A two-level classifier for discriminating similar languages

Judit Ács
Budapest University of Technology and Economics
judit@aut.bme.hu

László Grad-Gyenge
Eötvös Loránd University
laszlo.grad-gyenge@creo.hu

Thiago Bruno Rodrigues de Rezende Oliveira
Universidade Federal do Vale do São Francisco
thiago_brro@hotmail.com

Abstract

The BRUniBP team's submission is presented for the Discriminating between Similar Languages Shared Task 2015. Our method is a two phase classifier that utilizes both character and word-level features. The evaluation shows 100% accuracy on language group identification and 93.66% accuracy on language identification. The main contribution of the paper is a memory-efficient correlation based feature selection method.

1 Introduction

The discrimination of similar languages (DSL) (Zampieri et al., 2015) can be defined as the subtask of the language identification (LI) problem. LI is a fundamental task in the area of natural language processing (NLP). The primary goal of LI is to determine the language of a written text. In practical applications, LI acts as a preprocessor of various NLP techniques as for example machine translation, sentiment analysis or even web search. LI is currently an actively researched topic, DSL is also in the focus of interest (Tiedemann and Ljubešić, 2012).

Unlike well-separated languages, multilingualism, varieties or dialects of language can seriously degrade the quality of LI. DSL, noisy data, non-well-formatted text, short sentences, mixed language (i.e. tweets) are other examples of challenging problems in this field. In this paper we focus on the DSL problem on a shared task. Our experiment shows that discrimination between Bosnian, Croatian and Serbian and between Argentinian and Peninsular Spanish are the most challenging tasks for our methods.

Most state of the art methods solve the DSL task in two phases. In the first phase the language group is to be identified, in the second phase the language is to be selected. The first decision of the model is more coarse and high level, the second labelling is to be more specialized as different language groups have different separating features. Regarding the information representation, most methods work with statistical features of the source text. The statistical features are n-grams at the word and at the character level. The parameter n in the fixed-length character and word n-gram models ranges from 1 to 6.

In our approach a maximum entropy classifier and SVM with different kernels were evaluated. The results show that maximum entropy delivers comparable results to SVM while it is considerably faster. To tackle the issue of zero probabilities resulting from unseen n-grams, Katz's back-off smoothing (Katz, 1987) is applied. Training a classifier on a large number of features requires substantial computing resources, which we do not have readily accessible. Features are pruned to less than 10,000 according to their pairwise Pearson correlation with the labels. The code is available on GitHub.[1]

Section 2 presents related work. Section 3 describes the dataset the methods are evaluated on. Section 4 provides an overview of the architecture of our method and describes the classification method. Section 5 gives insight into how the text is preprocessed before calculating the statistical features. Section 6 presents the evaluation results. Section 7 concludes the paper.

2 Related work

Most of the DSL methods have a two phase architecture. The first level is to determine the language group, the second level is to discriminate within the language group.

(Porta and Sancho, 2014) utilize maximum entropy models for the DSL task. The first classifier

[1] http://github.com/juditacs/dsl

determines the language group, the second works with empirically selected features that achieved best performance for the specific language group. (Lui et al., 2014) also define a two phase approach involving a POS-tagger. (Goutte et al., 2014) label the language group with a probabilistic model based on word co-occurrences in documents. To discriminate at the language group level, SVM based classification is used. (King et al., 2014) compare naïve Bayes, logistic regression and SVM based classifiers. They also preprocess the data with manually defined methods as named entity removal and English word removal. (Purver, 2014) introduces a single-level approach, training a linear SVM on word and character n-grams of length 1-3.

3 Dataset

Our method is evaluated on the DSLCC dataset (Tan et al., 2014), which dataset is provided for the shared task. As Tan et al. describe the collection and the preparation of the dataset in detail, we only provide a summary in Table 1. The dataset contains 6 language groups of closely related languages and dialects plus one group called *other*. The language groups are presented in the first column (`Group`) of the table. The second column (`Language`) identifies the language, the third column (`code`) contains a short identifier for each language.

For each language, the dataset consists of 20 000 sentences. Each list of sentences is divided into two parts as 18 000 sentence training sample and 2 000 sentence development sample.

Group	Language	Code
A	Bulgarian	bg
	Macedonian	mk
B	Bosnian	bs
	Croatian	hr
	Serbian	sr
C	Czech	cs
	Slovak	sk
D	Argentinian Spanish	es-AR
	Peninsular Spanish	es-ES
E	Brazilian Portuguese	pt-BR
	European Portuguese	pt-PT
F	Malay	my
	Indonesian	id
X	other	xx

Table 1: Language groups and languages

4 Method

To solve the shared task, we introduce a two-level architecture. On the first level we utilize a classifier to distinguish between the language groups. We refer to this classifier later as *inter-group classifier*. The inter-group classifier is described in Section 4.1. To conduct a more specialized decision, to distinguish between the languages in a language group, a second-level classifier is utilized. This classifier is titled the *intra-group* classifier and is described in Section 4.2

4.1 Inter-group classifier

Although the dataset contains 7 language groups, we trained the classifier on 14 labels according to the languages (instead of groups) and grouped the corresponding labels together according to the language groups.

From the variety of features tested (see Section 4.2), tf-idf delivered the best results for the inter-group classification. Although tf-idf is a more common method for information retrieval tasks, it can also be defined for the current task as follows

document set of all sentences in one language,

term one word, see Section 5 for details,

query one test sentence.

The inter-group classifier operates in two steps. In the first step the top 100,000 keywords are extracted for each language. In the second step the weighted sum of keywords is computed for each sentence in each language and the language with the highest score is chosen.

The inter-group classifier provides 100% accuracy on language group identification. Regarding language labelling, the accuracy of the inter-group classifier is 92.54%.

We used the following tf, idf and qf weightings:

$$tf_{t,d} = \log(1 + f_{t,d}),$$

where $f_{t,d}$ is the raw frequency of a term in all sentences in a language.

$$idf_t = \log(1 + \frac{N}{n_t}),$$

where N is the number of languages and n_t is the number of languages in which term t appears and

$$qf_t = \left(0.5 + 0.5\frac{f_{t,q}}{\max_t f_{t,q}}\right) \times \log\frac{N}{n_t},$$

where qf_t is the weight of term t, $\max_t f_{t,q}$ is the highest *tf* score for term t in any language.

4.2 Intra-group classifier

The language groups are refined by the intra-group classifier further. In the case of groups A, C, D, E and F, there are 2 languages two distinguish between. In the case of group B, there are 3 languages to label. In the case of "group" X there is only 1 language, the intra-group classifier is not used in this case.

Various features are extracted and 6 models are trained for the 6 groups. Character and word n-grams, Katz's backoff scores, tf-idf scores and stopword n-grams are used as features.

4.2.1 N-grams

Character n-grams proved to be the most prominent feature in last year's DSL task (Zampieri et al., 2014). However, the number of character n-grams grows exponentially with n in theory and subexponentially in practice but it still results in a large number of features. This is the reason why we involved feature selection.

Since PCA and other popular dimension reduction methods are very memory-intensive, Pearson correlation is involved as a feature selection method. To select the most relevant features, for each feature, the absolute value of the Pearson correlation with the labels is calculated and based on this value, the top n features are selected.

4.2.2 Katz's backoff smoothing

Our baseline system is an implementation of Katz's backoff smoothing with training option that works well in the general setting.[2] It is possible to train and test with this system up to $n = 4$ grams with reasonable memory consumption. For further memory-saving, see Section 5.

There are several variants of Katz's backoff smoothing, the one used here discounts the Maximum Likelihood estimations with a constant factor and distributes the leftover probability mass according to lower order n-grams.

$$P_{bo}(c_n|c_1,\ldots,c_n) =$$
$$= \begin{cases} \frac{C(c_1,\ldots,c_n)-d}{C(c_1,\ldots,c_{n-1})}, & \text{if } C(c_1,\ldots,c_n) > 0 \\ \alpha_{c_1,\ldots,c_{n-1}} P_{bo}(c_n|c_2,\ldots,c_{n-1}) & \text{otherwise,} \end{cases}$$

where $\alpha_{c_1,\ldots,c_{n-1}}$ is the left-over probability mass from discounting:

$$\alpha_{c_1,\ldots,c_{n-1}} = 1 - \sum_{c_n : C(c_1,\ldots,c_n)>0} \frac{C(c_1,\ldots,c_n) - d}{C(c_1,\ldots,c_{n-1})}.$$

The probabilities for all the languages are calculated on n-grams of various size, n is ranging from 1 to 4. The language with the highest probability is selected. Both the probabilities and the language are used as features and are passed to the intra-group classifier.

4.2.3 Tf-idf

Similarly to the case of the inter-group classifier, tf-idf scores are calculated for each language group. The language group specific tf-idf scores are based on the sentences only in the specific language group. The tf-idf scores are the used as features later for the intra-group classification.

4.2.4 Word bigrams

Word bigrams are extracted and after selection are used as features. The selection of word bigrams is similar to the selection of the character n-grams. The absolute value of Pearson correlation of the bigrams with the labels is calculated and then the top n bigrams are selected. In our experiment n is set to 1 000.

4.2.5 Stopwords

Although language varieties may use virtually the same vocabulary, we assume that common expressions, word and clause order may differ and the order of stopwords reflects this difference. In each language, we filtered the corpus to its 200 most frequent words, most of which are stopwords. The filtered sentences were fed as input to the tf-idf (see Section 4.1) and the word bigram extractor (see Section 4.2.4), resulting in a much smaller feature number, therefore no feature selection was necessary.

[2] https://github.com/juditacs/langid

	Accuracy
run1	0.9331428571
run2	0.9366428571
run3	0.9348571429

Table 2: Overall accuracy of our three runs

4.2.6 Classifier

Scikit-learn's (Pedregosa et al., 2011) maximum entropy and SVM based classifiers are evaluated. In the case of SVM, different kernels are utilized. Due to space limitations and focus we do not publish these evaluation results. As the maximum entropy classifier delivers comparable results to the SVM based method, we use the maximum entropy classifier as it is considerably faster.

5 Preprocessing and tokenization

In order to reduce the number of components that do not contribute much to language identification, the corpus is preprocessed before feature extraction. The preprocessing pipeline consists of the following steps

lowercasing Python's `unicode.lower` function is used

puncutation filtering standard punctuation (Python's `string.punctuation`) and additional quotation symbols are removed

whitespace normalization multiple consecutive whitespaces in the same sentence are replaced with a single space

digit replacement numbers are replaced with a single 0

All steps are applied before feature extraction except in the case of tf-idf, where lowercasing is not performed.

The preprocessed text is tokenized with NLTK (Bird, 2006). Although the tokenizer is trained on English punctuation corpus, it performs reasonably well for the current languages.

6 Results

Three runs are submitted. The first two runs are the same except that in the second run we took advantage of the fact that the labels were balanced. The third run only differed in thresholding for group B. The overall accuracy of each run is listed in Table 2 and the detailed results of our best run can be found in Table 3.

Group	Languages	Accuracy
A	bg,mk	1.0
B	bs,hr,sr	0.8417
C	cs,sk	1.0
D	es-AR,es-ES	0.867
E	pt-BR,pt-PT	0.929
F	id,my	0.998
other	xx	1.0
sum	0.9366	

Table 3: Detailed results of our best run

To have a deeper insight into the limitations of solving the shared task, a manual evaluation has been performed on the 152 sentences misclassified Portuguese sentences by our best run by two Brazilian native speakers. The annotators have been asked to label each sentence as BR (Brazilian), PT (European Portuguese) or UN (Unknown). UN tag has been introduced to avoid guessing. Their very low agreement on the labels (Cohen's kappa 0.28) and the fact that only 22 sentences have been labeled correctly (according to the gold standard) by both of them suggests that there is very little room for improvement on the shared task.

7 Conclusion

We presented the system description of our DSL2015 task submission which performed an overall accuracy of 93.66%. We ended up the 5th place out of 8 submissions.

We introduced a two-level classifier with a variety of features: character and word n-grams, tf-idf, stopword bigrams and tf-idf, and smoothed language models. The first level solely relies on the output of tf-idf, capable of grouping languages with 100% accuracy. The second level combines all features and uses the maximum entropy classifier to classify languages within language groups.

Memory efficiency is a key issue in our research. Our most important steps consume less than 5GB RAM.

References

Steven Bird. 2006. NLTK: the natural language toolkit. In *Proceedings of the COLING/ACL on Interactive presentation sessions*, pages 69–72. Association for Computational Linguistics.

Cyril Goutte, Serge Léger, and Marine Carpuat. 2014. The NRC system for discriminating similar languages. In *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects*, pages 139–145, Dublin, Ireland, August. Association for Computational Linguistics.

S. Katz. 1987. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech and Signal processing*, 35(3):400–401.

Ben King, Dragomir Radev, and Steven Abney. 2014. Experiments in sentence language identification with groups of similar languages. In *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects*, pages 146–154, Dublin, Ireland, August. Association for Computational Linguistics.

Marco Lui, Ned Letcher, Oliver Adams, Long Duong, Paul Cook, and Timothy Baldwin. 2014. Exploring methods and resources for discriminating similar languages. In *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects*, pages 129–138, Dublin, Ireland, August. Association for Computational Linguistics.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Jordi Porta and José-Luis Sancho. 2014. Using maximum entropy models to discriminate between similar languages and varieties. In *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects*, pages 120–128, Dublin, Ireland, August. Association for Computational Linguistics.

Matthew Purver. 2014. A simple baseline for discriminating similar languages. In *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects*, pages 155–160, Dublin, Ireland, August. Association for Computational Linguistics.

Liling Tan, Marcos Zampieri, Nikola Ljubešic, and Jörg Tiedemann. 2014. Merging comparable data sources for the discrimination of similar languages: The dsl corpus collection. In *Proceedings of the 7th Workshop on Building and Using Comparable Corpora*, pages 11–15, Reykjavik, Iceland.

Jörg Tiedemann and Nikola Ljubešić. 2012. Efficient discrimination between closely related languages. In *Proceedings of COLING 2012*, pages 2619–2634, Mumbai, India.

Marcos Zampieri, Liling Tan, Nikola Ljubešić, and Jörg Tiedemann. 2014. A report on the DSL Shared Task 2014. In *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects*, pages 58–67, Dublin, Ireland, August. Association for Computational Linguistics.

Marcos Zampieri, Liling Tan, Nikola Ljubešić, Jörg Tiedemann, and Preslav Nakov. 2015. Overview of the DSL Shared Task 2015. In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects (LT4VarDial)*, Hissar, Bulgaria.