Iterative Language Model Adaptation for Indo-Aryan Language Identification

Tommi Jauhiainen
University of Helsinki
@helsinki.fi

Heidi Jauhiainen
University of Helsinki
@helsinki.fi

Krister Lindén
University of Helsinki
@helsinki.fi

Abstract

This paper presents the experiments and results obtained by the SUKI team in the Indo-Aryan Language Identification shared task of the VarDial 2018 Evaluation Campaign. The shared task was an open one, but we did not use any corpora other than what was distributed by the organizers. A total of eight teams provided results for this shared task. Our submission using a HeLI-method based language identifier with iterative language model adaptation obtained the best results in the shared task with a macro F1-score of 0.958.

1 Introduction

In the past, the VarDial workshops have hosted several different shared tasks related to language identification and especially the identification of close languages, language varieties, and dialects (Zampieri et al., 2014; Zampieri et al., 2015; Malmasi et al., 2016; Zampieri et al., 2017). The fifth VarDial workshop included for the first time a shared task for Indo-Aryan language identification (ILI) (Zampieri et al., 2018). The goal of the shared task was to identify the language used in unlabeled texts written in Hindi and four related languages using the Devanagari script: Bhojpuri, Awadhi, Magahi, and Braj.

We have participated in the shared tasks of three previous VarDial workshops using systems based on different variations of the HeLI method (Jauhiainen et al., 2015b; Jauhiainen et al., 2016; Jauhiainen et al., 2017a). The HeLI method has turned out to be robust and competitive with other state-of-the-art language identification methods, gaining shared first place in the VarDial 2016 Discriminating between Similar Languages (DSL) shared task. The HeLI method is not especially tailored to be a dialect identification method, but it is a general purpose language identification method capable of distinguishing between hundreds of languages, some of which might be very close to each other (Jauhiainen et al., 2017b). In the Kone foundation funded Finno-Ugric Languages and the Internet project, a language identifier implementing the HeLI method has been used together with the Heritrix web-crawler to collect text in Uralic languages from the internet (Jauhiainen et al., 2015a). The language identifier using the HeLI method is available for download in GitHub[1]. In the current workshop, we wanted to try out some new variations and possible improvements to the original method. For the ILI task, we used the basic HeLI method, HeLI with adaptive language models, as well as an iterative version of the language model adaptation method.

2 Related work

The first automatic language identifier for digital text was described by Mustonen (1965). During more than 50 years, hundreds of conference and journal articles describing language identification experiments and methods have been published. For a recent survey on language identification and the methods used in the literature, see Jauhiainen et al. (2018). The HeLI method was first presented by Jauhiainen (2010)

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

and later more formally by Jauhiainen et al. (2016), but we also provide a full description of the exact variation of the method used to submit the best results on the ILI shared task.

2.1 Language identification for Devanagari script

The language identification between languages using the Devanagari script has been considered earlier. Kruengkrai et al. (2006) presented language identification results between ten Indian languages, including four languages written in Devanagari: Sanskrit, Marathi, Magahi, and Hindi. For the ten Indian languages they obtained over 90% accuracy with mystery texts 70 bytes in length. As language identification method, they used support vector machines (SVM) with string kernels. Murthy and Kumar (2006) compared the use of language models based on bytes and aksharas. Aksharas are the syllables or orthographic units of the Brahmi scripts (Vaid and Gupta, 2002). After evaluating the language identification between different pairs of languages, they concluded that the akshara-based models perform better than byte-based. They used multiple linear regression as the classification method.

Sreejith et al. (2013) tested language identification with Markovian character and word *n*-grams from one to three with Hindi and Sanskrit. A character bigram-based language identifier fared the best and managed to gain the accuracy of 99.75% for sentence-sized mystery texts. Indhuja et al. (2014) continued the work of Sreejith et al. (2013) investigating the language identification between Hindi, Sanskrit, Marathi, Nepali, and Bhojpuri. They also evaluated the use of Markovian character and word *n*-grams from one to three. For this set of languages word unigrams performed the best, obtaining 88% accuracy with the sentence-sized mystery texts.

Bergsma et al. (2012) collected tweets in three languages written with the Devanagari script: Hindi, Marathi, and Nepali. They managed to identify the language of the tweets with 96.2% accuracy using a logistic regression (LR) classifier (Hosmer et al., 2013) with up to 4-grams of characters. Using an additional training corpus, they reached 97.9% accuracy with the A-variant of prediction by partial matching (PPM). Later, Pla and Hurtado (2017) experimented with the corpus of Bergsma et al. (2012). Their approach using words weighted with TF-IDF (product of term frequency and inverse document frequency) and SVMs reached 97.7% accuracy on the tweets when using only the provided tweet training corpora. Hasimu and Silamu (2018) included the same three languages in their test setting. They used a two-stage language identification system, where the languages were first identified as a group using Unicode code ranges. In the second stage, the languages written with the Devanagari script were individually identified using SVMs with character bigrams. Their tests resulted in an F1-score of 0.993 within the group of languages using Devanagari with 700 best distinguishing bigrams. Indhuja et al. (2014) provided test results for several different combinations of the five languages and for the set of languages used by Hasimu and Silamu (2018) they reached 96% accuracy with word unigrams.

Rani et al. (2018) described a language identification system, which they used for discriminating between Hindi and Magahi. Their language identifier using lexicons and three character suffixes obtained an accuracy of 86.34%. Kumar et al. (2018) provided an overview of experiments on an earlier version of the dataset used in this shared task. They managed to obtain the accuracy of 96.48% and a macro F1-score of 0.96 on the dataset they used. For sentence level identification these results are quite good, and as such they indicate that the languages, at least in their written form as evidenced by the corpus, are not as closely related as for example the Balkan languages Croatian, Serbian, and Bosnian.

2.2 Unsupervised language model adaptation

In unsupervised language model adaptation, the language models are modified while identifying the language of previously unseen and unlabeled text. The goal is to adapt the models to better suit the language or languages used in the texts to be identified in order to reach higher identification accuracy.

The use of on-line language model adaptation for language identification of digital text has been very limited. Blodgett et al. (2017) experimented with a method where they first identified the language of tweets using standard *langid-py* (Lui and Baldwin, 2012), and then collected the tweets with high posterior probability for English. From the collected tweets they generated a second language model for English to be used by the language identifier. Language identifiers can have several language models for one language, all of them providing the same classification if chosen. Their experiments produced a small increase in recall.

Chen and Liu (2005) use language model adaptation with language identification of speech similarly as we are using it in the language identification of text. The language identification system used by Chen and Liu (2005) first runs the speech through Hidden Markov Model-based phone recognizers (one for each language), which tokenize the speech into sequences of phones. The probabilities of these phone sequences for corresponding languages are calculated using language models and the most probable language is selected. An adaptation routine is then used so that each of the phonetic transcriptions of the individual speech utterances is used to calculate probabilities for words t, given a word n-gram history of h as in Equation 1.

$$P_a(t|h) = \lambda P_o(t|h) + (1 - \lambda) P_n(t|h), \tag{1}$$

where P_o is the original probability calculated from the training material, P_n the probability calculated from the data being identified, and P_a the new adapted probability. λ is the weight given to original probabilities. Using this adaptation method resulted in decreasing the language identification error rate in a three-way identification between Chinese, English, and Russian by 2.88% and 3.84% on an out-of-domain (different channels) data, and by 0.44% on in-domain (same channel) data.

Zhong et al. (2007) describe a confidence measure which they use with language identification of speech and define as follows:

$$C(g_i, M) = \frac{1}{n}[\log(P(M|g_i)) - \log(P(M|g_j))], \tag{2}$$

where M is the sequence to be identified, n the number of frames in the utterance, g_i the best identified language, and g_j the second best identified language. In the evaluations of Zhong et al. (2007), this confidence measure performed clearly better than two other ones they experimented with. They also evaluated an ensemble of all three confidence measures which managed to slightly improve the results. They then use the same language adaptation method as Chen and Liu (2005), using the confidence measures to set the λ for each utterance.

Bacchiani and Roark (2003) used unsupervised language model adaptation in a speech recognition task. They experimented with iterative adaptation on their language models. One additional adaptation iteration raised the accuracy gain of the language model adaptation from 3.4% to 3.9%, but subsequent iterations made the accuracy worse.

3 Task setup and data

For the preparation of the shared task, the participants were provided with training and development datasets. An early version of the dataset used, as well as its creation, was described by Kumar et al. (2018). The dataset used for the shared task included text in five languages, Bhojpuri, Hindi, Awadhi, Magahi, and Braj as shown in Table 1. The size of the training material was considerably smaller for the Awadhi language at slightly over 9,000 lines compared with the others which were around 15,000 long. The difference in size of the training material might produce problems for some methods that have been used for language identification. The HeLI method has turned out to be very robust in this respect, so we did not need to take this into any special consideration.

Language name	Code used	Training data (lines)	Development data (lines)
Bhojpuri	BHO	14,897	2,003
Hindi	HIN	15,642	2,253
Awadhi	AWA	9,307	1,480
Magahi	MAG	15,306	2,285
Braj	BRA	15,111	2,308

Table 1: List of languages with the sizes of their training and development sets.

The task was an open one, allowing the use of any additional data or means. However, we did not try to use any external means and our results would have been exactly the same in a closed version of the task. Participants were allowed to submit three runs for the ILI task and the best out of those submissions would be ranked. We submitted one with the original HeLI method, one using language model adaptation, and one using an iterative version of language model adaptation.

4 The HeLI method, run 1

To make this article more self-contained, we present the full description of the method as used in the submitted runs. This description differs from the original by Jauhiainen et al. (2016) mostly in that we leave out the cut-off value c for the size of the language models. In this year's shared tasks we found, and have already noticed it earlier, that if the corpora used as the training corpus is of good quality it is generally advisable to use all the available material. Furthermore, the penalty value compensates for some of the possible impurities in the language models. The final submissions were done with a system not using words at all, so we leave them out of the description as well.

Description of the HeLI method The goal is to correctly guess the language $g \in G$ for each of the lines in the test set. In the HeLI method, each language g is represented by several different language models only one of which is used for every word t in the line M. The language models in this version are based on character *n*-grams from one to n_{max}. When none of the *n*-grams of the size n_{max} generated from the word under scrutiny are found in any of the language models, we back off to using the *n*-grams of the size $n_{max} - 1$. If needed, we continue backing off until character unigrams.

The training data is tokenized into words using non-alphabetic and non-ideographic characters as delimiters. The data is lowercased, even though the actual Devanagari script does not use capital letters, but there is some material in the data in other scripts as well. 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. Then we transform the relative frequencies into scores using 10-based logarithms. Among the language models generated from the ILI training corpus, the largest model, Hindi 5-grams, included 80,539 different *n*-grams.

The corpus containing only the *n*-grams of the length n 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 any language $g \in G$. 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 3.

$$v_{C^n_g}(u) = \begin{cases} -\log_{10}\left(\frac{c(C^n_g,u)}{l_{C^n_g}}\right) & \text{, if } c(C^n_g,u) > 0 \\ p & \text{, if } c(C^n_g,u) = 0, \end{cases} \tag{3}$$

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

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 4.

$$d_g(t,n) = \sum_{i=1}^{l_t-n} \begin{cases} 1 & \text{, if } u^n_i \in dom(O(C^n)) \\ 0 & \text{, otherwise.} \end{cases} \tag{4}$$

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 5.

$$v_g(t,n) = \begin{cases} \frac{1}{d_g(t,n)} \sum_{i=1}^{l_t-n} v_{C_g^n}(u_i^n) & \text{, if } d_g(t,n) > 0 \\ v_g(t,n-1) & \text{, otherwise,} \end{cases} \tag{5}$$

where $d_g(t,n)$ is the number of *n*-grams u_i^n found in the domain $dom(O(C_g^n))$. 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 6.

$$v_g(t,0) = p \tag{6}$$

The mystery text is tokenized into words using the non-alphabetic and non-ideographic characters as delimiters. The words are lowercased when lowercased models are being used. After this, a score $v_g(t)$ is calculated for each word t in the mystery text for each language g, as shown in Equation 7.

$$v_g(t) = v_g(t, min(n_{max}, l_t + 2)) \tag{7}$$

If 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 line 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 8.

$$R_g(M) = \frac{\sum_{i=1}^{l_{T(M)}} v_g(t_i)}{l_{T(M)}}, \tag{8}$$

where $T(M)$ is the sequence of words and $l_{T(M)}$ is the number of words in the line 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.

Results of the run1 on the development and the test sets The development set was used for finding the best values for the parameter p and to decide which language models to use. We experimented with several different combinations of language models and the resulting recall-values of these trials can be seen in Table 2. "Original n_{max}" refers to the maximum size used with the original *n*-grams and "Lowercased n_{max}" to the size used with the lowercased *n*-grams. The differences in recall between the combinations are not very high.

Original words	Original n_{max}	Lowercased words	Lowercased n_{max}	Penalty p	Recall
no	-	no	6	5.9	95.26%
no	-	no	5	6.4	95.11%
no	-	no	7	6.0	95.08%
no	8	no	8	5.7	95.01%
no	7	no	8	5.7	95.01%
no	6	no	8	5.7	95.01%
no	-	no	8	5.7	95.01%
no	-	no	4	6.7	95.00%
yes	8	yes	8	6.0	94.81%
yes	8	no	8	6.0	94.81%
no	8	yes	8	6.0	94.81%
no	-	yes	8	6.0	94.81%

Table 2: Baseline HeLI recall in development data with different combinations of parameters.

We decided to use lowercased character *n*-grams from one to six with the penalty value of 5.9 for the first run. We included the development set in the training material to generate the final language models. The recall for the test set was 89.28% and the macro F1-score, which is used for ranking in the ILI shared task, was 0.8873.

5 Unsupervised language model adaptation, run 2

The idea behind language model adaptation is to incorporate new language material into the language models while previously unseen and untagged text is processed. Most language identifiers that can indicate how well they perform could be used with language model adaptation. The system also benefits if adding new information to the language models is reasonably easy. Our method is recursive and it builds on the fact that we can process the same batch of previously unseen texts several times before providing the final labels. In our method, the information from the sentences in the unseen text is added to the language models one sentence at a time. The sentence to be processed next is always the one that the language identifier deems to be the one that is most probably correctly identified using the current language models. In order to determine which of the sentences is most identifiable, we could use the probabilities given to the sentence by the language models. However, this probability can be almost equally high for several languages if they are very close to each other. What we want to find is a sentence that gains high probability in one of the languages, but low probability in others. We achieve this by maximizing the difference between the probabilities of the first and the second identified languages.

Description of the unsupervised language model adaptation method All the lines M are first identified using the HeLI method. Then the best identified line, as ranked by the confidence score CM, is set as identified. In order to rank the identified lines to use for language model adaptation, we must be able to tell how confident the language identifier is in its decision. As confidence measure CM, we used the difference between the scores of the best $R_g(M)$ and the second best $R_h(M)$ identified language for each line. This is basically the same as the confidence measure proposed by Zhong et al. (2007). We did not test the other two methods presented by Zhong et al. (2007), or their Bayesian classifier-based ensemble. In our case, the confidence measure is calculated using Equation 9:

$$CM(C_g, M) = R_h(M) - R_g(M), \tag{9}$$

where M is the line containing the mystery text. The character n-grams up to the length of six are created from the line with the best confidence and they are added to the language models of the winning language. After this, the rest of the lines are re-identified with the adapted models and the line with the best confidence is again added to the models of the language it was identified to be written with. This process is repeated until all the lines have been added to the language models. Each time the lines are re-identified there is one less line to process. Nevertheless, the number of identifications is exponential relative to the number of lines to be identified when compared with only identifying them once.

Results of run2 on the development and the test sets In preparation for the second run, we used the same language models and penalty value as for the first run. The language identifier with language model adaptation achieved 96.22% recall on the development set. It was an increase of 0.96% on top of the recall of the basic HeLI method. For the submission run, we used both the development and the training sets to generate the initial language models. The submitted second run reached a recall of 95.66% on the test set, a formidable increase of 6.38% when compared with the first run. In other words, using the language model adaptation reduced the error rate by 59.5%. The fact that the percentage gain using language model adaptation was clearly more considerable on the test set than on the development set indicates that the test set is more out-of-domain from the combined development and trainings sets than the development set was from the training set. The macro F1-score obtained on our second run was 0.9553.

6 Iterative language model adaptation, run 3

While we were experimenting with language model adaptation, we noticed that if the initial language models are good enough, the adaptation process can be repeated. The additional accuracy gained was usually very small, but the repeated adaptation only very rarely affected the results in any negative manner. This is in contrast to the findings of Bacchiani and Roark (2003), who found that performing subsequent adaptations made the results worse.

Iterative language model adaptation basically means that the process for language model adaptation is restarted after one learning epoch. We noted the time it took to produce the results on the second run and decided to use four epochs for our third run on the basis of time left before the submissions were due. We used iterative adaptation with four epochs on the test set, gaining a small additional increase of 0.22% to recall, reaching 95.88% with the F1 score of 0.9576. The final results of all our runs and the best runs of the other teams are listed in Table 3.

Method (or team)	F1 (macro)
HeLI with iterative language model adaptation (run3)	**0.9576**
HeLI with language model adaptation (run2)	**0.9553**
taraka_rama	0.9022
XAC	0.8933
ILIdentification	0.8895
HeLI (run1)	**0.8873**
safina	0.8627
dkosmajac	0.8472
we_are_indian	0.8360
LaMa	0.8195
Random Baseline	0.2024

Table 3: Macro F1 scores obtained by different runs submitted by the SUKI-team (bolded) and the best runs of the other teams.

Figure 1 shows the confusion matrix for the Indo-Aryan languages in our third and final run. From the figure it seems that the largest misclassified group was 146 sentences in Bhojpuri, which were identified as Hindi. We randomly selected some Bhojpuri sentences to try with Google translator and it detected them all as Hindi and was also able to produce seemingly intelligible English translations for them. Unfortunately, our limited understanding of the languages in question prevents us from doing any deeper error analysis.

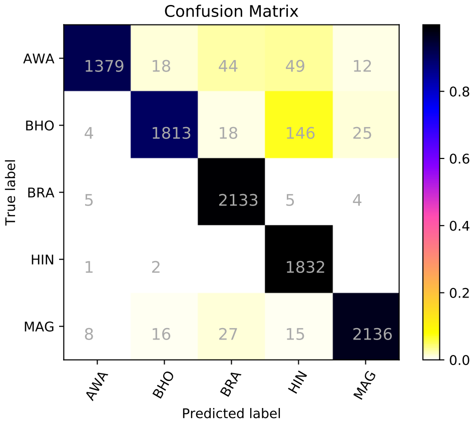

Figure 1: Confusion matrix for final submitted run.

7 Other experiments

We experimented with leaving out shorter lowercased *n*-grams with $n_{max} = 6$. Leaving out character unigrams and bigrams did not affect the recall, but leaving out trigrams dropped the recall to 94.53% indicating that the HeLI back-off function is also needed for these languages. With the German dialect identification task we ended up using only 4-grams of characters.

We also experimented with an unsupervised language set adaptation method. In unsupervised language set adaptation, the mystery text is first identified using all the available languages. The language with the worst score is left out and the text re-identified with the remaining languages. The process is continued until only one language is left. In a non-discriminative language identification method, the effect of leaving out languages with the worst scores does not affect the order of the top scoring languages.

However, if the back-off function of the HeLI method is used, it gives equal penalty values to those languages in which a word is not found. If the word was found in an otherwise poorly scoring language, which was subsequently left out, the following run might use the back-off function with the word in question and find a difference between the better candidates using character *n*-grams. We expected the effect to be small, and it turned out to be slightly negative reducing the recall from 95.26% to 95.22%.

We, furthermore, evaluated the same non-linear mappings, the gamma and the loglike functions, we used in the DSL shared task at VarDial 2017 (Jauhiainen et al., 2017a). The experiments with the gamma function ended up with the same recall of 95.26% as the original method. Several different trials with loglike functions fell short of the recall of the original method at 95.25%.

8 Conclusions

The language model adaptation scheme works very well on the ILI test set. With the German dialect identification task, we noticed that the language adaptation method works especially well when the test set is out-of-domain compared with the training set. The very good results in the ILI task might indicate that there is a clear domain difference between the training/development sets and the test set. The iterative use of the adaptation method with 4 epochs also turned to be beneficial, reducing the remaining errors by 5.1%.

Acknowledgments

This research was partly conducted with funding from the Kone Foundation Language Programme (Kone Foundation, 2012).

References

Michiel Bacchiani and Brian Roark. 2003. Unsupervised language model adaptation. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP 2003)*, pages 224–227.

Shane Bergsma, Paul McNamee, Mossaab Bagdouri, Clayton Fink, and Theresa Wilson. 2012. Language Identification for Creating Language-specific Twitter Collections. In *Proceedings of the Second Workshop on Language in Social Media (LSM2012)*, pages 65–74, Montréal, Canada.

Su Lin Blodgett, Johnny Tian-Zheng Wei, and Brendan O'Connor. 2017. A Dataset and Classifier for Recognizing Social Media English. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 56–61, Copenhagen, Denmark.

Yingna Chen and Jia Liu. 2005. Language Model Adaptation and Confidence Measure for Robust Language Identification. In *Proceedings of International Symposium on Communications and Information Technologies 2005 (ISCIT 2005)*, volume 1, pages 270–273, Beijing, China.

Maimaitiyiming Hasimu and Wushour Silamu. 2018. On Hierarchical Text Language-Identification Algorithms. *Algorithms*, 11(39).

David W. Hosmer, Stanley Lemeshow, and Rodney X. Sturdivant. 2013. *Applied logistic regression*. Wiley Series in Probability and Statistics. Wiley, Hoboken, N.J., USA, 3rd ed edition.

K. Indhuja, M. Indu, C. Sreejith, and P. C. Reghu Raj. 2014. Text Based Language Identification System for Indian Languages Following Devanagiri Script. *International Journal of Engineering Reseach and Technology*, 3(4):327–331.

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. 2016. HeLI, a Word-Based Backoff Method for Language Identification. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)*, pages 153–162, Osaka, Japan.

Tommi Jauhiainen, Krister Lindén, and Heidi Jauhiainen. 2017a. Evaluating HeLI with Non-Linear Mappings. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pages 102–108, Valencia, Spain.

Tommi Jauhiainen, Krister Lindén, and Heidi Jauhiainen. 2017b. Evaluation of Language Identification Methods Using 285 Languages. In *Proceedings of the 21st Nordic Conference on Computational Linguistics (NoDaLiDa 2017)*, pages 183–191, Gothenburg, Sweden. Linköping University Electronic Press.

Tommi Jauhiainen, Marco Lui, Marcos Zampieri, Timothy Baldwin, and Krister Lindén. 2018. Automatic Language Identification in Texts: A Survey. *arXiv preprint arXiv:1804.08186*.

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

Kone Foundation. 2012. The language programme 2012-2016. http://www.koneensaatio.fi/en.

Canasai Kruengkrai, Virach Sornlertlamvanich, and Hitoshi Isahara. 2006. Language, Script, and Encoding Identification with String Kernel Classifiers. In *Proceedings of the 1st International Conference on Knowledge, Information and Creativity Support Systems (KICSS 2006)*, Ayutthaya, Thailand.

Ritesh Kumar, Bornini Lahiri, Deepak Alok, Atul Kr. Ojha, Mayank Jain, Abdul Basit, and Yogesh Dawar. 2018. Automatic Identification of Closely-related Indian Languages: Resources and Experiments. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC)*, Miyazaki, Japan.

Marco Lui and Timothy Baldwin. 2012. langid.py: An Off-the-shelf Language Identification Tool. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012) Demo Session*, pages 25–30, Jeju, Republic of Korea.

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.

Kavi Narayana Murthy and G. Bharadwaja Kumar. 2006. Language Identification from Small Text Samples. *Journal of Quantitative Linguistics*, 13(1):57–80.

Seppo Mustonen. 1965. Multiple Discriminant Analysis in Linguistic Problems. *Statistical Methods in Linguistics*, 4:37–44.

Ferran Pla and Lluís-F. Hurtado. 2017. Language Identification of Multilingual Posts from Twitter: A Case Study. *Knowledge and Information Systems*, 51(3):965–989.

Priya Rani, Atul Kr. Ojha, and Girish Nath Jha. 2018. Automatic language identification system for hindi and magahi. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan.

C. Sreejith, M. Indu, and P. C. Reghu Raj. 2013. N-gram based Algorithm for Distinguishing Between Hindi and Sanskrit Texts. In *Proceedings of the Fourth IEEE International Conference on Computing, Communication and Networking Technologies*, Tiruchengode, India.

Jyotsna Vaid and Ashum Gupta. 2002. Exploring Word Recognition in a Semi-Alphabetic Script: The Case of Devanagari. *Brain and Language*, 81:679–690.

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, 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)*, pages 1–9, Hissar, Bulgaria.

Marcos Zampieri, Shervin Malmasi, Nikola Ljubešić, Preslav Nakov, Ahmed Ali, Jörg Tiedemann, Yves Scherrer, and Noëmi Aepli. 2017. Findings of the VarDial Evaluation Campaign 2017. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, Valencia, Spain.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Ahmed Ali, Suwon Shon, James Glass, Yves Scherrer, Tanja Samardžić, Nikola Ljubešić, Jörg Tiedemann, Chris van der Lee, Stefan Grondelaers, Nelleke Oostdijk, Antal van den Bosch, Ritesh Kumar, Bornini Lahiri, and Mayank Jain. 2018. Language Identification and Morphosyntactic Tagging: The Second VarDial Evaluation Campaign. In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, Santa Fe, USA.

Shan Zhong, Yingna Chen, Chunyi Zhu, and Jia Liu. 2007. Confidence measure based incremental adaptation for online language identification. In *Proceedings of International Conference on Human-Computer Interaction (HCI 2007)*, pages 535–543, Beijing, China.