BMSCE_ISE@INLI-FIRE-2017: A simple n-gram based approach for Native Language Identification

Sowmya Lakshmi B S.[1], Dr. Shambhavi B R.[2]

Department of ISE, BMS College of Engineering, Bangalore, India
sowmyalakshmibs.ise@bmsce.ac.in[1], shambhavibr.ise@bmsce.ac.in[2]

ABSTRACT

Native Language Identification (NLI) aims to identify native language L1 of an author by analysing the text written by him/her in other language L2. NLI is often implemented as a supervised classification problem. In this paper, we report a NLI system implemented using character tri-grams, word uni-grams and bi-grams methods using linear classifier, Support Vector Machines (SVM). The work demonstrated is a participant of Indian Native Language Identification@FIRE 2017, achieving 0.27 overall accuracy for the corpus with 6 native languages. Furthermore, with subsequent evaluations, the best accuracy score obtained was 0.73 with 10 fold cross-validation on training data. We were able to achive above accuracy by incorporating uni-grams and bi-grams of words.

KEYWORDS

Language Identification; Supervised Classification; Feature Selection.

1 INTRODUCTION

Recently, author profiling is gaining more importance to improve performance of certain applications like forensics, security and marketing. Author profiling aims to detect author's details like age, educational level and native language. Native Language Identification (NLI) is a sub-class of author profiling where, native language L1 of a writer is automatically detected by analysing the text written in the second language L2. NLI is often implemented as a multiclass supervised classification task.

The applications of NLI are categorised into two categories: security related applications and Second Language Acquisition (SLA)- related applications. Security related applications are identifying phishing sites or spam e-mails that usually consist of strange sentences that might be written by non-native persons. SLA applications are to analyse the effect of L1 on later learned languages.

As proved by preceding work in this area there exist quite a few linguistic hints that helps in predicting native language. With the impact of their native language, authors tend to make common mistakes in spelling, punctuation and grammar while using other languages.

In this work, we examine the possibility of building native language classifiers by ignoring grammatical errors and semantic analysis of the text written in L2. A naive set of features using n-grams of words and characters are explored to develop NLI system.

2 PREVIOUS WORK

The work presented in this study was a participant of Indian Native Language Identification@FIRE 2017 shared task. Several researchers have investigated NLI and similar problems. An overview of few common methods used for NLI prior to this shared task is provided.

Most of the researchers have featured NLI as a supervised classification task, where classifiers were trained on data from different L1. Most commonly included features for NLI are character n-grams, POS n-grams, content words, function words and spelling mistakes. An SVM model [1-3] was trained on these features and obtained an accuracy of 60%-80%.

In the recent past, word embedding and document embedding has gained much attention along with other features. Continuous Bag of Words (CBOW) and Skip Grams were used to obtain vectors of word embedding. Vector representations for documents were generated with distributed bag-of-words architectures using Doc2Vec tool. In [4], authors developed a native language classifier using document and word embedding with an accuracy of 82% for essays and 42% on speech data.

LIBSVM2, variant of SVM was verified to be efficient for text classification. In [5], authors developed a NLI algorithm for Arabic language with LIBSVM2. They combined production rules, function words and POS bi-grams to perform machine learning process and obtained an accuracy of 45%.

First NLI shared task was organized with BEA workshop in 2013. System participated in closed training task was presented in [6]. The model was trained on 11 L1 languages of TOEFL11 corpus and cross-validation testing was performed for unseen essays resulted in accuracy of about 84.55%. Authors adopted features like n-grams of words, characters and POS and spelling errors with TF-IDF weighing to train SVM model.

In [7], author reported the work participated in essay track of the Second NLI Shared Task 2017 held at BEA-12 workshop. A novel 2-stacked sentence-document architecture was introduced by considering lexical and grammatical features of text. A stack of two SVM classifiers were used, where first and second classifier were sentence and document classifiers respectively. First classifier aimed at predicting the native language of each sentence of a document whereas, these predictions were adopted as features by document classifier. Finally, system was used to predict native language of unseen documents which resulted F1-score of 0.88.

3 TASK DESCRIPTION AND DATA

NLI has drawn the attention of many researchers in recent years. With the influx of new researchers, the most substantive study in this field has led to INLI@FIRE 2017 shared task [8]. Task focuses on identifying native language of a writer based on his writing in other language. In this case, the second language was English. The task was, native language prediction of a writer from the given Text/XML file which contains Facebook comments in English language. Six Indian languages were proposed to consider for this task. They were Tamil, Hindi, Kannada, Malayalam, Bengali and Telugu.

Dataset

The training dataset for the task was xml files, which contains a set of Facebook comments in English by different native language speakers. Xml files were annotated as BE, HI, KA, MA, TE, and TA for Bengali, Hindi, Kannada, Malayalam, Telugu and Tamil language respectively. Table 1 shows the training data statistics that was used for the task.

Table 1: Training data

Native Language	Files
Bengali	202
Hindi	211
Kannada	203
Telugu	207
Malayalam	200
Tamil	210

4 FEATURES

NLI has been formulated as a multiclass classification task. We used language-independent features such as character tri-grams and word n-grams for NLI as described in [1]. From the previous works we observed that character tri-grams were useful for NLI, and they suggested that this might be due to the impact of author's native language. To reflect this, we calculate character n-grams and word n-grams as features. For characters, we consider tri-grams. The features are generated over the entire training data, i.e., every tri-gram in the training dataset is used as a feature. Similarly, uni-grams and bi-grams of words were used as separate features.

5 APPROACH

Training dataset provided were xml files which contained Facebook comments in English written by different native language speakers and files were annotated w.r.t native language of the speaker. As a part of preprocessing, these xml files were scraped to extract Facebook comments and comments related to similar native language were saved in a text file. We extracted features from the text files generated and developed two methods for NLI using python as explained below.

Character tri-grams method

The tri-gram model reads text files and extracts all tri-grams (sequence of three bytes) and their corresponding counts from the text. Frequencies of tri-grams are pursued for every training language separately. For every language, frequencies are relativized by dividing individual tri-gram counts through the number of all tri-grams in the training corpus and are sorted based on the relative frequency (the probability of the tri-gram in the given corpus of a language) to create language model of that language. A language model for each language in the corpus provided was created.

Relative frequencies of the tri-grams for test dataset is calculated and compared with the tri-grams in language models. Intuitively, we would say that the tri-gram frequencies of tri-grams extracted from two different texts of the same native language speaker should be very similar. The absolute difference was calculated by subtracting the relative frequency of individual tri-gram in the test dataset from the relative frequency of corresponding tri-gram in each language model. The absolute differences were summed up. For instance, if we compare test data with 5 language models, we would have 5 different values for the sum of absolute differences. The minimum value represents the best match for test data. Algorithm 1 describes the algorithm for character tri-gram approach.

Algorithm 1: Character tri-grams method

```
Input: Train Dataset for each language, Test Dataset
Output: Native Language Identification of Test Dataset
begin
for each language in language set
    for each document in Train Dataset of language
        Derive all possible tri-grams
    end for
    Language model<- Frequency of each tri-gram in Train
    Dataset of language
end for
for every document in Test Dataset
    Derive all possible tri-grams
    Calculate relative frequency of each tri-gram in the document
end for
for each language in language model
    for every tri-gram in the Test Dataset document
        Calculate absolute difference
        Absolute difference <- (Relative frequency of tri-gram
        in Test Data) – (Relative frequency of corresponding tri-
        gram in language model)
    end for
    Sum up the calculated absolute differences of each language
    model
end for
Best match <- Among all the computed absolute differences select
the one with   minimum value.
end
```

Word n-grams method

Frequencies of word uni-grams and bi-grams were collected for each language irrespective of their meanings and order of words in the document. We instantiated countvectorizer module in python to achieve word uni-grams and bi-grams. A Document Term Matrix *X [i, j]* was formed, where *i* is the document id, *j* represents dictionary index of each word and *Wij* is the frequency of occurrence of each word *w* in document *i*. Each uni-gram and bi-gram of test data was compared with their frequencies of occurrences in the documents of all languages.

In this experiment, we used Document Term Matrix with n-grams and applied linear SVM from scikit-learn as a classification algorithm for NLI.

6 RESULT ANALYSIS

We submitted the output of the system for test data provided to INLI@FIRE 2017 shared task workshop. A single run of each method for six different languages was submitted and the results of native language classification for all the languages are recapitulated in Table 2 and Table 3. Character tri-gram model achieved 22% accuracy and word n-grams model achieved an overall accuracy of 27%.

The combined features of uni-grams and bi-grams on the training data was used to perform 10 fold cross-validation. With these features an improved accuracy of 73% was achieved.

Table 2: Character tri-grams

Class	Prec.	Rec.	F1
BE	0.40	0.292	0.338
HI	0.50	0.080	0.160
KA	0.117	0.270	0.163
MA	0.1730	0.641	0.272
TA	0.533	0.080	0.139
TE	0.267	0.383	0.315
Overall Accuracy			0.22

Table 3: Word n-grams

Class	Prec.	Rec.	F1
BE	0.389	0.551	0.456
HI	0.545	0.072	0.127
KA	0.190	0.446	0.266
MA	0.223	0.315	0.261
TA	0.218	0.260	0.237
TE	0.154	0.123	0.137
Overall Accuracy			0.27

7 CONCLUSION

In this paper, a supervised system for Indian Native Language Identification has been presented. We describe character tri-grams, word uni-grams and bi-grams features, which are the subset of frequently used features for NLI task. Results of the supervised classification using these features on a test data set consisting of 6 languages were reported as part of INLI@FIRE 2017 shared task. Our future work lies in improving the performance of NLI system by considering features, which can classify native languages in a better way.

REFERENCES

[1] Nicolai, Garrett, Md Asadul Islam, and Russ Greiner. "Native Language Identification using probabilistic graphical models." Electrical Information and Communication Technology (EICT), 2013 International Conference on. IEEE, 2014.

[2] Abu-Jbara, A., Jha, R., Morley, E., & Radev, D. R. (2013, June). Experimental Results on the Native Language Identification Shared Task. In BEA@ NAACL-HLT (pp. 82-88).

[3] Mizumoto, T., Hayashibe, Y., Sakaguchi, K., Komachi, M., & Matsumoto, Y. (2013, June). NAIST at the NLI 2013 Shared Task. In BEA@ NAACL-HLT (pp. 134-139).

[4] Vajjala, Sowmya, and Sagnik Banerjee. "A study of N-gram and Embedding Representations for Native Language Identification." In Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications, pp. 240-248. 2017.

[5] Mechti, Seifeddine, Lamia Hadrich Belguith, Ayoub Abbassi, Rim Faiz, and Carthage IHEC. "An empirical method using features combination for Arabic native language identification."

[6] Gebre, B. G., Zampieri, M., Wittenburg, P., & Heskes, T. (2013).” Improving native language identification with tf-idf weighting”. In the 8th NAACL Workshop on Innovative Use of NLP for Building Educational Applications (BEA8) (pp. 216-223).

[7] Cimino, A., & Dell'Orletta, F. (2017). “Stacked Sentence-Document Classifier Approach for Improving Native Language Identification”. In Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications (pp. 430-437).

[8] Anand Kumar M, Barathi Ganesh HB, Shivkaran S, Soman K P, Paolo Rosso. Overview of the INLI PAN at FIRE-2017 Track on Indian Native Language Identification. In: Notebook Papers of FIRE 2017, FIRE-2017, Bangalore, India, December 8-10, CEUR Workshop Proceedings.