and grammatical-class models (Dueire Lins and Gonçalves, 2004).

LangID has been applied in a variety of domains, including USENET messages (Cavnar and Trenkle, 1994), web pages (Kikui, 1996; Martins and Silva, 2005; Liu and Liang, 2008), and web search queries (Hammarstrom, 2007; Ceylan and Kim, 2009). It has been shown to improve performance of other tasks such as parsing (Alex et al., 2007) and multilingual text retrieval (McNamee and Mayfield, 2004). It has also been used for gathering data for linguistic corpus creation (Ghani et al., 2004; Baldwin et al., 2006; Xia et al., 2009; Xia and Lewis, 2009), and is an important area of research for supporting low-density languages (Hughes et al., 2006; Abney and Bird, 2010).

Transfer learning refers to the use of data from external domains to improve task performance on a target domain. Pan and Yang (2010) provide a survey, in which they define *transductive* transfer learning, where labels are available in source domain(s) but not in the target domain. This corresponds exactly to our task, where we wish to train a classifier using language-labelled data from a variety of sources, and apply this classifier to target data without making any assumptions about its domain. Pan and Yang (2010) also discuss the phenomenon of *negative transfer*, whereby including data from the source domain(s) results in reduced performance in the target domain.

Dataset	Docs	Langs	Doc Length (bytes)
JRC-ACQUIS	20000	22	18478.5±60836.8
CLUEWEB09	20000	10	36909.0±20735.2
DEBIAN	21735	89	12329.8±30902.7
RCV2	20000	13	3382.7±1671.8
WIKIPEDIA	20000	68	7531.3±16522.2
N-EUROGOV	1500	10	17460.5±39353.4
N-TCL	3174	60	2623.2±3751.9
N-WIKIPEDIA	4963	67	1480.8±4063.9

Table 1: Summary of the LangID datasets

Other work has used the term *domain adaptation* to refer to *inductive* transfer learning, where labels are available in both the source and target domains, and the goal is to improve the performance in the target domain (Daumé III and Marcu, 2006; Daumé III, 2007). Daumé III and Marcu (2006) tackle this problem by using a mixture model, where data in a specific domain is modelled as coming from a mixture of domain-specific and general components, and the linkage between domains is achieved by sharing a single general component. In our task, we have no labels in the target domain, and thus know nothing about the domain-specific component. Thus, our challenge is to build a suitable model of the general component, which we do by eliminating features that make minimal contribution to the general component. This approach makes the conversion of documents to a standardized representation much simpler than a model that decomposes individual features into general and domain-specific components.

3 Data Sources

For this work, we collected language-labelled development data from five sources of diverse origin, and use these as the basis of our examination of in-domain, inductive (all-domain) and transductive (cross-domain) learning. We additionally use three independent test data sets to validate the effectiveness of the final methodology. Statistics of all datasets are provided in Table 1.

3.1 Development data sets

JRC-ACQUIS: JRC-ACQUIS is an aligned multilingual parallel corpus (Steinberger et al., 2006) totalling 463792 documents in 22 languages. From the corpus, we randomly selected 20000 documents without replacement, maintaining the relative skew of languages. For each document, only the text enclosed in the `<body>` tags was retained.

CLUEWEB09: The ClueWeb09 dataset (Callan and Hoy, 2009) consists of about 1 billion web pages in 10 languages.[1] The language of each document was automatically detected using `TextCat`, an implementation of the algorithm of Cavnar and Trenkle (1994). We sampled 20000 instances from the dataset by selecting the first instance in each of 20000 files selected without replacement. Because the language labels are automatically assigned, they do not constitute a true gold-standard, and we thus use CLUEWEB09 exclusively as a training dataset in Section 6.

WIKIPEDIA: Wikimedia provides dumps of the complete contents of all Wikipedia wikis.[2] Individual languages have their own wiki, usually under the corresponding ISO 639-1 code. We obtained XML dumps of the wikis with valid ISO 639-1 codes. From these dumps, we selected 20000 pages in a skew-preserving fashion. In order to ensure that each language contained at least 20 documents, we limited selection to the 68 largest wikis by page count. The data we used was obtained from July–August 2010.

RCV2: Reuters RCV2[3] consists of over 487000 Reuters News stories in 13 languages. We randomly selected 20000 documents in a skew-preserving fashion.

DEBIAN: The Debian Project maintains manual translations of the content strings of a large number of software packages.[4] We obtained all translations with codes corresponding to valid ISO 639-1 codes. This resulted in 21735 language-package pairs in 89 languages.

For each dataset, we randomly divided the documents into two equal-sized partitions. One partition was used for selecting language features and for training, and the other was used for testing. To distinguish between them, we label the partitions A and B, respectively. For example, DEBIAN$_{\text{A}}$ is the partition of the DEBIAN dataset used to compute feature weights and language models, and DEBIAN$_{\text{B}}$ is the partition used for testing. We also make frequent use of the union of the A partitions across all datasets, and will refer to this as UNION$_{\text{A}}$.

3.2 Test data sets

In order to evaluate accuracy in the transductive learning setting, we make use of N-EUROGOV, N-TCL and N-WIKIPEDIA, the three datasets described in detail by Baldwin and Lui (2010a). N-EUROGOV was sourced from the EuroGOV collection used in the 2005 Web-CLEF task, N-TCL was manually sourced by the Thai Computational Linguistics Laboratory (TCL) in 2005 from online news sources, and N-WIKIPEDIA is drawn from a 2008 dump of Wikipedia, with normalization.

4 Learning Algorithms

In this work, we use a multinomial naive Bayes learner. For brevity, we only give a short sketch of the technique; it is described in much more detail by McCallum and Nigam (1998). The crux of the method is to compute the probability that an instance belongs to a class C_i from a given closed set C, and hence assign the most probable class to a document D, consisting of a vector of n features $x_1 ... x_n$:

$$c = argmax_{C_i \in C} P(C_i|D)$$

Bayes' theorem allows us to re-express this as:

$$c = argmax_{C_i \in C} \frac{P(D|C_i)P(C_i)}{P(D)}$$

where $P(D)$ is a normalizing constant independent of C_i. Thus for classification, we only need to estimate $P(D|C_i)$ and $P(C_i)$. C is modelled as a categorical distribution over classes, and so $P(C_i)$ is obtained via a maximum likelihood estimate. In order to estimate $P(D|C_i)$, we make the naive assumption that each term is conditionally independent, hence:

$$P(D|C_j) = \prod_{i=1}^{n} \frac{P(t_i|C_j)^{N_{D,t_i}}}{N_{D,t_i}!}$$

where N_{D,t_i} is the frequency with which term t_i occurs in D.

[1] `http://boston.lti.cs.cmu.edu/clueweb09/`

[2] `http://dumps.wikimedia.org/backup-index.html`

[3] `http://trec.nist.gov/data/reuters/reuters.html`

[4] `http://i18n.debian.net/material/po/unstable/main/`

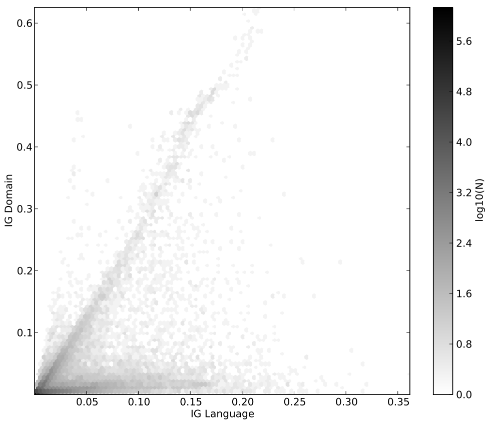

Figure 1: Scatter plot of IG for language vs. IG for domain (byte trigrams)

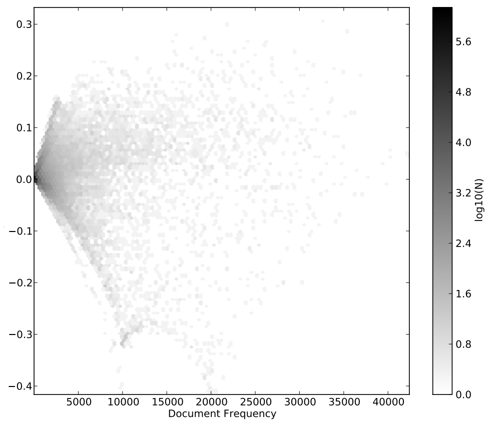

Figure 2: Scatter plot of DF vs. $\mathcal{LD}$ (byte trigrams)

The reason we select multinomial naive Bayes is that it is relatively lightweight and has been shown to be highly accurate when combined with feature selection (McCallum and Nigam, 1998; Manning et al., 2008). To establish the generalizability of our results, we also experimented with a nearest prototype classifier based on skew divergence (Lee, 2001), based on the findings of Baldwin and Lui (2010a). However, when combined with feature selection, we found it was consistently outperformed by the naive Bayes classifier, and thus omit results from this paper. We also experimented with a linear-kernel SVM learner, but once again omit results as we found it to be comparable in accuracy to naive Bayes when combined with feature selection, but much slower to retrain.

5 Feature Selection

The document representation that we use is a mixture of byte n-grams (Cavnar and Trenkle, 1994; Baldwin and Lui, 2010a), as it is language-neutral in that it does not make any assumptions about the language or language type of each document. In particular, we make no assumptions about word delimitation (e.g. via white space) in each language. Results from Baldwin and Lui (2010a) additionally suggest that explicit encoding detection is not necessary in LangID, and that the simple byte tokenization strategy also used in this work is superior to encoding-aware codepoint-based tokenization. Where we have data in more than one encoding, we do not transcode it; instead we simply extract byte features across all the encodings. In practice this is not an issue as the data that we use is mostly UTF8-encoded, with small quantities of other encodings (esp. in N-TCL). We make no further mention of encoding as it is not the focus of this work.

Cavnar and Trenkle (1994) perform feature selection over such a mixture of n-grams, where $1 \leq n \leq 5$. Their feature selection method is based on the frequency of terms, where the N most frequent terms for each language are retained in the global feature set. Related to term frequency is document frequency, where rather than counting individual terms in a class, we count the number of documents that a particular term appears in. Feature selection based on document frequency has been shown to be inferior to other methods. Generally it has been found that Information Gain ($\mathcal{IG}$: Quinlan (1986)) is particularly suited to feature selection in multiclass problem settings such as LangID (Yang and Pedersen, 1997; Forman, 2003).

The novel aspect of our research is that we consider $\mathcal{IG}$ of particular n-gram features along multiple dimensions: (1) with respect to the set of all languages; (2) with respect to a given language; and (3) with respect to the domain the data was obtained from. We are interested in identifying features that have high $\mathcal{IG}$ with respect to language but low $\mathcal{IG}$ with respect to domain.

For each of 1-grams, 2-grams and 3-grams we computed the $\mathcal{IG}$ of each feature with respect to the set of all languages in our development datasets (97 languages), as well as with respect to the set of 5 domains. A scatter plot of $\mathcal{IG}$ for language vs. domain for 3-grams is presented in Figure 1. From this analysis, it is clearly visible that there are two distinct groups of features: one where the $\mathcal{IG}$ for language is strongly correlated with that for domain, and one where the $\mathcal{IG}$ for language is largely independent of that for domain. We are interested in identifying features in the second group, and so for each feature we compute a $\mathcal{LD}$ ($\mathcal{L}$anguage-$\mathcal{D}$omain) score, defined as:

$$\mathcal{LD}^{all}(t) = \mathcal{IG}^{all}_{lang}(t) - \mathcal{IG}_{domain}(t)$$

The number of features to consider grows exponentially with n-gram order. This makes calculating the $\mathcal{IG}$ for higher token n-gram orders computationally infeasible. Yang and Pedersen (1997) found that document frequency ($\mathcal{DF}$) and $\mathcal{IG}$ were strongly correlated. Thus, we studied the relationship between $\mathcal{LD}$ and $\mathcal{DF}$ in our data, and found that low $\mathcal{DF}$ is a good predictor of low $\mathcal{LD}$ score, but not vice versa, as seen in Figure 2. As $\mathcal{DF}$ is much cheaper to compute than $\mathcal{IG}$, we first identify the 15000 features with highest $\mathcal{DF}$ for a given n-gram order, and assign a $\mathcal{LD}$ score of 0 to all features outside this set.

We also consider an alternative formulation of $\mathcal{LD}$. Due to skews between the quantities of data for each language, the highest-ranked features by $\mathcal{LD}$ tend to be biased towards the more common languages. In order to mitigate this, we also consider a formulation whereby we compute a $\mathcal{LD}^{bin}$ score for each feature conditioned on each language l:

$$\mathcal{LD}^{bin}(t|l) = \mathcal{IG}^{bin}_{language}(t|l) - \mathcal{IG}_{domain}(t)$$

In order to give better representaiton to less-frequent languages, we then select the top-scoring features from each language, and define the $\mathcal{LD}^{bin}$ feature set as the union of these features.

The number of features N is a parameter in both methods that we will investigate empirically in Section 6.

6 Experimental Setup and Results

The results presented in Baldwin and Lui (2010a) are based on cross-validation within datasets, without considering the applicabiliy of a model learned in one domain to LangID in another domain. Baldwin and Lui (2010a) also presented preliminary results with regards to feature selection, based on the Cavnar and Trenkle (1994) method. We first compare results with and without feature selection using our $\mathcal{LD}^{bin}$ and $\mathcal{LD}^{all}$ metrics, as well as $\mathcal{IG}^{all}_{lang}$—where we select the top features according to their information gain with the set of all languages—and finally $\mathcal{IG}^{bin}_{lang}$—where we select a fixed number of features per language, according to their information gain with the language.

Baldwin and Lui (2010a) found that byte bigrams performed well as a feature set for LangID, so we use them for this initial experiment. Tokenizing across all of our datasets results in 57160 unique bigrams. For $\mathcal{IG}^{all}_{lang}$ and $\mathcal{LD}^{all}$, we consider the top 2000 to 10000 features, in increments of 1000. For the per-language metrics, we experiment with selecting 100 to 800 features per langauge, in increments of 100.

We perform this experiment in two settings: (1) the supervised learning setting, where we use training and test data from the same domain; and (2) the inductive transfer learning setting, where we train a single classifier on UNION$_{\text{A}}$—the union of the A partitions across the five domains—and use this to classify partition B from each of the five domains in turn. We found that for $\mathcal{IG}^{all}_{lang}$, 10000 features produced the best results; for $\mathcal{IG}^{bin}_{lang}$, it was 100 features per class, corresponding to 4078 features; for $\mathcal{LD}^{all}$, it was 3000 features; and for $\mathcal{LD}^{bin}$ it was 200 features per class, corresponding to 3086 features. We report the results for the best parametrization of each feature selection metric in Tables 2 and 3. In each case, we present the classification accuracy for the full feature set ("Full"), followed by the classification accuracy for feature selection over the same combination of training and test dataset, as the Δ over Full. In all our experiments, the language skew present in each domain is preserved in both the training and the test partitions.

In the supervised case, the accuracy attained is similar when using the full feature set as for the respective reduced feature sets. This shows that utilizing the reduced feature sets does not harm in-domain classification accuracy, implying that we are able to preserve the key features required for classifying the data.

In the case of inductive transfer learning, we observe negative transfer: the greater amount of training data causes the accuracy to drop. This is particularly evident over RCV2, where adding in data from the other four domains results in a drop in accuracy from 0.973 to 0.576. We find that the $\mathcal{LD}$ features are generally able to better mitigate this than the $\mathcal{IG}_{lang}$ features. $\mathcal{LD}^{bin}$ features pro-

Train	Eval	Full	$\mathcal{IG}^{all}_{lang}$	$\mathcal{IG}^{bin}_{lang}$	$\mathcal{LD}^{all}$	$\mathcal{LD}^{bin}$
JRC-Acquis$_{A}$	JRC-Acquis$_{B}$	0.985	+0.000	+0.007	+0.007	+0.005
Debian$_{A}$	Debian$_{B}$	0.963	+0.003	+0.002	−0.001	−0.016
RCV2$_{A}$	RCV2$_{B}$	0.973	−0.017	−0.016	−0.003	+0.026
Wikipedia$_{A}$	Wikipedia$_{B}$	0.935	+0.017	+0.020	+0.030	+0.001

Table 2: In-domain supervised learning accuracy, relative to all features ("Full")

Train	Eval	Full	$\mathcal{IG}^{all}_{lang}$	$\mathcal{IG}^{bin}_{lang}$	$\mathcal{LD}^{all}$	$\mathcal{LD}^{bin}$
Union$_{A}$	JRC-Acquis	0.931	−0.001	+0.039	+0.060	+0.062
	Debian	0.817	−0.031	+0.049	+0.122	+0.127
	RCV2	0.576	−0.048	+0.129	+0.347	+0.410
	Wikipedia	0.739	−0.150	−0.070	+0.179	+0.179

Table 3: Inductive Transfer Learning (all-domain) accuracy, relative to all features ("Full")

vide the best accuracy over each dataset in the inductive transfer learning setting, increasing accuracy over RCV2 to 0.986, exceeding the accuracy of the in-domain classification on the full feature set. We find that $\mathcal{LD}^{bin}$ features can also improve accuracy for in-domain classification, increasing accuracy on RCV2 to a near-perfect 0.999. These results validate the choice of $\mathcal{LD}^{bin}$ as a suitable metric for selecting general features for LangID.

Since our aim is to build a classifier in a transductive transfer learning setting, we examined the behaviour of the $\mathcal{LD}^{bin}$ features in such a setting over our 5 datasets. For each dataset, we trained a classifier on the A partitions, and evaluated the classifier on the B partition of each of the other datasets. Since each dataset covers a different set of languages, there may be languages in the evaluation dataset that are not present in the training dataset. It makes no sense to attempt to classify documents in these languages since we will by definition misclassify them, so the results resported in Table 4 are only over languages in the evaluation set that are also present in the training set. As a result, caution must be exercised in naively comparing accuracy figures across different combinations of training and test datasets.

In Table 4, we see several examples of language models not generalizing to other domains. For example, we see that when using all features, the language models learned from RCV2 classify data from the RCV2 domain with accuracy 0.973, but this drops to 0.838, 0.889 and 0.796 in other domains. We also find that language models from other domains have reduced accuracy when classifying RCV2 data. We find that in all these cases, using the $\mathcal{LD}^{bin}$ features mitigates this loss in accuracy. On RCV2 in particular, use of the $\mathcal{LD}^{bin}$ features results in over 0.95 accuracy when training with any of the 5 domains.

There is a number of domains where the use of the $\mathcal{LD}^{bin}$ features results in a loss in accuracy relative to the full feature set. This contrasts with our results in the inductive learning setting. We hypothesize that this is due to the $\mathcal{LD}^{bin}$ features having been selected from the union of all 5 datasets. While this feature set represents a general model across these 5 domains, for a specific pair of domains there will be some additional features that are strong predictors of language.

Preliminary work by Baldwin and Lui (2010a) suggested that feature selection over a mixture of n-grams yielded promising results. This is also supported by Cavnar and Trenkle (1994), who use a mixture of 1- to 5-grams. We thus investigated the interaction between the range of n-gram orders used for feature selection, the number of features selected per language and classification accuracy. Based on the results in Tables 2, 3 and 4, we used the $\mathcal{LD}^{bin}$ metric exclusively.

We conducted a parameter search for maximum token order M (e.g. $M = 3$ would mean all 1-, 2- and 3-grams) and number of features per language N. We considered $1 \leq M \leq 9$ and $100 \leq N \leq 800$ in increments of 100 features. We tested all combinations exhaustively in a supervised learning task across the union of all 5 datasets. We found that the best results are obtained for $M \geq 4$ and $N \geq 300$. In order to maximize classification rate we need to minimize the number of features used, so we chose $M = 4$ and $N = 300$, producing a feature set consisting of 7480 features.

In building a universal LangID tool, we need to quantify the effect of training on languages extraneous to the target domain. To investigate this, we perform experiments over the datasets of Baldwin and Lui (2010a) using two different classifiers: a reference classifier trained on all languages

	Test Dataset							
Training	JRC-Acquis		Debian		RCV2		Wikipedia	
	Full	$\mathcal{LD}^{bin}$	Full	$\mathcal{LD}^{bin}$	Full	$\mathcal{LD}^{bin}$	Full	$\mathcal{LD}^{bin}$
JRC-Acquis	0.985	+0.005	0.979	−0.055	0.812	+0.174	0.977	−0.026
ClueWeb09	0.981	−0.005	0.989	−0.010	0.925	+0.069	0.927	−0.039
Debian	0.983	−0.003	0.963	−0.016	0.689	+0.261	0.937	−0.058
RCV2	0.838	+0.148	0.889	−0.003	0.973	+0.026	0.796	+0.154
Wikipedia	0.837	+0.141	0.863	+0.011	0.561	+0.407	0.935	+0.001

Table 4: Transductive transfer learning (cross-domain) accuracy relative to all features ("Full")

Test Dataset	`langid.py`		`TextCat`		`TextCat` (retrained)		`GoogleAPI`	
	Accuracy	docs/s	ΔAcc	Slowdown	ΔAcc	Slowdown	ΔAcc	Slowdown
JRC-Acquis	0.991	69.8	−0.164	18.5×	−0.075	11.1×	+0.004	7.0×
Debian	0.969	94.1	−0.305	25.1×	−0.129	14.2×	−0.043	9.4×
RCV2	0.992	146.6	−0.642	34.9×	−0.922	19.1×	+0.005	14.6×
Wikipedia	0.959	102.7	−0.210	26.2×	−0.365	14.9×	−0.012	10.3×
N-EuroGOV	0.987	68.5	−0.046	18.1×	−0.083	11.1×	+0.006	6.9×
N-TCL	0.904	172.1	−0.299	38.8×	−0.232	22.5×	+0.018	17.2×
N-Wikipedia	0.913	209.2	−0.207	45.9×	−0.227	25.7×	−0.010	20.9×

Table 5: Comparison of standalone classification tools, in terms of accuracy and speed (documents/second), relative to `langid.py`

present in the training set, and a domain-specific classifier trained only on languages in the test set. The reference classifier is trained on 1–4-grams, selecting 300 features per language on the full 97 languages, whereas each domain-specific classifier is trained only on the subset of languages present in the target domain.

Figure 3 shows a scatter plot of the per-language accuracy over the different datasets. Rather than harming performance, we find that adding languges extraneous to the target domain generally has no impact on accuracy over the languages in the dataset. In fact, it occasionally positively impacts on accuracy (points to the left of the diagonal), and for the rare instances where it hurts accuracy (points to the right of the diagonal), the difference is relatively modest.

6.1 Comparison to existing tools

Our ultimate interest is in building a standalone classifier that is fast and accurate. For comparison to existing tools, we implemented our method as a Python module (`langid.py`), in the form of a single Python file with pre-trained models for 97 languages. It can act as a standalone LangID system, an embedded Python module, or a web service with an AJAX API.[5]

We compared the speed and accuracy of our system to `TextCat`, as well as `GoogleAPI`. `GoogleAPI` is constrained to: (1) limit the classification rate to 10 documents/second, and (2) base the classification only on the first 500 bytes of the document. These constraints are imposed by the service's terms of use. For `TextCat`, we test it: (1) off-the-shelf using the pre-trained language models, and (2) after retraining it over the same training data as `langid.py`.

Table 5 shows the accuracy of each system across 7 test datasets, as well as the speed in documents per second. We present absolute accuracy and speed for `langid.py`, and relative accuracy and slowdown for the other classifiers. The machine used to perform this experiment was a commodity desktop-class machine, with an Intel Q9400 4-Core CPU, 4GB of RAM and a 7200RPM SATA II hard drive. The slowdown for `GoogleAPI` is reported based on a classification rate of 10 documents per second.

We find that in general, `langid.py` is faster and more accurate than `TextCat` (both off-the-shelf and re-trained). The difference is smallest over a "traditional" LangID dataset like N-EuroGOV which contains only 10 languages, and is much more pronounced over a dataset like N-TCL which has a much larger variety of languages. Comparing `langid.py` and `GoogleAPI`, the two systems are evenly matched in accuracy, but again, `langid.py` is significantly faster. All differences in accuracy are statistically significant (McNemar's test, $p < 0.01$).

We note that the accuracy of `langid.py` is slightly lower on N-TCL than on other datasets. N-TCL has a high proportion of non-UTF8 doc-

[5] The code is available for public download from `http://www.csse.unimelb.edu.au/research/lt/resources/langid/`

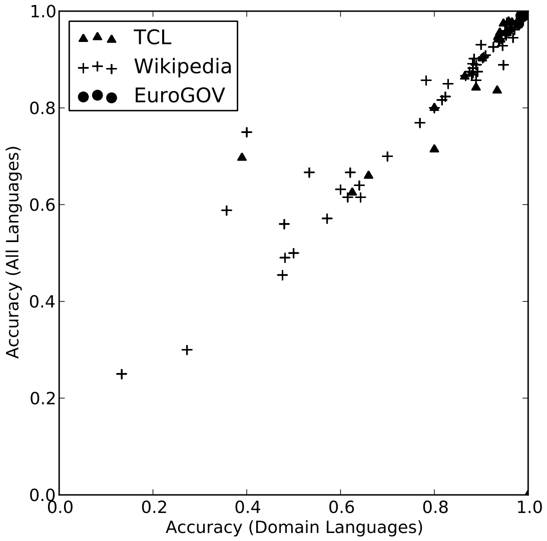

Figure 3: Per-language accuracy of a classifier trained only on languages present in the evaluation domain (x-axis) vs. all languages (y-axis)

Test dataset	Feature selection		
	TextCat	$\mathcal{CT}$	$\mathcal{LD}^{bin}$
N-EUROGOV	0.904	+0.067	+0.083
N-TCL	0.672	+0.016	+0.227
N-WIKIPEDIA	0.686	+0.084	+0.223

Table 6: Comparison of `TextCat` to a `NB` classifier using $\mathcal{CT}$and $\mathcal{LD}^{bin}$ for feature selection, using UNION$_A$ as the training data

uments (57%). To quantify the effect of this, we transcoded all the documents in N-TCL to UTF8 and repeated the experiment. We found that after transcoding, the accuracy of `langid.py` increased from 0.904 to 0.947. This indicates that unsupported encodings have a small impact on accuracy, though it is relatively minor considering that the majority of the data is not UTF8-encoded.

The are two key conceptual differences between `TextCat` and `langid.py`: feature selection and learning algorithm. `TextCat` ($\mathcal{CT}$) selects features per language by term frequency, whereas `langid.py` uses $\mathcal{LD}^{bin}$ (the focus of this work). On the other hand, the learning algorithm used by `TextCat` is a nearest-prototype method using the token rank difference metric of Cavnar and Trenkle (1994), whereas `langid.py` uses multinomial naive Bayes. In order to consider these differences independently, we combine the $\mathcal{CT}$feature selection with the multinomial naive Bayes learner, selecting the union of the top-300 features per language by document frequency over 1- to 5-grams, yielding 10846 features. For comparison, we also selected the top 300 features by $\mathcal{LD}^{bin}$ over 1- to 5-grams, yielding 8166 features. We then trained a naive Bayes classifier over UNION$_A$ using the respective feature sets, and used it to classify each of N-EUROGOV, N-TCL and N-WIKIPEDIA. In Table 6, we compare the accuracy of these two classifiers to (retrained) `TextCat`. Across all datasets, $\mathcal{LD}^{bin}$ is superior to $\mathcal{CT}$. Additionally, `NB` with $\mathcal{CT}$is superior to `TextCat` (which uses the same feature selection strategy, with the nearest prototype learner), although the difference here is much smaller. This suggests that the bulk of the improvement of `langid.py` over `TextCat` is due to the use of $\mathcal{LD}^{bin}$, as developed in this research.

7 Conclusion

We demonstrated the problem of negative transfer in training a LangID system using language-labelled data from a variety of domains. We developed a method for identifying features that are strongly predictive of language across multiple domains by examining the difference in information gain of each feature with language and with the source domain. We used this method to compile a feature set from 50,000 documents in 97 languages across 5 datasets, and implemented this as a standalone LangID system using a naive Bayes classifier. We empirically compared our system to state-of-the-art LangID systems, and found our system to be faster whilst maintaining competitive accuracy.

Acknowledgments

The authors wish to thank Google for providing increased access to the Google AJAX Language Identification API for the purposes of this research. The first author was additionally a recipient of travel support from Google Australia.

NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

References

Steven Abney and Steven Bird. 2010. The human language project: building a Universal Corpus of the world's languages. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 88–97, Uppsala, Sweden.

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.

Timothy Baldwin and Marco Lui. 2010a. Language identification: The long and the short of the matter. In *Proceedings of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT 2010)*, pages 229–237, Los Angeles, USA.

Timothy Baldwin and Marco Lui. 2010b. Multilingual language identification: ALTW 2010 shared task dataset. In *Proceedings of the Australasian Language Technology Workshop 2010 (ALTW 2010)*, pages 5–7, Melbourne, Australia.

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/.

Jamie Callan and Mark Hoy, 2009. *ClueWeb09 Dataset*. Available at http://boston.lti.cs.cmu.edu/Data/clueweb09/.

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.

Hakan Ceylan and Yookyung Kim. 2009. Language identification of search engine queries. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1066–1074, Singapore.

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

Hal Daumé III and Daniel Marcu. 2006. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*, 26(1):101–126.

Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007)*, pages 256–263, Prague, Czech Republic.

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.

George Forman. 2003. An Extensive Empirical Study of Feature Selection Metrics for Text Classification. *Journal of Machine Learning Research*, 3(7-8):1289–1305.

Rayid Ghani, Rosie Jones, and Dunja Mladenic. 2004. Building Minority Language Corpora by Learning to Generate Web Search Queries. *Knowledge and Information Systems*, 7(1):56–83.

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.

Harald Hammarstrom. 2007. A Fine-Grained Model for Language Identication. In *Proceedings of Improving Non English Web Searching (iNEWS07)*, pages 14–20.

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.

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

Genitiro Kikui. 1996. Identifying the coding system and language of on-line documents on the internet. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING '96)*, pages 652–657, Kyoto, Japan.

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.

Jicheng Liu and Chunyan Liang. 2008. Text Categorization of Multilingual Web Pages in Specific Domain. In *Proceedings of the 12th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, PAKDD'08, pages 938–944, Osaka, Japan.

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 (SAC '05)*, page 764.

Andrew McCallum and Kamal Nigam. 1998. A comparison of event models for Naive Bayes text classification. In *Proceedings of the AAAI-98 Workshop on Learning for Text Categorization*, Madison, USA.

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

Sinno Jialin Pan and Qiang Yang. 2010. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, October.

J.R. Quinlan. 1986. Induction of Decision Trees. *Machine Learning*, 1(1):81–106, October.

Ralf Steinberger, Bruno Pouliquen, Anna Widiger, Camelia Ignat, Tomaž Erjavec, Dan Tufis, and Dániel Varga. 2006. The JRC-Acquis: A multilingual aligned parallel corpus with 20+ languages. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC'2006)*, Geona, Italy.

Gertjan van Noord, 1997. *TextCat*. Software available at http://odur.let.rug.nl/~vannoord/TextCat/.

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.

Yiming Yang and Jan O. Pedersen. 1997. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning*.