HeLI, a Word-Based Backoff Method for Language Identification

Tommi Jauhiainen
University of Helsinki
@helsinki.fi

Krister Lindén
University of Helsinki
@helsinki.fi

Heidi Jauhiainen
University of Helsinki
@helsinki.fi

Abstract

In this paper we describe the Helsinki language identification method, HeLI, and the resources we created for and used in the 3rd edition of the Discriminating between Similar Languages (DSL) shared task, which was organized as part of the VarDial 2016 workshop. The shared task comprised of a total of 8 tracks, of which we participated in 7. The shared task had a record number of participants, with 17 teams providing results for the closed track of the test set A. Our system reached the 2nd position in 4 tracks (A closed and open, B1 open and B2 open) and in this paper we are focusing on the methods and data used for those tracks. We describe our word-based backoff method in mathematical notation. We also describe how we selected the corpus we used in the open tracks.

1 Introduction

The 3rd edition of the Discriminating between Similar Languages (DSL) shared task, (Malmasi et al., 2016), was divided into two sub-tasks: ”Similar Languages and Language Varieties” and ”Arabic dialects”. Furthermore, the first sub-task was divided into three test sets: A, B1 and B2. Each of the test sets for both tasks had a closed and an open track. On the closed track the participants were allowed to use only the training data provided by the organizers, whereas on the open track the participants could use any data source they had at their disposal.

The first sub-task had a language selection comparable to the 1st (Zampieri et al., 2014) and 2nd (Zampieri et al., 2015b) editions of the shared task. The languages and varieties of the sub-task 1 are listed in the Table 2. The differences from the previous year’s shared task were the inclusion of the French language varieties and the Mexican Spanish, as well as the exclusion of Bulgarian, Macedonian, Czech, and Slovak. The four latter languages were practically 100% correct in most of the submissions to the 2nd edition of the shared task. On the other hand, discriminating between the two French varieties could be expected to be more difficult (Zampieri, 2013). Also the extra category ”unknown language” introduced in 2015 was left out from the 3rd edition repertoire. These changes resulted in a drop of the best reported accuracy of any team (test set A) from the 95.54% of the 2nd edition closed track to the 89.38% of the 3rd edition closed track. The second sub-task comprised of discriminating between Modern Standard Arabic and four dialects: Egyptian, Gulf, Levantine, and North-African. The Arabic dialects were included in the shared task for the first time.

For the 2015 edition of the task, we used the word-based backoff language identification method first introduced in 2010 (Jauhiainen, 2010) and made several modifications to it in order to improve the method for the task of discriminating similar languages and to cope with the unknown language (Jauhiainen et al., 2015b). In the 3rd edition of the task, the unknown language was left out, which meant that the original method was directly applicable. We also felt that the modifications we made in 2015 complicated the system and did not really improve the results that much, so we decided to use the basic method out-of-the-box for the 3rd edition of the shared task. The word-based backoff method, now named HeLI, is a general purpose language identification method which we have used for collecting text

Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects,
pages 153–162, Osaka, Japan, December 12 2016.

material written in Uralic languages in the Finno-Ugric Languages and the Internet project (Jauhiainen et al., 2015a) funded by the Kone foundation. We have also used the method as a language identifier part when developing a method for language set identification in multilingual documents (Jauhiainen et al., 2015c). The language identifier tool using the HeLI-method is available as open source from GitHub[1].

2 Related Work

Automatic language identification has been researched for more than 50 years. The first article on the subject was written by Mustonen (1965). For the history of automatic language identification (of textual material), as well as an excellent overview of the subject, the reader is suggested to take a look at the literature review chapter of Marco Lui's doctoral thesis (Lui, 2014). Recent surveys and overviews by Garg et al. (2014) and Shashirekha (2014) could also be of interest.

Automatic identification of Malay and Indonesian was studied by Ranaivo-Malançon (2006). Distinguishing between South-Slavic languages has been researched by Ljubešic et al. (2007), Tiedemann and Ljubešic (2012), Ljubešic and Kranjcic (2014), and Ljubešic and Kranjcic (2015). Automatic identification of Portuguese varieties was studied by Zampieri and Gebre (2012), whereas Zampieri et al. (2012), Zampieri (2013), Zampieri et al. (2013), and Maier and Gómez-Rodríguez (2014) researched language variety identification between Spanish dialects. Discriminating between French dialects was studied by Zampieri et al. (2012) and Zampieri (2013). Arabic dialect identification was researched by Elfardy and Diab (2013), Darwish et al. (2014), Elfardy et al. (2014), Sadat et al. (2014), Salloum et al. (2014), Tillmann et al. (2014), Zaidan and Callison-Burch (2014), Al-Badrashiny et al. (2015), Malmasi et al. (2015), and Ali et al. (2016).

The system description articles provided for the previous shared tasks are all relevant and references to them can be found in (Zampieri et al., 2014) and (Zampieri et al., 2015b). Detailed analysis of the previous shared task results was done by Goutte et al. (2016).

3 Methodology

The basic idea of the HeLI method was first introduced in (Jauhiainen, 2010). It was also described in the proceedings of the previous task (Jauhiainen et al., 2015b). In this paper, we present the complete description of the method for the first time. First we introduce the notation used in the description of the method.

3.1 On notation [2]

A corpus C consists of individual tokens u which may be words or characters. A corpus C is a finite sequence of individual tokens, $u_1, ..., u_l$. The total count of all individual tokens u in the corpus C is denoted by l_C. A feature f is some countable characteristic of the corpus C. When referring to all features F in a corpus C, we use C^F and the count of all features is denoted by l_{C^F}. The count of a feature f in the corpus C is referred to as $c(C, f)$. An n-gram is a feature which consists of a sequence of n individual tokens. An n-gram starting at position i in a corpus is denoted $u_{i,...,i-1+n}$. If $n = 1$, u is an individual token. When referring to all n-grams of length n in a corpus C, we use C^n and the count of all such n-grams is denoted by l_{C^n}. The count of an n-gram u in a corpus C is referred to as $c(C, u)$ and is defined by Equation 1.

$$c(C,u) = \sum_{i=1}^{l_C+1-n} \begin{cases} 1 & \text{, if } u = u_{i,...,i-1+n} \\ 0 & \text{, otherwise} \end{cases} \tag{1}$$

if a token, aka. an n-

The set of languages is G, and l_G denotes the number of languages. A corpus C in language g is denoted by C_g. A language model O based on C_g is denoted by $O(C_g)$. The features given values by the model $O(C_g)$ are the domain $dom(O(C_g))$ of the model. In a language model, a value v for the feature f is denoted by $v_{C_g}(f)$. For each potential language g of a corpus C in an unknown language, a resulting score $R_g(C)$ is calculated. A corpus in an unknown language is also referred to as a mystery text.

[1] https://github.com/tosaja/HeLI

[2] We would like to thank Kimmo Koskenniemi for many valuable discussions and comments.

3.2 HeLI method

The goal is to correctly guess the language $g \in G$ in which the monolingual mystery text M has been written, when all languages in the set G are known to the language identifier. In the method, each language $g \in G$ is represented by several different language models only one of which is used for every word t found in the mystery text M. The language models for each language are: a model based on words and one or more models based on character n-grams from one to n_{max}. Each model used is selected by its applicability to the word t under scrutiny. The basic problem with word-based models is that it is not really possible to have a model with all possible words. When we encounter an unknown word in the mystery text M, we back off to using the n-grams of the size n_{max}. The problem with high order n-grams is similar to the problem with words: there are simply too many of them to have statistics for all. If we are unable to apply the n-grams of the size n_{max}, we back off to lower order n-grams. We continue backing off until character unigrams, if needed.

A development set is used for finding the best values for the parameters of the method. The three parameters are the maximum length of the used character n-grams (n_{max}), the maximum number of features to be included in the language models (cut-off c), and the penalty value for those languages where the features being used are absent (penalty p). The penalty value has a smoothing effect in that it transfers some of the probability mass to unseen features in the language models.

3.3 Description of the method

The task is to select the most probable language g, given a mystery text M, as shown in Equation 2.

$$argmax_g P(g|M) \tag{2}$$

$P(g|M)$ can be calculated using the Bayes' rule, as in Equation 3.

$$P(g|M) = \frac{P(M|g)P(g)}{P(M)} \tag{3}$$

In Equation 3, $P(M)$ is equal for the languages $g \in G$ and can be omitted. Also, we assume that all languages have equal a priori probability, so that $P(g)$ can be omitted as well, leaving us with the Equation 4.

$$argmax_g P(g|M) = argmax_g P(M|g) \tag{4}$$

We approximate the probability $P(M|g)$ of the whole text through the probabilities of its words $P(t|g)$, which we assume to be independent as in Equation 5.

$$P(M|g) \approx P(t_1|g)P(t_2|g)...P(t_{l_M}|g) \tag{5}$$

We use the relative frequencies of words and character n-grams in the models for language g for estimating the probabilities $P(t|g)$.

3.3.1 Creating the language models

The training data is lowercased and tokenized into words using non-alphabetic and non-ideographic characters as delimiters. The relative frequencies of the words are calculated. Also the relative frequencies of character n-grams from 1 to n_{max} are calculated inside the words, so that the preceding and the following space-characters are included. The n-grams are overlapping, so that for example a word with three characters includes three character trigrams. Word n-grams are not used in this method, so all subsequent references to n-grams in this article refer to n-grams of characters.

The c most common n-grams of each length and the c most common words in the corpus of a language are included in the language models for that language. We estimate the probabilities using relative frequencies of the words and character n-grams in the language models, using only the relative frequencies of the retained tokens. Then we transform those frequencies into scores using 10-based logarithms.

The derived corpus containing only the word tokens retained in the language models is called $C^{'}$. $dom(O(C^{'}))$ is the set of all words found in the models of all languages $g \in G$. For each word $t \in dom(O(C^{'}))$, the values $v_{C^{'}_{g}}(t)$ for each language g are calculated, as in Equation 6

$$v_{C^{'}_{g}}(t) = \begin{cases} -\log_{10}\left(\frac{c(C^{'}_{g},t)}{l_{C^{'}_{g}}}\right) & \text{, if } c(C^{'}_{g},t) > 0 \\ p & \text{, if } c(C^{'}_{g},t) = 0 \end{cases} \tag{6}$$

where $c(C^{'}_{g},t)$ is the number of words t and $l_{C^{'}_{g}}$ is the total number of all words in language g. If $c(C^{'}_{g},t)$ is zero, then $v_{C^{'}_{g}}(t)$ gets the penalty value p.

The derived corpus containing only the n-grams retained in the language models is called $C^{'n}$. The domain $dom(O(C^{'n}))$ is the set of all character n-grams of length n found in the models of all languages $g \in G$. The values $v_{C^{'n}_{g}}(u)$ are calculated similarly for all n-grams $u \in dom(O(C^{'n}))$ for each language g, as shown in Equation 7

$$v_{C^{'n}_{g}}(u) = \begin{cases} -\log_{10}\left(\frac{c(C^{'n}_{g},u)}{l_{C^{'n}_{g}}}\right) & \text{, if } c(C^{'n}_{g},u) > 0 \\ p & \text{, if } c(C^{'n}_{g},u) = 0 \end{cases} \tag{7}$$

where $c(C^{'n}_{g},u)$ is the number of n-grams u found in the derived corpus of the language g and $l_{C^{'n}_{g}}$ is the total number of the n-grams of length n in the derived corpus of language g. These values are used when scoring the words while identifying the language of a text.

3.3.2 Scoring n-grams in the mystery text

When using n-grams, the word t is split into overlapping n-grams of characters u^{n}_{i}, where $i = 1, ..., l_t - n$, of the length n. Each of the n-grams u^{n}_{i} is then scored separately for each language g in the same way as the words.

If the n-gram u^{n}_{i} is found in $dom(O(C^{'n}_{g}))$, the values in the models are used. If the n-gram u^{n}_{i} is not found in any of the models, it is simply discarded. We define the function $d_g(t,n)$ for counting n-grams in t found in a model in Equation 8.

$$d_g(t,n) = \sum_{i=1}^{l_t - n} \begin{cases} 1 & \text{, if } u^{n}_{i} \in dom(O(C'^{n})) \\ 0 & \text{, otherwise} \end{cases} \tag{8}$$

When all the n-grams of the size n in the word t have been processed, the word gets the value of the average of the scored n-grams u^{n}_{i} for each language, as in Equation 9

$$v_g(t,n) = \begin{cases} \frac{1}{d_g(t,n)} \sum_{i=1}^{l_t - n} v_{C^{'n}_{g}}(u^{n}_{i}) & \text{, if } d_g(t,n) > 0 \\ v_g(t, n-1) & \text{, otherwise} \end{cases} \tag{9}$$

where $d_g(t,n)$ is the number of n-grams u^{n}_{i} found in the domain $dom(O(C^{'n}_{g}))$. If all of the n-grams of the size n were discarded, $d_g(t,n) = 0$, the language identifier backs off to using n-grams of the size $n-1$. If no values are found even for unigrams, a word gets the penalty value p for every language, as in Equation 10.

$$v_g(t,0) = p \tag{10}$$

3.3.3 Language identification

The characters in the mystery text are lowercased, after which the text is tokenized into words using the non-alphabetic and non-ideographic characters as delimiters. After this, a score $v_g(t)$ is calculated for each word t in the mystery text for each language g. If the word t is found in the set of words $dom(O(C^{'}_{g}))$, the corresponding value $v_{C^{'}_{g}}(t)$ for each language g is assigned as the score $v_g(t)$, as shown in Equation 11.

$$v_g(t) = \begin{cases} v_{C'_g}(t) & \text{, if } t \in dom(O(C'_g)) \\ v_g(t, min(n_{max}, l_t + 2)) & \text{, if } t \notin dom(O(C'_g)) \end{cases} \tag{11}$$

If a word t is not found in the set of words $dom(O(C'_g))$ and the length of the word l_t is at least $n_{max} - 2$, the language identifier backs off to using character n-grams of the length n_{max}. In case the word t is shorter than $n_{max} - 2$ characters, $n = l_t + 2$.

The whole mystery text M gets the score $R_g(M)$ equal to the average of the scores of the words $v_g(t)$ for each language g, as in Equation 12

$$R_g(M) = \frac{\sum_{i=1}^{l_{T(M)}} v_g(t_i)}{l_{T(M)}} \tag{12}$$

where $T(M)$ is the sequence of words and $l_{T(M)}$ is the number of words in the mystery text M. Since we are using negative logarithms of probabilities, the language having the lowest score is returned as the language with the maximum probability for the mystery text.

4 Data

Creation of the earlier DSL corpora has been described by Tan et al. (2014). The training data for the test sets A and B consisted of 18,000 lines of text for each of 12 languages. The corresponding development set had 2,000 lines of text for each language. The training data for the test set C had 7,619 lines for all of the five varieties of the Arabic language and there was no separate development set available. The training and the tests sets for Arabic were produced using automatic speech recognition software (Ali et al., 2016). The amount of training data was different for each variety of Arabic as can be seen in Table 1.

Arabic variety	Number of lines
Modern Standard Arabic	999
Egyptian Arabic	1,578
Gulf Arabic	1,672
Levantine Arabic	1,758
North-African Arabic	1,612

Table 1: The number of lines of training material available for the Arabic varieties.

Test set A consisted of excerpts of journalistic texts similar to the training data provided for the task and the test sets B1 and B2 consisted of Bosnian, Croatian, Serbian, Brazilian Portuguese and European Portuguese tweets. Both the test sets B1 and B2 were formed out of tweets so that several tweets from the same user had been concatenated on one line, separated by a tab-character. The exact nature and format of the B1 and B2 test sets was revealed only a few days before the results were due to be returned. Before that the test set B had been characterized as an out-of-domain social media data. The test sets included a lot of material almost unique to the format of tweets. Without any prior experience on automatically handling tweets, it was very difficult to process them.

For the open tracks of test sets A, B1, and B2 we created a new corpus for each language. We collected from the Common Crawl [3] corpus all the web pages from the respective domains as in Table 2. When language models were created directly from the pages, the accuracy on the DSL development corpus was 49.86%, which was much lower than the 85.09% attained with the DSL training corpus. We used several ad-hoc techniques to improve the quality of the corpus.

The shortest sensible sentence in the development corpus was 25 characters, so we first removed all the lines shorter than that from our open track corpus. The accuracy rose to 51.08%. Then we removed all lines that did not include one of the top 5 characters (in the DSL training data) for the language in question. Furthermore, we only kept the lines which included at least one of the top-5 words with at least 2 characters of the respective language. With these adjustments, the accuracy rose to 62.42%. Moreover, we created lists of characters starting and ending lines in the DSL training corpus.

[3] http://commoncrawl.org/

Domain ending	Country	Language	Size, raw (tokens)	Size, final (tokens)
.ba	Bosnia and Herzegovina	Bosnian	41,400,000	5,500,000
.hr	Croatia	Croatian	282,700,000	9,700,000
.rs	Serbia	Serbian	148,300,000	12,600,000
.my	Malaysia	Malay	239,700,000	8,100,000
.id	Indonesia	Indonesian	549,700,000	35,100,000
.br	Brazil	Portuguese	3,689,300,000	264,500,000
.pt	Portugal	Portuguese	307,000,000	13,400,000
.ar	Argentina	Spanish	909,900,000	27,500,000
.mx	Mexico	Spanish	1,092,400,000	51,000,000
.es	Spain	Spanish	2,865,900,000	46,200,000
.fr	France	French	4,878,600,000	240,800,000
.ca	Canada	French	7,414,500,000	13,600,000

Table 2: The languages and varieties of the sub-task 1 and the collected domains for the corpus used in the open tracks.

We chose almost all of the characters from both categories and kept only the lines starting and ending with those characters. We then sorted all the lines alphabetically and removed duplicates. Furthermore, we made a character set out of the whole DSL training corpus (all languages in one) and removed all lines that had characters which were not in the character set. After these changes we managed to get an accuracy of 68.34%. Moreover, we used the language identifier service that we had set up for the SUKI project web crawler, with almost 400 languages and dialects, and identified the language of each line. If Canadian or French lines were in French, they were accepted and so on also for the other languages and dialects. The accuracy rose to 69.19%. Subsequently, we instead used the language models created from the DSL training data and kept only the lines which were identified as the proper language or dialect. The accuracy rose to 74.66%. These accuracies were attained using language models with cut-off of 10,000 tokens. We did some optimizing and ended up with a cut-off of 75,000 tokens which gave us an accuracy of 80.93%. Additionally, we created a very simple sentence detection program and divided the corpora into sentences, keeping only complete sentences from each line with each sentence on its own line. Furthermore, we again removed all lines shorter than 25 characters after which we identified the lines using the project language identifier keeping only the lines identified with correct languages. Moreover, we identified the lines again using the DSL models and kept the lines identified with the corresponding dialect or language. The accuracy was now 83.15%. Subsequently, we again sorted the lines alphabetically and removed duplicates and after some optimizing of the parameters (using 100,000 tokens in the language models) the accuracy was 84.90%. The sizes of each language in the final corpus we created can be seen in the ”Size, final (Gb)” column of the Table 2. In hindsight, there would be more straightforward ways to end up with the same corpus. By doing some of the before mentioned steps in another order, some other steps could be omitted completely, but we did not have time to redo the corpora creation process within the time constraints of the shared task.

Then we added the DSL training data to the corpora we created and the results on the development improved. We also tried to add the relevant parts of the 2nd edition of the shared task corpus, but including them did not improve the results on the development set. Instead, we finally added also the development material to the corpus to create the final language models for the open tracks of the test sets A, B1 and B2.

5 Results

In order to find the best possible parameters (n_{max}, c, and p), and language models for words (lowercased or not), we applied a simple form of the greedy algorithm separately for each development set. The use of capital letter words is not detailed in the description of the method. However, the language identifier begins with a language model for words which includes capital letters and if it is not applicable it backs off to using lowercased models for words and so on. The parameters for each run are included in Tables 3-8. We have also included the best results and the name of the winning team in each category.

5.1 Sub-task1

5.1.1 Test set A

For the test set A we only did one run for each of the closed and the open tracks. The results can be seen in the Table 3. On the closed track we used all of the training and the development data to create the language models.

Run	Accuracy	F1 (macro)	n_{max}	c	p	Cap. words	Low. words
SUKI closed	0.8879	0.8877	6	120,000	6.6	no	yes
tubasfs closed	0.8938	0.8938					
SUKI open	0.8837	0.8835	7	180,000	8.2	yes	yes
nrc open	0.8903	0.8889					

Table 3: Results for test set A (closed training).

5.1.2 Test set B1

For the test set B1 we did three runs on both the closed and the open tracks.

Closed training After the first two runs with the basic HeLI method, for the third run we used the unknown language detection thresholds we came up with in the 2015 edition of the shared task (Jauhiainen et al., 2015b). We first identified each tweet separately and removed all tweets that were supposedly in an unknown language. Then we identified the tweets that were left as one line. The results can be seen in Table 4.

Run	Accuracy	F1 (macro)	n_{max}	c	p	Cap. words	Low. words
SUKI run1	0.68	0.662	6	110,000	6.5	no	yes
SUKI run2	0.676	0.6558	0	110,000	6.5	no	yes
SUKI run3	0.688	0.6719	8	2,000,000	6.6	yes	yes
GWU_LT3	0.92	0.9194					

Table 4: Results for test set B1 (closed training).

Open training For the first run we did not do any preprocessing. Before the second run, we used the language identifier set up for our web crawler to remove those individual tweets that it detected to be of non-relevant language. For the third run we also removed all the http- and https-addresses from the tweets to be tested. The results can be seen in the Table 5.

Run	Accuracy	F1 (macro)	n_{max}	c	p	Cap. words	Low. words
SUKI run1	0.714	0.6999	6	180,000	8.1	yes	yes
SUKI run2	0.806	0.7963	8	2,000,000	6.6	yes	yes
SUKI run3	0.822	0.815	8	2,000,000	6.6	yes	yes
nrc	0.948	0.948					

Table 5: Results for test set B1 (open training).

5.1.3 Test set B2

For the test set B2 we did two runs on both the closed and the open tracks. On the second run of both tracks, our language identifier occasionally returned an unknown language as a result of our preprocessing that had emptied some of the lines completely. In order to comply with the way our results were handled by the shared task organizers, we used the 'pt-PT' which was the language identified for the majority of the lines with the unknown language in the first runs. The correct way to handle this problem would have been to put the exact answers from the first runs as the unknown language, but there was no time for this. The effects on the results should anyway be only fractions of a percent.

Closed training For the first run we did not do any preprocessing, but for the second run we used the unknown language detection in the same way as in the B1 closed track run 3. From the results in Table 6, it can be seen that this actually lowered the identification accuracy.

Run	Accuracy	F1 (macro)	n_{max}	*c*	*p*	Cap. words	Low. words
SUKI run1	0.642	0.6229	6	110,000	6.5	no	yes
SUKI run2	0.614	0.5991	6	110,000	6.5	no	yes
GWU_LT3	0.878	0.8773					

Table 6: Results for test set B2 (closed training).

Open training The results for the B2 open track can be seen in Table 7. For the first run we did not do any preprocessing. For the second run we used the language identifier set up for our web crawler to remove those individual tweets that it detected to be of non-relevant language. We also removed all the http- and https-addresses from the tweets to be tested.

Run	Accuracy	F1 (macro)	n_{max}	*c*	*p*	Cap. words	Low. words
SUKI run1	0.75	0.7476	6	180,000	8.1	yes	yes
SUKI run2	0.796	0.7905	8	2,000,000	6.6	yes	yes
nrc	0.9	0.9					

Table 7: Results for test set B2 (open training).

5.2 Sub-task2

For the sub-task 2 we made only one run on the closed track. The character *n*-grams in the language models created for the test set C also included capital letters due to the nature of the corpus, unlike in the regular HeLI method where the character *n*-grams are created from lowercased words. The results can be seen in Table 8.

Run	Accuracy	F1 (macro)	n_{max}	*c*	*p*	Cap. words	Low. words
SUKI run1	0.4883	0.4797	8	5,000	4.6	yes	no
MAZA	0.5117	0.5132					

Table 8: Results for test set C (closed training).

The best accuracy on the closed track was 51.17% and that of the open track 52.18%. Our system came 7th on the closed track with 48.83% accuracy.

6 Discussion

Seventeen teams provided results for the closed track of test set A, which is quite a large increase over the 9 teams of the previous year. We were surprised to achieve the second place in this track, considering that we did not really try to improve the system from the last year's shared task, where we were in the 4th place. Instead, we made it simpler than last year, leaving out the extra discriminating features as well as the first stage of language group identification. As of this writing, we do not have much information on the nature of the language identification methods the other teams used, so we can only compare our method with the methods used in the previous task. The winner of the 2015 shared task used Support Vector Machines (SVMs), which heavily rely on finding the discriminating features (Malmasi and Dras, 2015). SVMs were also used by the NRC (Goutte and Leger, 2015) and MMS (Zampieri et al., 2015a) teams, which shared the second place last year. The language identification method we propose is generative in nature. It does not rely on finding discriminating features between languages. The language models for each language can be built without any knowledge of the other languages to be included in the repertoire of the language identifiers. This makes adding more languages to the language identifier very easy, there is no need to change the already existing models or to compare the new language with the already existing ones. It is possible that the generative nature gives our method more robustness in the case that the development and test data are not from exactly the same source. We suspect that the reason that we did not fare so well with the test sets B1 and B2 is mostly our inability to handle the format of the tweets well enough. It would have been interesting to see how our method would have succeeded in an out-of-domain test without the preprocessing challenges.

References

Mohamed Al-Badrashiny, Heba Elfardy, and Mona Diab. 2015. Aida2: A hybrid approach for token and sentence level dialect identification in arabic. In *Proceedings of the 19th Conference on Computational Language Learning*, pages 42–51, Beijing, China.

Ahmed Ali, Najim Dehak, Patrick Cardinal, Sameer Khurana, Sree Harsha Yella, James Glass, Peter Bell, and Steve Renals. 2016. Automatic dialect detection in arabic broadcast speech. In *Interspeech 2016*, pages 2934–2938.

Kareem Darwish, Hassan Sajjad, and Hamdy Mubarak. 2014. Verifiably effective arabic dialect identification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1465–1468, Doha, Qatar.

Heba Elfardy and Mona Diab. 2013. Sentence level dialect identification in arabic. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 456–461, Sofia.

Heba Elfardy, Mohamed Al-Badrashiny, and Mona Diab. 2014. Aida: Identifying code switching in informal arabic text. In *Proceedings of The First Workshop on Computational Approaches to Code Switching*, pages 94–101, Doha, Qatar.

Archana Garg, Vishal Gupta, and Manish Jindal. 2014. A survey of language identification techniques and applications. *Journal of Emerging Technologies in Web Intelligence*, 6(4):388–400.

Cyril Goutte and Serge Leger. 2015. Experiments in discriminating similar languages. In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects (LT4VarDial)*, pages 78–84, Hissar, Bulgaria.

Cyril Goutte, Serge Léger, Shervin Malmasi, and Marcos Zampieri. 2016. Discriminating Similar Languages: Evaluations and Explorations. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016).*

Heidi Jauhiainen, Tommi Jauhiainen, and Krister Lindén. 2015a. The finno-ugric languages and the internet project. *Septentrio Conference Series*, 0(2):87–98.

Tommi Jauhiainen, Heidi Jauhiainen, and Krister Lindén. 2015b. Discriminating similar languages with token-based backoff. In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects (LT4VarDial)*, pages 44–51, Hissar, Bulgaria.

Tommi Jauhiainen, Krister Lindén, and Heidi Jauhiainen. 2015c. Language Set Identification in Noisy Synthetic Multilingual Documents. In *Proceedings of the Computational Linguistics and Intelligent Text Processing 16th International Conference, CICLing 2015*, pages 633–643, Cairo, Egypt.

Tommi Jauhiainen. 2010. Tekstin kielen automaattinen tunnistaminen. Master's thesis, University of Helsinki, Helsinki, Finland.

Nikola Ljubešic and Denis Kranjcic. 2014. Discriminating between very similar languages among twitter users. In *Proceedings of the Ninth Language Technologies Conference*, pages 90–94, Ljubljana, Slovenia.

Nikola Ljubešic and Denis Kranjcic. 2015. Discriminating between closely related languages on twitter. *Informatica*, 39.

Nikola Ljubešic, Nives Mikelic, and Damir Boras. 2007. Language indentification: How to distinguish similar languages? In *Information Technology Interfaces, 2007. ITI 2007. 29th International Conference on*, pages 541–546, Cavtat/Dubrovnik, Croatia.

Marco Lui. 2014. *Generalized language identification*. Ph.D. thesis, The University of Melbourne.

Wolfgang Maier and Carlos Gómez-Rodríguez. 2014. Language variety identification in spanish tweets. In *Proceedings of the EMNLP'2014 Workshop: Language Technology for Closely Related Languages and Language Variants (LT4CloseLang 2014)*, pages 25–35, Doha, Qatar.

Shervin Malmasi and Mark Dras. 2015. Language identification using classifier ensembles. In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects (LT4VarDial)*, pages 35–43, Hissar, Bulgaria.

Shervin Malmasi, Eshrag Refaee, and Mark Dras. 2015. Arabic Dialect Identification using a Parallel Multidialectal Corpus. In *Proceedings of the 14th Conference of the Pacific Association for Computational Linguistics (PACLING 2015)*, pages 209–217, Bali, Indonesia, May.

Shervin Malmasi, Marcos Zampieri, Nikola Ljubešić, Preslav Nakov, Ahmed Ali, and Jörg Tiedemann. 2016. Discriminating between similar languages and arabic dialect identification: A report on the third dsl shared task. In *Proceedings of the 3rd Workshop on Language Technology for Closely Related Languages, Varieties and Dialects (VarDial)*, Osaka, Japan.

Seppo Mustonen. 1965. Multiple discriminant analysis in linguistic problems. *Statistical Methods in Linguistics*, 4:37–44.

Bali Ranaivo-Malançon. 2006. Automatic identification of close languages–case study: Malay and indonesian. *ECTI Transaction on Computer and Information Technology*, 2(2):126–133.

Fatiha Sadat, Farnazeh Kazemi, and Atefeh Farzindar. 2014. Automatic identification of arabic language varieties and dialects in social media. In *Proceedings of the Second Workshop on Natural Language Processing for Social Media (SocialNLP)*, pages 22–27, Dublin, Ireland.

Wael Salloum, Heba Elfardy, Linda Alamir-Salloum, Nizar Habash, and Mona Diab. 2014. Sentence level dialect identification for machine translation system selection. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Short Papers)*, pages 772–778, Baltimore, USA.

H. L. Shashirekha. 2014. Automatic language identification from written texts - an overview. *International Journal of Innovative Research in Computer and Communication Engineering*, 2(5):156–160.

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*, Reykjavik.

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

Christoph Tillmann, Yaser Al-Onaizan, and Saab Mansour. 2014. Improved sentence-level arabic dialect classification. In *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects*, pages 110–119, Dublin, Ireland.

Omar F. Zaidan and Chris Callison-Burch. 2014. Arabic dialect identification. *Computational Linguistics*, 40(1):171–202.

Marcos Zampieri and Binyam Gebrekidan Gebre. 2012. Automatic identification of language varieties: The case of portuguese. In *11th Conference on Natural Language Processing (KONVENS) - Empirical Methods in Natural Language Processing - Proceedings of the Conference on Natural Language Processing 2012*, pages 233–237, Vienna.

Marcos Zampieri, Binyam Gebrekidan Gebre, and Sascha Diwersy. 2012. Classifying pluricentric languages: Extending the monolingual model. In *Proceedings of the Fourth Swedish Language Technlogy Conference (SLTC2012)*, pages 79–80, Lund.

Marcos Zampieri, Binyam Gebrekidan Gebre, and Sascha Diwersy. 2013. N-gram language models and pos distribution for the identification of spanish varieties. In *Actes de TALN'2013 : 20e conférence sur le Traitement Automatique des Langues Naturelles*, pages 580–587, Sables d'Olonne.

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 (VarDial)*, pages 58–67, Dublin, Ireland.

Marcos Zampieri, Binyam Gebrekidan Gebre, Hernani Costa, and Josef van Genabith. 2015a. Comparing approaches to the identification of similar languages. In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects (LT4VarDial)*, pages 66–72, Hissar, Bulgaria.

Marcos Zampieri, Liling Tan, Nikola Ljubešić, Jörg Tiedemann, and Preslav Nakov. 2015b. Overview of the dsl shared task 2015. In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects (LT4VarDial)*, pages 1–9, Hissar, Bulgaria.

Marcos Zampieri. 2013. Using bag-of-words to distinguish similar languages: How efficient are they? In *Computational Intelligence and Informatics (CINTI), 2013 IEEE 14th International Symposium on*, pages 37–41, Budapest.