Efficient Discrimination Between Closely Related Languages

Jörg TIEDEMANN[1] *Nikola LJUBEŠIĆ*[2]
(1) Department of Linguistics and Philology, Uppsala University,
Box 635, SE-751 26 Uppsala, Sweden
(2) Faculty of Humanities and Social Sciences, University of Zagreb,
Ivana Lučića 3, 10000 Zagreb, Croatia
jorg.tiedemann@lingfil.uu.se, nikola.ljubesic@ffzg.hr

ABSTRACT
In this paper, we revisit the problem of language identification with the focus on proper discrimination between closely related languages. Strong similarities between certain languages make it very hard to classify them correctly using standard methods that have been proposed in the literature. Dedicated models that focus on specific discrimination tasks help to improve the accuracy of general-purpose language identification tools. We propose and compare methods based on simple document classification techniques trained on parallel corpora of closely related languages and methods that emphasize discriminating features in terms of blacklisted words. Our experiments demonstrate that these techniques are highly accurate for the difficult task of discriminating between Bosnian, Croatian and Serbian. The best setup yields an absolute improvement of over 9% in accuracy over the best performing baseline using a state-of-the-art language identification tool.

KEYWORDS: language identification, language discrimination, closely related languages.

Proceedings of COLING 2012: Technical Papers, pages 2619–2634,
COLING 2012, Mumbai, December 2012.

1 Introduction

Language identification becomes increasingly important in applications and research that rely on data collected from the web and user contributed content. The increased interest in automatic classification of texts can be seen in the number of tools and publications related to this task. Rather simple statistical techniques based on character N-grams and other orthographic features have been shown to be very effective even for the distinction of quite a large number of different languages. However, closely related languages are much harder to distinguish (even for humans) and standard approaches usually fail badly. Furthermore, low-density languages that are strongly related to other languages with larger resources are often not supported by language identifiers with pre-trained models. Some popular tools, such as the language detector integrated in the Google Chromium project, cannot be trained at all and, therefore, cannot be extended easily.

For these reasons, we consider in this paper the specific task of discriminating between closely related languages as a sub-task in automatic language identification. We will show that dedicated, yet simple models can greatly contribute to a better classification of those languages and, in this way, can help to build up resources for low-density languages. In our experiments, we focus on three related south-slavic languages: Bosnian, Croatian and Serbian. They represent a prototypical example for the problem we like to concentrate our work on. Their genetical closeness and lexical similarities make it very hard to distinguish between texts written in either language, which we will see further down in our initial experiments with standard language identification tools.

2 Related work

The problem of language identification was for quite some time considered a rather easy one and mostly solved. One of the most frequently used system in the academic world was TextCat based on the algorithm described in (Cavnar and Trenkle, 1994). This system uses character n-grams as features of length 1<=n<=4 that are most frequent in the specific language. Beside character N-grams, some systems also use the most frequent words as features (Batchelder, 1992).

With the dramatic increase of multilingual data on the web, the language identification problem has received new attention. Additionally, with the rise of new domains, such as microblogging, domain adaptation for language identification has become an important problem. Lui and Baldwin (2011), for example, tackle the problem by calculating information gain over multiple domains choosing those character n-gram features that are the most domain independent ones.

A harder problem emerging with the availability of data in many languages is the problem of discriminating between closely related languages. However, only a few researchers dealt with that problem in the past. Padró and Padró (2004) report problems with the distinction of Catalan and Spanish when using standard language identification methods showing that a character-based Markov chain is the least erroneous one having an error rate of roughly 6% on very short texts and 1% on longer ones, This accounts for 40% of errors on a task of discriminating between six languages. Furthermore, the inability to discriminate Portugese and Brazilian Portuguese with TextCat has been reported by Martins and Silva (2005). Finally, in our previous work (Ljubešić et al., 2007) we discuss the identification of Croatian in a pool of Slovenian, Croatian and Serbian documents showing very good results with a second-order Markov chain with 100% accurate discrimination of Slovene and a 96% accurate discrimination between Croatian and Serbian. We also show that the latter could be improved by using simple lists of forbidden words to achieve up to 99% accuracy. These findings, in particular, motivated us to further explore the identification of these three languages with the techniques we present below.

Let us first have a quick look at the differences between the languages we focus on before introducing our approaches for language discrimination.

2.1 Differences between Bosnian, Croatian and Serbian

The three languages are considered very similar and until recently there was an open discussion if these languages should be considered linguistically separate.

If we consider the usage of Latin script only,[1] the largest difference between Croatian and Serbian is the present form of the proto-Slavic vowel *jat*, resulting in Serbian following more the ekavian and Croatian the ijekavian reflex which introduces many lexical differences (*child* – *dete* (sr) vs. *dijete* (hr, bs)). In this feature, Bosnian generally follows Croatian.

Internationalisms and proper nouns are handled differently as well with transliterations being frequent in Serbian, less frequent in Bosnian while in Croatian foreign proper names from Latin-script languages are written in the original orthography (*Nju Jork* (sr,bs) vs. *New York* (hr)).

In morphology the most important difference is the existence of synthetic future tense in Serbian (*videću*) beside the analytical one (*vidjet ću* or *ću vidjeti*) while Croatian and Bosnian only use the analytical form. There are also systematic differences in the derivation of nouns, adjectives and verbs (*organizovati*, *organizovan* (sr,bs) vs. *organizirati*, *organiziran* (hr,bs)) with Bosnian using both again.

There are some syntactic differences as well. The most visible one is the structure *modal verb + da + present* in Serbian and *modal verb + infinitive* in Croatian (*hoću da radim* (sr, bs) vs. *hoću raditi* (hr, bs)), Bosnian allowing both. Finally, Croatian and Serbian show a series of differences in the general vocabulary (*fabrika* (sr, bs) vs. *tvornica* (hr,bs)), Bosnian again allowing both, but additionally introducing a lot of lexical units culturally bound with the Moslem world. Apart from the fact that these three languages have common origins, today they are three distinct and codified standards, and texts in these standard languages appear regularly.[2]

Being aware that this is an oversimplification, we conclude that Croatian and Serbian are visibly different languages while Bosnian is a mixture of the two with a tendency towards Croatian. To support this claim, we present in table 1 the overlap of lowercased tokens calculated on the parallel corpus used later on for training our classifiers. We can observe that Croatian and Serbian are the most different languages, Bosnian and Serbian coming second and Bosnian and Croatian being the most similar ones. By not calculating type, but token intersections, we are also able to observe the amount of symmetry in the token overlap inside language pairs. The largest difference in the token overlap is shown between Croatian and Serbian showing 5%

[1] In contemporary Serbian around 50% of texts appear written in Cyrillc, Bosnian much less, while Croatian doesn't use the Cyrillc script at all.

[2] In our research we did not take into account the appearance of a new Montenegrin standard, with its reformed orthography that makes it distinct from any of these three languages. This development appeared only recently and this language should be considered as an extremely low-density language (total number of speakers is less than half a million).

more tokens of Serbian appearing in Croatian texts than vice versa. The most likely reason for this phenomenon is the tendency of Croatian language to break from the tradition of language unification efforts that existed in Yugoslavian times.

	bs	hr	sr
bs		0.952	0.915
hr	0.950		0.857
sr	0.930	0.902	

Table 1: Overlap of lowercased tokens between languages in the parallel corpus of Bosnian (bs), Croatian (hr) and Serbian (sr). Rows represent tokens, columns corpora of specific languages.

In order to stress the problems in automatic discrimination between these languages, we trained models using three popular tools (TextCat[3], Lingua::Identify[4] and langid.py[5]) with data taken from SETimes[6], a collection of parallel texts from an on-line news portal for the Southeast European region that publishes “news and views from Southeast Europe” in ten languages.[7] The trilingual parallel corpus we extracted from the dataset contains 5,536 parallel documents and roughly 2.7 million words per language. We trained models for only the three languages of interest in order to avoid any further confusions of the classifiers. For evaluation purposes, we extracted and manually verified 600 documents (200 per language) from three on-line resources, one in each language.[8] Table 2 shows the confusion matrices produced by these tools.

TextCat
Overall accruacy: 55.5%

	bs	hr	sr
bs	90	36	74
hr	68	65	67
sr	14	8	178

Accuracy for Bosnian: 45%

Lingua::Identify
Overall accuracy: 48.8%

	bs	hr	sr
bs	65	117	18
hr	43	151	6
sr	41	82	77

Accuracy for Bosnian: 32.5%

langid.py
Overall accuracy: 87.7%

	bs	hr	sr
bs	139	56	5
hr	11	187	2
sr	0	0	200

Accuracy for Bosnian: 69.5%

Table 2: Special-purpose classifiers trained with standard tools for language identification: bs (Bosnian), hr (Croatian) and Serbian (sr).

As we can see, there is significant difference between the results of the three classifiers. langid.py performs much better than the other two, which is most probably due to the more sophisticated learning algorithm used in that approach (Lui and Baldwin, 2012). Namely, it uses information gain for the selection of features that discriminate best between languages. An additional remark should be made that this system focuses on domain robustness and not the problem of discriminating similar languages. On the other hand TextCat only uses the most frequent N-grams per language and Lingua::Identify uses prefixes, suffixes and frequent words and, apparently, none of these features can well discriminate similar languages such as the ones we

[3] http://www.let.rug.nl/~vannoord/TextCat/

[4] http://search.cpan.org/~ambs/Lingua-Identify-0.51/

[5] https://github.com/saffsd/langid.py

[6] http://www.setimes.com

[7] The data set is freely available from http://www.nljubesic.net/resources/corpora/setimes/.

[8] http://www.dnevniavaz.ba, http://www.vecernji.hr and http://www.politika.rs

deal with. Although langid.py performs much better than the other two systems, the overall accuracy is still much below general language identification performances reported in the literature. This is especially true for the discrimination between Bosnian and Croatian. The accuracy of 0.695 for the recognition of Bosnian texts is just not acceptable.[9]

3 Discriminating Between Closely Related Languages

In this section, we will discuss two approaches to language discrimination: One is based on a document classification method and the other one focuses on the identification of indicator features in terms of blacklisted words. Both approaches are in their essence quite similar, but they show a different performance on various amounts of training data and on data which is not entirely parallel, but comparable.

3.1 Learning a Document Classifier

The idea of using document classification techniques for language identification relies on the prerequisite of possessing parallel data of closely related languages. Since parallel texts communicate identical content, the differences in the bitext of closely related languages are exactly the differences between these two languages. By learning to discriminate between these datasets we actually learn the difference between the languages. Using document classification techniques on non-parallel data would model content alongside language specificities and is expected to perform worse than the models built on parallel data.

We chose to use the multinomial Naive Bayes classifier (McCallum and Nigam, 1998) because of its general good performance and speed. Additionally, because its parameters are actually probabilities of words given the category, i.e. language, this makes the model easily readable for humans. We estimate the model parameters as probabilities of words given the class

$$\hat{P}(w_i|c_j) = \frac{c(w_i, c_j)}{c(w_i)}$$

where $c(w_i, c_j)$ is the count for word w_i in texts of class c_j and $c(w_i)$ is the count of word w in the whole corpus.

We predict the language by maximum a posteriori class c_{map}

$$c_{map} = \arg\max_{c} \sum_{w} \log \hat{P}(w_i|c_j)$$

by summing over logarithms of probabilities of words given the class for each word in the document to be classified. We use simple add-one smoothing to take care of unseen events. Since we consider all languages equiprobable, we omit the prior probability in the procedure.

[9] We decided to refer to the performance on individual languages in terms of accuracy when applying only that subset of the data to the classifier. Alternatively, one could look at precision and recall values for individual languages computed over the entire data set. In that case, our language-specific accuracy values correspond to recall. Using the example of langid.py, precision of Bosnian would then be 0.927 and for Croatian as low as 0.77.

3.2 Learning “Blacklisted Words”

The difference between related languages can often be explained by some distinctive words that occur quite frequently in one language but never in the other language (at least not with exactly the same spelling). The use of “forbidden words” in language identification has already been shown in Ljubešić et al. (2007) as discussed in the introduction. This observation leads to the idea of building a classifier entirely based on those distinctive features that clearly discriminate between two languages. One could say that these words are on a “blacklist” for the language in which they should not appear. Observing one of those words should give very strong evidence against the language they are blacklisted for. We consider this idea to be strictly binary between two well-defined classes. Blacklists should only provide evidence to distinguish one language from exactly one other one.

Blacklists could be built manually using linguistic intuitions by native speakers. However, it is also possible to derive such data sets from corpora simply by comparing word frequencies. As stated above, we are looking for words that are (rather) common in one language but forbidden in the other. The most simplistic approach would use raw frequencies to find relatively frequent words that do not appear in texts of the other language. A simple frequency threshold could be sufficient for this purpose.

$$w \in B_2 \;\; if \;\; c_1(w) > \theta \wedge c_2(w) = 0$$

where B_2 is the set of blacklisted words for language L_2, $c_1(w)$ and $c_2(w)$ are the frequency counts of word w in language L_1 and L_2, respectively, and θ is the threshold to filter out infrequent words. These blacklists could be used as strict filters (as in spam filtering) but this would lead to (possibly a lot) of documents that are blacklisted for both languages as some of the words may still be valid for a language in which they do not appear in the training data. Certainly, it is again important to focus on texts from similar domains in order to avoid spurious signals indicating domain differences instead of language differences.

One possibility to make the classifier more robust is to use counts of matching blacklist words to, for example, classify document D:

$$lang(D) = \begin{cases} L_2 & if \; \sum_{w \in D \wedge w \in B_1} 1 \; > \sum_{w \in D \wedge w \in B_2} 1 \\ L_1 & \text{otherwise} \end{cases}$$

However, in this approach each blacklisted word has the same impact on the final classification and the frequency threshold θ becomes very important when collecting the data sets. Furthermore, it may not be wise to restrict the blacklists to words that do not appear at all in the other language. Certain elements (quoted expressions, names, titles) may spoil the data and useful discriminators will be missed by this strict approach.

Hence, another idea is to use the difference of relative frequencies (as an MLE-based approximation of unigram probabilities) between words appearing in one or both languages. In order to measure the difference, we define the following ratio (N_1 and N_2 are the total word counts in language L_1 and L_2, respectively):

$$\delta(w, L_1, L_2) = \frac{c_1(w)/N_1 - c_2(w)/N_2}{c_1(w)/N_1 + c_2(w)/N_2} = \frac{c_1(w)N_2 - c_2(w)N_1}{c_1(w)N_2 + c_2(w)N_1}$$

Furthermore, we restrict the candidates to words that appear less than a certain frequency threshold α in one language and more than another frequency threshold β in the other language. Yet another threshold (γ) can be set to restrict the set to the most discriminating words by adding the constraint $|\delta(w, L_1, L_2)| > \gamma$. Negative scores indicate a feature supporting L_2 and positive scores support L_1.

Using this set of weighted features, we can now define an adjusted decision rule in the following way:

$$lang(D) = \begin{cases} L_2 & if \ \sum_{w \in D} \delta(w, L_1, L_2) < 0 \\ L_1 & \text{otherwise} \end{cases}$$

Using this definition, the Blacklist classifier becomes quite similar to a two-class Naive Bayes approach but with heavy feature selection using only the most promising discriminating tokens for classification. We could even introduce yet another threshold to define a margin that describes the grey area of uncertain decisions. In that case, we would end up with a classifier that uses the following decision rule:

$$lang(D) = \begin{cases} L_1 & if \ \sum_{w \in D} \delta(w, L_1, L_2) > \mu \\ L_2 & if \ \sum_{w \in D} \delta(w, L_1, L_2) < -\mu \end{cases}$$

However, we do not apply this method in the present paper as this introduces yet another free parameter that needs to be adjusted.

Another practical consideration is to avoid spurious tokens such as proper names or tokens containing non-alphabetic characters. Some texts may frequently mention certain names, numbers, dates or other named entities that do not appear in texts of the other language. In that case, they would easily end up in blacklists without being appropriate for language discrimination. On the other hand, certain punctuation differences may also work quite well for distinguishing between languages. However, in our experiments we simply dismiss all tokens containing non-alphabetic characters.

For the discrimination between more than two languages, we can use a simple cascaded setup of pairwise classifications. The disadvantage of this procedure is that the order of classification steps may influence the final result. However, we could not see any significant impact on our results in the experiments described below. We, therefore, did not try to optimize this procedure for better discrimination.

4 Experiments

Our experiments are based on the same data sets as we have used for our baseline approaches presented in section 1. In particular, we used the portion of the parallel SETimes corpus containing Bosnian, Croatian and Serbian with its 2.7 million words per language. The evaluation data contains 200 documents per language with about 78 thousand words of Bosnian, 70 thousand words of Croatian and 113 thousand words of Serbian. Hence, the average document length is rather short – between 350 and 500 tokens per document.

In order to compare our approaches to a strong baseline, we built a second-order Markov chain as described in (Ljubešić et al., 2007) that has been proven to be very efficient for the

	bs	hr	sr	accuracy
bs	173	17	10	0.865
hr	30	170	0	0.850
sr	1	0	199	0.995

Table 3: Applying a second-order Markov chain to our test data.

discrimination between related languages such as Slovene, Croatian and Serbian. The confusion matrix of classifying our test set using this method is shown in table 3.

The overall accuracy of this approach is 90.3%. As expected, it still has major problems with distinguishing Bosnian and Croatian.

Table 4 shows the results of our proposed classifiers when applied to the test data.

Naive Bayes: overall accuracy: 95.7%

	bs	hr	sr	accuracy
bs	181	11	8	0.905
hr	7	193	0	0.965
sr	0	0	200	1.000

Blacklist Classifier: overall accuracy: 97%

	bs	hr	sr	accuracy
bs	187	12	1	0.935
hr	5	195	0	0.975
sr	0	0	200	1.000

Table 4: The confusion matrices of applying our classifiers to the test data.

For the blacklist approach, we always apply Serbian-Croatian discrimination before discriminating between the preferred language from the first step and Bosnian. The blacklisted word extraction parameters are set to intuitively chosen values: $c_{low}(w) < \alpha = 4$, $c_{high}(w) > \beta = 9$ and $|\delta| > \gamma = 0.8$. We did not perform any optimization of these settings so far, which we, however, plan to do in future work.

We can see that the overall accuracy is much higher than for the general-purpose tools even when trained for this specific task. The difference is especially striking for the case of Bosnian which could be identified with over 90% accuracy in both cases. To test the statistical significance of the differences in performance of the classifiers we use the approximate randomization procedure (Hoeffding, 1952; Yeh, 2000) with 1000 repetitions. The obtained p-values are presented in Table 5.

classifier 1	classifier 2	difference in accuracy	p-value
Naive Bayes	TextCat	0.401	**0.001**
Naive Bayes	Lingua::Identify	0.468	**0.001**
Naive Bayes	langid.py	0.080	**0.001**
Naive Bayes	Markov chain	0.053	**0.001**
Blacklist	TextCat	0.415	**0.001**
Blacklist	Lingua::Identify	0.482	**0.001**
Blacklist	langid.py	0.093	**0.001**
Blacklist	Markov chain	0.067	**0.001**
Naive Bayes	Blacklist	0.013	0.188

Table 5: Comparison of the presented approaches in terms of overall accuracy.

The difference between the Naive Bayes and Blacklist classifiers has shown not to be statistically significant on this size of the evaluation set ($p = 0.188$), but the difference is an interesting fact that should be looked into in future work. However, the difference between the Naive Bayes and Blacklist classifiers to the other classifiers is highly statistically significant ($p < 0.001$) while the difference between our baseline Markov chain and the langid.py classifier (0.027) is marginal ($p = 0.094$).

4.1 Size of Training Data

Despite their higher performance, a possible disadvantage of our token-based classifiers (compared to classifiers based on character sequences) is that they may have larger problems when trained on limited amount of data, non-parallel texts and non-comparable domains. Figure 1 plots the learning curves for our two methods when training with various amounts of parallel data.

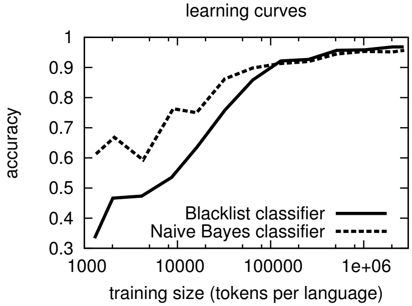

Figure 1: Learning curves of the proposed classifiers with various amounts of training data.

The figure shows that Naive Bayes is learning much quicker to distinguish between the three languages. This is not very surprising as the blacklist classifier only considers a fraction of the possible features included in the training data (see Table 6). However, after about 100,000 tokens per language, it surpasses the accuracy of the Naive Bayes classifier and performs consequently better thereafter.

It is also interesting to consider the learning curves for the individual languages. Looking a bit closer at the blacklist classifier (Figure 2), we can see that the main problem is the identification of Bosnian. It takes much more data to train a decent classifier for Bosnian than required for distinguishing the other two languages.

Interesting are also the drops in performance when recognizing Serbian in the beginning and when recognizing Croatian after about 10,000 tokens of training data. This development is due to our cascaded setup and the lack of evidence in the learned classification models. In the beginning, no (or only a few) blacklisted words are found and the classifier cannot discriminate between the three languages. Our method simply classifies every document as Serbian. After a few thousand tokens, the classifier learns to identify Croatian quickly but starts confusing it with Serbian and later with Bosnian. At about 10,000 tokens, the classifier improves significantly for Serbian again but seems to be more confused about Bosnian and Croatian before fixing most of these problems with larger amounts of training data.

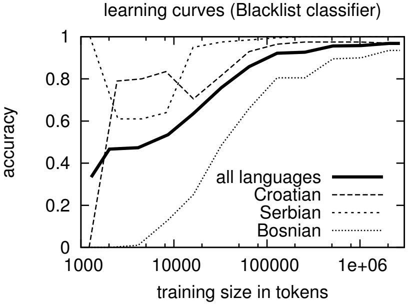

Figure 2: Learning curves for individual languages in the Blacklist classifier.

The learning curves above reveal one of the main weaknesses of the blacklist approach. It is not reliable with very little amounts of training data and the results will significantly depend on the setup of the binary decisions in that case. Table 6 lists the total number of tokens included in the blacklists when using default settings. For example, after about 32,000 tokens only a few hundred words are selected, which does not seem to be sufficient according to Figure 1. However, only a few thousand words selected at about 128,000 tokens of training material perform pretty well for all three languages. This is a very promising result. Furthermore, the amount of selected words for classification can certainly be adjusted by the extraction parameters α, β and γ. Initial experiments show that less restrictive parameters lead to better accuracies with small amounts of training data but lower overall performance when using the entire training set. This is also expected as the classifier becomes more similar to the Naive Bayes classifier and its parameters. We will leave a proper optimization of these model parameters to future work.

size in number of tokens		CPU time in seconds	
train data	blacklists	training	classification
1k	2	0.06	1.35
2k	5	0.08	1.38
4k	10	0.14	1.39
8k	32	0.27	1.40
16k	95	0.53	1.41
32k	361	1.01	1.40
64k	867	2.08	1.44
128k	1766	4.08	1.46
256k	3463	8.25	1.49
512k	6280	16.72	1.53
1M	10522	32.96	1.50
2M	16707	66.60	1.61
2.5M	19261	93.93	1.70

Table 6: Size of the learned blacklists and time spent for training and classification.

Table 6 also lists the time spent for training blacklist models with various amounts of training data and the time required for classifying our test set with those models. We can see that training is fast[10] even with our unoptimized code (based on a scripting language) and classification time is almost constant with some overhead because of the increasing size of the extracted models. The simplicity of the blacklist approach allows very efficient computation and produces compact models with high accuracy. This is certainly one of the major advantages of this approach compared to more sophisticated machine learning techniques.

4.2 Parallel versus Comparable Training Data

An additional question we like to consider in this paper is how robust our classifiers are with respect to non-parallel data. To simulate a loss in comparability we have split our trilingual dataset into three folds. In the first setting we train three models on parallel data while in the second setting we train models on the six permutations of non-parallel data so that for neither language there is any parallel data in the training set[11].

We evaluated each of the 3+6 models for both classifiers on our standard test set. The results are given in Table 7. With a loss of comparability of the training data we see an average decrease in performance of 1.6 points for the Naive Bayes classifier while there is no decrease in the case of the Blacklist classifier. The difference in the results of the two settings when using the Naive Bayes classifier is statistically significant according to the one-sided Student's t-test with unequal variance (p-value is 0.029). Furthermore, the difference between the results of the two classifiers on comparable corpora has become highly statistically significant (p-value below 0.001). Here we have identified a strong point of the Blacklist classifier – it is much more resistant to non-parallelity of the data when compared to the Naive Bayes classifier.

	Naive Bayes	Blacklist	p-value
parallel	0.953	0.963	0.233
comparable	0.937	0.963	**0.001**
p-value	**0.029**	0.875	

Table 7: Results for parallel and comparable training sets for both classifiers with the p-value from the Student's t-test.

The reason for such results can be sought in the fact that the Blacklist classifier does generalize more by selecting only the most informative features while this implementation of the Naive Bayes classifier does not perform any feature selection at all.

4.3 Sensitivity with Respect to Document Size

Finally, we would also like to investigate the influence of document size on classification performance. We expect that our word-based classifiers require larger amounts of input data for reliable classification. This is especially true for the blacklist approach that relies on strong discriminative features that might not be very frequent in all kinds of documents. Figure 3 illustrates the overall accuracy with varying document sizes. For this experiment, we selected all documents with more than 300 words from our initial test set (giving us 100 Bosnian

[10] Note that training speed is not as essential as classification speed as training is usually done once only.

[11] In the first setting we train on folds (0,0,0), (1,1,1) and (2,2,2) while in the second one we train on (0,1,2), (1,2,0) etc.

documents, 89 Croatian documents and 182 Serbian documents) and used the initial N words of each of them for classification.

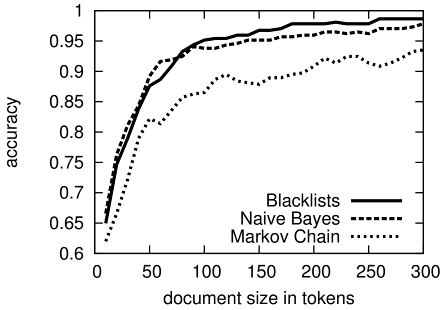

Figure 3: Classification performance with various text sizes.

As expected, we can see a significant decline of the accuracy with very short documents. However, already at about 70 words we have an overall performance of over 90%. Again, proper identification of Bosnian texts is the hardest task and about double as many words are needed to reach 90% accuracy for the Bosnian test set (not shown in Figure 3). In general, a modest document size of 150–200 words seems to be quite sufficient for our difficult task.

Another important result is that both classification approaches outperform our strongest baseline (the character-based Markov chain model) at all points. Furthermore, the blacklist approach is slightly worse on short documents, which was also to be expected.

4.4 Inspecting Language Discriminators

An interesting byproduct of both language identification approaches described above is that the parameters of their models are actually conditional probabilities and weights calculated on words and categories. This gives us the opportunity to inspect the strongest discriminators between these three languages and to classify them by the differences described in subsection 2.1.

In table 8 we list the twenty five highest conditional probabilities (and thereby strongest discriminators) of the final Naive Bayes classifier. In Bosnian there are only five conditional probabilities with a maximum value while in Croatian there are 34 of such words that do not appear in training corpora of other languages. In the Serbian model we have 21 maximum probabilities. This is in line with the previous claim from subsection 2.1 and table 1 that Croatian is the most specific language while Bosnian is a mixture of the other two.

The twenty five strongest Bosnian discriminators contain seven lexemes where Bosnian differs from Serbian regarding the ijekavian reflex like *obezbijediti* (*obezbediti* in Serbian) and *posjetioc* (*posetioc*). The latter form is written in Croatian with a different suffix morpheme – *posjetitelj*. Another difference to Serbian is the internationalism *historija* written *istorija* in Serbian while Croatian has its own word – *povijest*.

Bosnian		Croatian		Serbian	
sedmice	1.0	tjedna	1.0	evra	1.0
saopćenju	1.0	glede	1.0	sredu	1.0
izvještajima	1.0	izvješću	1.0	izveštaju	1.0
augusta	1.0	listopada	1.0	bezbednosti	1.0
saopćio	1.0	veljače	1.0	saveta	1.0
saopćila	0.999	siječnja	1.0	euleks	1.0
izvještaja	0.999	posebice	1.0	posete	1.0
obezbijediti	0.999	ožujka	1.0	bezbednost	1.0
sedmica	0.999	tvrtke	1.0	verovatno	1.0
saopćeno	0.999	prosinca	1.0	vestima	1.0
historiji	0.999	svibnja	1.0	predsednikom	1.0
istambulu	0.999	lipnja	1.0	savet	1.0
saopćili	0.999	srpnja	1.0	potpredsednik	1.0
unaprjeđivanju	0.999	rujna	1.0	cena	1.0
historijski	0.998	travnja	1.0	cene	1.0
historije	0.998	gospodarstva	1.0	vrednosti	1.0
augustu	0.998	rumunjskoj	1.0	dve	1.0
odista	0.998	tvrtka	1.0	organizovanog	1.0
historiju	0.998	izvješće	1.0	sledeće	1.0
posjetioci	0.998	priopćenju	1.0	zahtev	1.0
istambula	0.998	ravnatelj	1.0	ren	1.0
bezbjednost	0.998	gospodarstvo	1.0	nemačka	0.999
djelimično	0.998	priopćila	1.0	posetio	0.999
sedmicu	0.998	sustava	1.0	severnom	0.999
unaprjeđivanja	0.998	konca	1.0	poseti	0.999

Table 8: Twenty five highest conditional probabilities of the Naive Bayes classifier for each language

Croatian's strongest discriminators contain ten month names (*listopad*, *veljača* etc.) since Croatian uses its own names and Bosnian and Serbian use the international forms. The word *priopćiti* shows a difference in derivational morphology compared to Bosnian where we have *saopćiti*. The list contains also many words of Croatian origin like *ravnatelj*, *gospodarstvo* and *tvrtka* where in Bosnian and Serbian internationalisms or terms from the Yugoslav era are used.

The Serbian list contains many ekavian variants like *sreda* (written *srijeda* in Bosnian and Croatian), *savet* (*savjet*) and *vest* (*vijest*). A frequent morphological difference to Croatian and Bosnian is present in the word *organizovan* that would be formed as *organiziran* in these two languages.

It is important to note that not all differences caught by these models are actual differences in language use or language norm, but are sometimes just the result of the language policy applied on the website the dataset comes from. A good example is the word *izvještaj* from the Bosnian list that obviously never appears in any other language although it is regularly used in Croatian language and Croatian dictionaries define it as standard as well.

Conclusion and Future Work

In this paper, we have shown that language discrimination between closely related languages deserves special attention. Standard tools based on character sequence features are not sufficient for distinguishing languages with a large lexical overlap. We propose two token-based approaches, one based on a Naive Bayes classifier and one based on weighted lists of blacklisted words. Both perform very well and significantly outperform state-of-the-art approaches to language identification. Another conclusion from our experiments is that a Naive Bayes model performs better for smaller amounts of data but highly depends on the comparability of the language data it is trained on. The blacklist approach is similar in essence but includes heavy feature selection. This leads to a larger generalization of the model and makes it perform better on less parallel data sets. The overall performance of the blacklist approach is also higher given the entire data set we train on and improves the best baseline created using public language identification tools by over 9% absolute accuracy. The implementations of the two approaches along with the datasets used in the paper can be retrieved from
`http://www.nljubesic.net/resources/tools/bs-hr-sr-language-identifier/` and
`http://bitbucket.org/tiedemann/blacklist-classifier`.

In this work we were using parallel data, but did not exploit all the benefits one can expect from such a dataset. We did not use sentence and word alignments, but just the corpora as a whole having in mind that the frequencies of identical language elements across languages will agree very well. Using alignments from parallel corpora is one direction for future work.

Another direction is the opposite one: using weakly comparable, but larger corpora to obtain the same or better results. We have shown that the blacklist approach is very robust on strongly comparable corpora, but additional experiments are necessary to examine how it copes with lower comparability of the training data. On the other hand, the Naive Bayes classifier could be made less prone to non-parallelity as well by applying different feature selection methods.

Furthermore, using very large web corpora such as the hrWaC corpus (Ljubešić and Erjavec, 2011) 1.2 billion words in size and the bsWaC and srWaC corpora (under construction) opens the door for catching most of the token and N-gram domain-independent variation between closely related languages that could yield extremely high results we are used to for more distinct languages.

Finally, we would also like to look at recently proposed character-based language models that have successfully been used for the discrimination between language varieties (Zampieri and Gebre, 2012). Their approach is similar to our Markov chain baseline but uses larger character N-grams that often cover entire words. One of the disadvantages of this approach is the increase of the number of model parameters, blowing up the model size and causing a slower classification speed. However, one could hope for higher generalization and resistance to non-parallelity of the training data.

Acknowledgements

Research reported in this paper has been supported by the EU FP7 ICT-PSP project LetsMT!, grant agreement no. 250456, and the EU FP7 STREP project XLike, grant agreement no. 288342.

References

Batchelder, E. O. (1992). A learning experience: Training an artificial neural network to discriminate languages. Unpublished technical report.

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

Hoeffding, W. (1952). The large-sample power of tests based on permutations of observations. *Annals of Mathematical Statistics*, 23(2):169–192.

Ljubešić, N. and Erjavec, T. (2011). hrWaC and slWac: Compiling Web Corpora for Croatian and Slovene. In Habernal, I. and Matousek, V., editors, *Text, Speech and Dialogue - 14th International Conference, TSD 2011, Pilsen, Czech Republic, September 1-5, 2011. Proceedings*, volume 6836 of *Lecture Notes in Computer Science*, pages 395–402. Springer.

Ljubešić, N., Mikelić, N., and Boras, D. (2007). Language identification: How to distinguish similar languages. In Lužar-Stifter, V. and Hljuz Dobrić, V., editors, *Proceedings of the 29th International Conference on Information Technology Interfaces*, pages 541–546, Zagreb. SRCE University Computing Centre.

Lui, M. and Baldwin, T. (2011). Cross-domain feature selection for language identification. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 553–561, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Lui, M. and Baldwin, T. (2012). langid.py: An off-the-shelf language identification tool. In *ACL (System Demonstrations)*, pages 25–30.

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

McCallum, A. and Nigam, K. (1998). A comparison of event models for naive bayes text classification. In *IN AAAI-98 WORKSHOP ON LEARNING FOR TEXT CATEGORIZATION*, pages 41–48. AAAI Press.

Padró, L. and Padró, M. (2004). Comparing methods for language identification. *Procesamiento del Lenguaje Natural*, (33):155–162.

Yeh, A. (2000). More accurate tests for the statistical significance of result differences. In *Proceedings of the 18th conference on Computational linguistics - Volume 2*, COLING '00, pages 947–953, Stroudsburg, PA, USA. Association for Computational Linguistics.

Zampieri, M. and Gebre, B. (2012). Automatic identification of language varieties: The case of Portuguese. In Jancsary, J., editor, *Proceedings of KONVENS 2012*, pages 233–237. ÖGAI. Main track: poster presentations.