An Automatic Intelligent Language Classifier

Brijesh Verma[1], Hong Lee[1], and John Zakos[2]
[1] School of Computing Sciences, CQUniversity
Rockhampton, Queensland, Australia
{B.Verma, H.Lee1}@cqu.edu.au
[2]MyCyberTwin, Gold Coast, Queensland, Australia

Abstract. The paper presents a novel sentence-based language classifier that accepts a sentence as input and produces a confidence value for each target language. The proposed classifier incorporates Unicode based features and a neural network. The three features Unicode, exclusive Unicode and word matching score are extracted and fed to a neural network for obtaining a final confidence value. The word matching score is calculated by matching words in an input sentence against a common word list for each target language. In a common word list, the most frequently used words for each language are statistically collected and a database is created. The preliminary experiments were performed using test samples from web documents for languages such as English, German, Polish, French, Spanish, Chinese, Japanese and Korean. The classification accuracy of 98.88% has been achieved on a small database.

Keywords: Classifiers, Language Classification, Neural Networks

1 Introduction

Automatic language classification systems are needed in many real world applications such as web based communication, multilingual document classification, medical cross-language text retrieval systems, helpdesk call routing and spoken language classification just to mention a few.

Automatic language classification is the problem of identifying in which language a given sample text has been written. Living in a global community, we are surrounded by multi-lingual environments such as web documents, speeches, etc. Especially, global advances in the Internet communities have imposed a great deal of importance for language classification problem due to the huge amount of web documents published in multi-languages. Successful research outcomes can affect many industrial sectors. A multi language translation technique [1, 2] is one of the examples, where the input language needs to be classified prior to the translation to a target language. Also, the language identification plays a key role in the internet search engines by identifying the language of the search keys [3]. Researchers have found [4] that text-to-speech applications heavily depend on the language identification performances in multi-lingual environments.

Language classification tasks based on the written mono text (single language document) has been regarded as a relatively simple problem for small number of languages and when a large amount of sample texts in the identification stage are available. However, the task of language classification is very difficult and challenging when we have multi-language documents and large number of languages to classify. The complexity of the problem solving significantly increases [5] with the size of input text.

The main goal of the research presented in this paper is to investigate a novel classifier that accepts a sentence including multilingual/small sentence as input and provide a confidence value for each language. The paper is divided into five sections. Section 2 presents existing techniques for language classifiers, limitations and difficulties. Section 3 presents the proposed research methodology. The experimental results and analysis are presented in Section 4. Finally, a conclusion is presented in Section 5.

2 Background

The standard framework involved in language identification is modeling and classification. In language modeling stage, the most discriminative features of each target language is extracted and stored in its language model. During classification, similar feature extraction process is performed on input texts. Based on the models of each language and input text, the distance of similarity or dissimilarity is measured and the input text is identified according to the score. In [6], a language identification system has been presented which can achieve accuracy of 93% with as little as a three-word input.

There has been some research conducted in the area of automatic classification of languages and some papers have been published in recent years. In [3], an approach is proposed which can classify input texts' language by finding the maximum frequency of input words in each dictionary of Spanish, French, English, Portuguese, German and Italian. To identify the input language, heuristics are employed into the decision making process. The methodology is effective to classify input texts' language as accurate as 88% on randomly selected web pages and 99% on randomly selected well-formatted texts. In [5], a decision tree scheme for common letters of language in documents is used to identify Arabic from Persian. The decision tree is defined as a series of questions about the context of the current letter. If a common but discriminant letter from the other language is found, the classification is made on the incident. The experiment result shows that average of around 98.8% accuracy was achieved to identify 240 web documents (120 for Arabic and 120 for Persian). In [6], each language is modeled from a corpus of training documents on features extracted based on common words and N-gram methods. The features extracted by the common words are the probability distribution of the frequency of the most common words in the training documents in a language. Likewise, features of character N-gram is measured to reflect the frequency score and the rank of N-gram instances are stored. During classification, rather than modeling the whole input text, features of random sub-sections of the input texts are extracted to minimize the computational time. The random sampling is performed until the standard error of the random samples is larger than a threshold. The Monte Carlo method with N-gram and common words was tested on Danish, Dutch, English, French, German, Italian, Norwegian, Portuguese, Spanish, and Swedish from ECI database. However, it doesn’t report the numerical data on the performance of the classification apart from the comparative graph between difference methods. In [8], two identification methods, enhanced N-gram probabilities and decision tree are proposed to compare the performance of classification accuracy. The authors enhanced N-gram feature extraction technique by decomposing each word into three parts, head, body and tail. The decision trees are to identify the most likely language for each letter in the input word. The experiment results on local in-house guest names in four languages reported that 71.8% and 66.1% average identification accuracies were achieved by N-gram and decision tree methods accordingly. Vector-space based identification approach was proposed in [9] for 13 Latin character based languages. Features included in vectors were N-gram frequencies and word sizes with inverse document frequency weight incorporated. Between models and an input, cosine values are calculated and used to classify the input text. Experiment results report various performance accuracies depending on the input text size, which produced 100% accuracy on web documents with 1000 bytes. In [10], the authors incorporated feature extraction technique of the common words in a language, known as stop words like ‘the’, ‘of’ and ‘to’, to identify the language from scanned document images written in multi-lingual environments. In their research, the stop words, their frequency and word shape code are used as key feature vectors to classify the language which input documents were written in. The approach was as effective as 96.75% of accuracy rate at best on locally prepared database. In [11], Artemenko et al. evaluated performances of four different identification methodologies in two separate experiments of mono-lingual and multi-lingual web documents on 8 languages. Identification methods used in the experiments were Vector space cosine similarity, ‘out of place’ similarity between rankings and Bayesian classifier on N-gram feature spaces. A word frequency based classification was added to the comparison. The research inferred that N-gram based approach outperforms the word frequency based methods for short texts. The researchers were able to achieve 100% and 97% accuracies on mono and multi lingual documents accordingly. In [12], an approach was proposed which can count common words and character sequences of N-gram methods for a language. Then, the frequency was used as the key information to distinguish the input documents against models. The performance was measured on Europarl corpus test sets, and was satisfactory, 97.9% on German language was achieved. In [13], an algorithm is presented which extends the common N-gram corpus analysis complemented with heuristics. Classification was to measure the similarity between input text and model languages. The literature reports the performance on 12 languages of 6000 web documents was 100% accurate at most. Rendering character sequence into HMM language model was manipulated as a key ingredient for language classification task in [14]. The identification accuracy of 95% was achieved in their proposal. In [15], term frequency and its weight by entropy method over documents were used as feature for neural networks to categorize the web documents. In [16], an approach was proposed which uses tri-gram and frequency language modelling technique to identify the origin of names written-in-Latin, Japanese, Chinese and English. An accuracy of 92% was achieved to distinguish Japanese names from the others.

3 Proposed Research Methodology

The proposed research methodology is described in details in this section. Foremost, an overview of the proposed technique is presented, followed by analysis of language specific Unicode. The proposed feature extraction and classification algorithms are described at the end of this section.

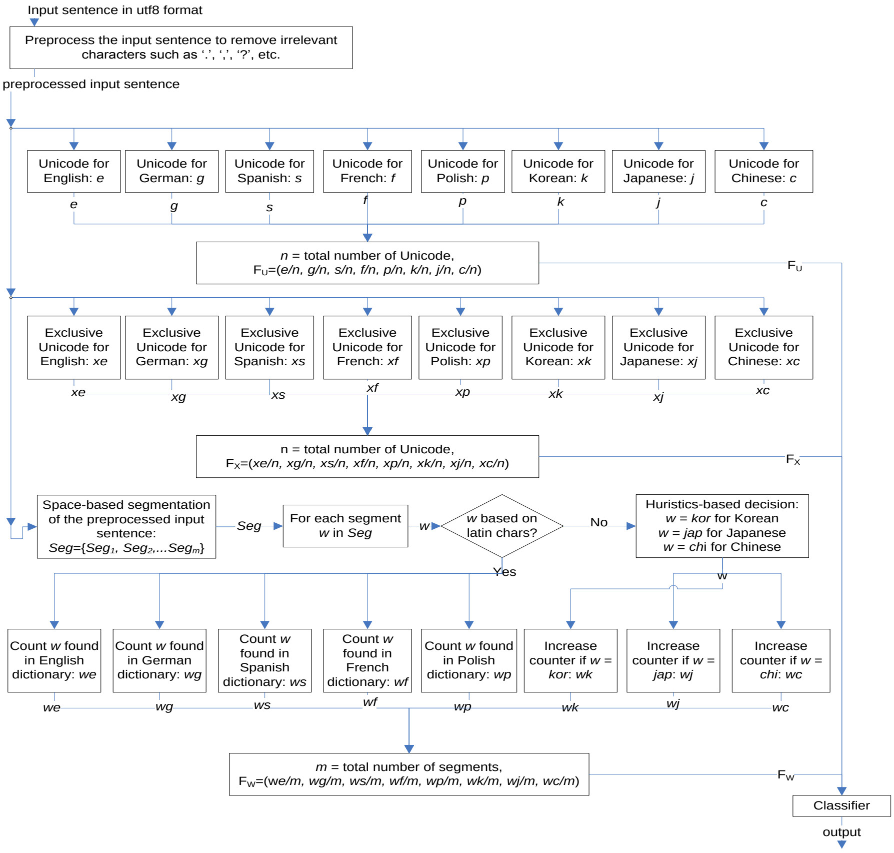
Input sentence in utf8 format
Preprocess the input sentence to remove irrelevant characters such as '.', ',', '?', etc.
preprocessed input sentence
Unicode for English: e
Unicode for German: g
Unicode for Spanish: s
Unicode for French: f
Unicode for Polish: p
Unicode for Korean: k
Unicode for Japanese: j
Unicode for Chinese: c
e
g
s
f
p
k
j
c
n = total number of Unicode, F_U=(e/n, g/n, s/n, f/n, p/n, k/n, j/n, c/n)
F_U
Exclusive Unicode for English: xe
Exclusive Unicode for German: xg
Exclusive Unicode for Spanish: xs
Exclusive Unicode for French: xf
Exclusive Unicode for Polish: xp
Exclusive Unicode for Korean: xk
Exclusive Unicode for Japanese: xj
Exclusive Unicode for Chinese: xc
xe
xg
xs
xf
xp
xk
xj
xc
n = total number of Unicode, F_X=(xe/n, xg/n, xs/n, xf/n, xp/n, xk/n, xj/n, xc/n)
F_X
Space-based segmentation of the preprocessed input sentence: Seg={Seg_1, Seg_2,...Seg_m}
Seg
For each segment w in Seg
w
w based on latin chars?
No
Huristics-based decision: w = kor for Korean w = jap for Japanese w = chi for Chinese
Yes
w
Count w found in English dictionary: we
Count w found in German dictionary: wg
Count w found in Spanish dictionary: ws
Count w found in French dictionary: wf
Count w found in Polish dictionary: wp
Increase counter if w = kor: wk
Increase counter if w = jap: wj
Increase counter if w = chi: wc
we
wg
ws
wf
wp
wk
wj
wc
m = total number of segments, F_W=(we/m, wg/m, ws/m, wf/m, wp/m, wk/m, wj/m, wc/m)
F_W
Classifier
output

Fig. 1 Proposed methodology

3.1 Overview

The proposed approach shown in Fig. 1 takes an UTF8 formatted sentence as an input. The irrelevant Unicodes from the input sentence, are removed through a preprocessing module before feature extraction. Based on the preprocessed input, Unicodes for each language are counted and divided by the total number of Unicodes. The second feature is to extract and count the language specific Unicodes for each language. Again, the count for exclusive Unicode of each language is divided by the total number of Unicodes in the preprocessed input sentence. The final feature is related to identifying each segment or word separated by a space. There are two different methodologies suggested for identification of each segment or word. Firstly, Latin character-based languages like English, German, Polish, French, Spanish, etc. put spaces between words. However, it is not true for Chinese, Japanese, Korean, etc. to distinguish words by spaces. So, it is more appropriate to call a component in a sentence separated by spaces a "segment". To identify each segment composed of Latin characters, a dictionary matching method is proposed. Heuristics are employed to identify each segment composed of Unicodes from Chinese, Japanese and Korean. Finally, the features are fed to a classifier to decide which language the input sentence belongs to and provide confidence values.

3.2 Unicode and UTF8

Unicode is a standard representation of characters to be expressed in computing [17]. UTF8 (8-bit Unicode Transformation Format) is a preferred protocol to encode the Unicode characters for storing or streaming them electronically [18, 19]. Characters for all languages are defined as ranges in UTF8 format [20].

3.3 Feature Extraction

The proposed methodology utilizes three features related to Unicode components in the pre-processed sentence. The three features are Unicode, exclusive Unicode and segments for each language.

3.3.1 Unicode Feature

$F_U = \{u_1, u_2, \ldots, u_n\}$, $u_m = \frac{g(m)}{t}$, $m = 1 \ldots n$, where *n* is the number of languages, *g(m)* is the total number of Unicodes for language *m*, and *t* is the total number of Unicode in an input sentence.

Example:

Input sentence: 'My name is 이상민 in Korean'

Unicode distribution:

English, German, French, Spanish, and Polish: M, y, n, a, m, e, i, s, i, n, k, o, r, e, a, n (Total: 16)

Chinese and Japanese: 0

Korean: 이, 상, 민 (Total: 3)

Total Unicode in the input sentence: 19

$F_U = \{16/19, 16/19, 16/19, 16/19, 16/19, 0, 0, 3/19\}$

3.3.2 Exclusive Unicode Feature

$F_X = \{u_1, u_2, \ldots, u_n\}$, $u_m = \frac{g(m)}{t}$, m = 1…n, where *n* is the number of languages, *g(m)* is the total number of exclusive Unicode for language *m*, and *t* is the total number of Unicode in an input sentence.

3.3.3 Segment feature

$F_W = \{S_1, S_2, \ldots, S_n\}$, $S_m = \frac{f(m)}{y}$, m = 1…n, where *n* is the number of languages, *y* is the total number of space-separated segments/words, and *f(m)* is the total number of space-separated segments/words identified by dictionary matching in a pre-processed input sentence.

3.4 Classification

The weighted sum feature classification and neural network based classification as described below have been investigated in this research.

3.4.1 Weighted Sum Feature Classification (WSFC)
The three features described in previous sections are extracted from the input sentence. The features are multiplied by a preset weight and sum of weighted features is calculated. The language with highest weighted feature value is selected.

3.4.2 Neural Classification (NC)
An overview of neural classification process is shown below in Fig. 2. The three features are extracted from the input sentence and fed to a neural classifier. The classifier fuses the features and gives the final confidence for each language. The final output contains the total score for each language. The neural network based classifier is trained using artificially generated training set before it is used for testing.

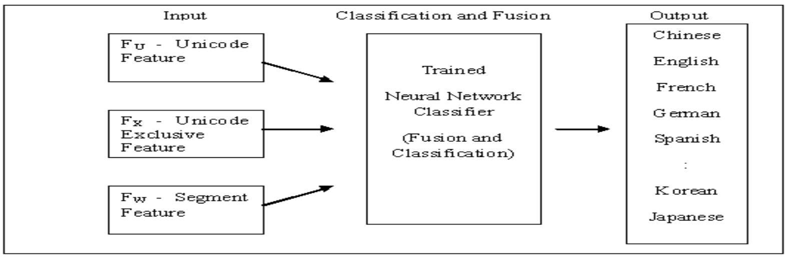
Input
Classification and Fusion
Output
F_U - Unicode Feature
F_X - Unicode Exclusive Feature
F_W - Segment Feature
Trained Neural Network Classifier (Fusion and Classification)
Chinese
English
French
German
Spanish
:
Korean
Japanese

Fig. 2. Overview of neural network classification process

4 Experiments and Results

The proposed methodology has been implemented in Java programming language. The experiments were conducted using weighted sum feature classification (WSFC) and neural classification (NC).

A small database of sentences with less than 10 words taken from web pages has been created. A news article for each language is selected and input samples for testing has been prepared by segmenting the article into sentences by finding a period symbol "." at the end of sentence. One hundred sentences for each language were collected, which give the total of eighty hundred sentences, and stored in an input file with UTF8 format.

The classification accuracies for experiments are shown in Table 1. The proposed approach has been compared to other methods in the literature. The proposed approach shows the similar performance over methodologies in [5, 8, 11, 12], but they are lower than the results from [9, 13]. However, considering the input language mode and the size of the input data, it is fair to conclude that the proposed approach were competitive to the existing approaches. Unlike the proposed approach, experimental results from [9] used longer input data size. The method in [11] has achieved the higher accuracy than the proposed approach on only mono input lingual mode. Finally, the approach in [13] achieved 100% accuracy on only English input documents.

Table 1 Experimental results

Technique	**Number of Sentences**	**Accuracy on Test Data [%]**
Proposed approach with WSFC	800	98.88
Proposed approach with NC		98.88

5 Conclusions and future research

In this paper, a novel approach for language classification has been presented and investigated. UTF8 encoding scheme has been used to construct the features for classification. The Unicode, exclusive Unicode and word matching score features in conjunction with a neural network are used to classify a language of an input sentence. Word matching score was extracted against a common word list of each language, rather than full length dictionaries, to simplify the computational searching cost. The experiments with the proposed approach produced very competitive results, considering the limited length of input sentences. In our future research, the focus will be on improving the training data for neural network and testing on shorter sentences.

Acknowledgements

This research work is supported by CQUniversity's industry collaborative small research grant and MyCyberTwin.

References

[1] Artemenko, O., Mandl, T., Shramko, M., & Womser-Hacker, C. Evaluation of a language identification system for mono- and multilingual text documents, Proceedings of the 2006 ACM symposium on applied computing, pp. 859-860, 2006.

[2] Dunning, T. Statistical identification of language, Technical report CRL MCCS-94-273, New Mexico State University, Computing Research Lab, March 1994.

[3] Hakkinen, J., & Tian, J. N-gram and decision tree based language identification for written words, IEEE workshop on automatic speech recognition and understanding (ASRU'01), pp. 335-338, 2001.

[4] Lins, R., & Goncalves, P. Automatic language identification of written texts. Proceedings of the 2004 ACM symposium on applied computing, pp. 1128-1133, 2004.

[5] Lu, S., & Tan, C. L. Retrieval of machine-printed Latin documents through word shape coding. Pattern Recognition, vol. 41, no. 5, pp. 1799-1809, 2008.

[6] Martins, B., & Silva, M. Language identification in web pages. Proceedings of the 2005 ACM symposium on applied computing, pp. 764-768, 2005.

[7] McNamee, P. Language identification: a solved problem suitable for undergraduate instruction. J. Comput. Small Coll., vol. 20, no. 3, 94-101, 2005.

[8] Muthusamy, Y., Barnard, E., & Cole, R. Automatic language identification: a review/tutorial. IEEE Signal Processing, vol. 11, 33-41, 1994.

[9] Poutsma, A. Applying monte carlo techniques to language identification. Proceedings of computational linguistics in the Netherlands (CLIN), vol. 45, pp. 179-189, 2001.

[10] Prager, J. Linguini: Language identification for multilingual documents. Proceedings of the 32nd annual Hawaii international conference on system sciences, vol. 2, p. 2035, 1999.

[11] Qu, Y., & Grefenstette, G. Finding ideographic representations of Japanese names written in Latin script via language identification and corpus validation. Proceedings of the 42nd annual meeting on association for computational linguistics, pp. 183, 2004.

[12] Romsdorfer, H., & Pfister, B. Text analysis and language identification for polyglot text-to-speech synthesis. Speech communication, vol. 49, no. 9, pp. 697-724, 2007.

[13] Selamat, A., & Omatu, S. Web page feature selection and classification using neural networks. Information sciences, vol. 158, pp. 69-88, 2004.

[14] Selamat, A., Ching, N. C., & Mikami, Y. Arabic script web documents language identification using decision tree-ARTMAP model. IEEE International conference on convergence information technology, pp. 721-726, 2007.

[15] The Unicode Consortium. The Unicode Standard, Version 5.0 (5th Edition). Addison-Wesley Professional, 2006.

[16] Unicode. (2008, June 26). Retrieved June 27, 2008, Wikipedia: http://en.wikipedia.org/wiki/Unicode

[17] Unicode/UTF-8-character table. Retrieved June 26, 2008, UTF8-CharTable: http://www.utf8-chartable.de/unicode-utf8-table.pl

[18] UTF-8. (2008, June 24). Retrieved June 26, 2008, Wikipedia: http://en.wikipedia.org/wiki/UTF-8

[19] Xafopoulos, A., Kotropoulos, C., Almpanidis, G., & Pitas, I. Language identification in web documents using discrete HMMs. Pattern recognition, vol. 37, no. 3, pp. 583-594, 2004.

[20] Yergeau, F. UTF-8, a transformation format of ISO 10646. In RFC 3629, internet engineering task force, 2003.