Style Markers Based on Stop-word List

Jan Rygl and Marek Medved'

Natural Language Processing Centre
Faculty of Informatics, Masaryk University
Botanická 68a, 602 00 Brno, Czech Republic
{rygl, xmedved1}@fi.muni.cz

Abstract. The analysis of author's characteristic writing style and vocabulary has been used to uncover the identity of authors of documents by both manual linguistic approaches and automatic algorithmic methods. The revealing of the gender, name, or age can help to expose pedophiles in social networks, false product reviews on the Internet servers, or machine translations submitted as manually translated texts.
These problems are predominantly solved by a combination of stylometry and machine learning techniques. Since the stylometry focuses on the author's style, word *n*-grams cannot be used as a style marker. Stop words are not influenced by a topic of documents, therefore they can be used to create style markers.
In this paper, we present a guidance on how to implement stop-word extraction and to include stop-words based style markers into a multilingual classification system based on the stylometry.

Keywords: style marker, stop-word list, corpus

1 Introduction

Anonymity is seen as the cornerstone of an Internet culture that promotes sharing and free speech. However, anonymity can also lead to crime. The uncovering of the true gender, name, or age of document authors can help to expose pedophiles in social networks, false product reviews on the Internet servers, or machine translations submitted as manually translated texts.

To reveal true identity of the document author, a variety of style markers have been used, with better or worse results. Style markers are documents features describing style and vocabulary of the author [1].

There are many stylometric features, such as word and sentence length, typography errors, vocabulary richness, punctuation marks, n-grams of syntactic labels, ... [2,3,4,5]

In the first place, a vector of at least 100 most frequent words was used in a vast majority of recent approaches [6]. In the absence of stop-word sources, this paper provides a technical report how to generate list of stop words and implement style markers based on stop-word list including recommendation of tools for text preprocessing.

Aleš Horák, Pavel Rychlý (Eds.): Proceedings of Recent Advances in Slavonic Natural Language Processing, RASLAN 2014, pp. 85–89, 2014.

2 Text preprocessing

To extract stop words from texts, we need to preprocess documents. Therefore, we advise to create a program for document preprocessing that takes raw text or HTML document as an input and outputs document objects that consist of:

1. raw source document
2. document language
3. character set that is used in the document
4. plain text without any HTML tags except a paragraph tag and a link tag where a diacritic check is provided on plain text's words
5. tokenization of the plain text
6. morphological annotation of the tokenized text
7. lemmatization of the tokenized text

The language of input HTML text can be determined by **langid** tool (for more information see [7,8]) that takes a text as a input and returns a language code in ISO 639-1 standard. Than the character set is derived from the input HTML text and it's language by **Chared** [9] (developed at Masaryk University in Brno). Information about the language of a document increases an accuracy of encoding detecton, therefore we recommend to do language detection before character set recognition.

Next we use `lxml.html.clean` from Python library (other tools depending on your programming language can be used) and get rid of all HTML tags except paragraph tags and links which can be useful for other style markers. In this step we also process all plain text's words by **czaccent** [10] (also developed at Masaryk University in Brno) tool that provide completion of diacritics if a word is spelled without or with incorrect diacritics. This process is necessary only for languages using characters with diacritics.

In the following step, the text is tokenized. We recommend to use **Unitok** [11] (universal tokenizing) program developed at Masaryk University in Brno that splits text into tokens and add predefined XML-like tags:

- `<doc>` – beginning of the document
- `<s>` – beginning of the sentence
- `</g>` – omitted space between tokens
- ...

After tokenization we pass the output into **Desamb** [12] tool (morphological desambiguator). The desamb uses morphological analyzer **Majka** [13] to annotate each word by morphological categories from the tagset of Majka and by lemmas. Then the best fitting variant of morphological category and lemma is selected for each token.

For our purposes we also train the **Majka** analyzer on Czech, Slovak and English data, thus our system can operate with this three languages. The output of desambiguation consists of morphological tags and lemmas for each token in the text. Lemmas statistics can be used instead of tokens to create a stop-word list used for style markers.

The whole process is illustrated in Figure 1.

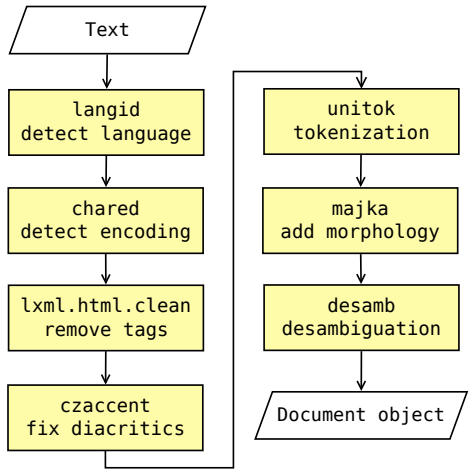

Fig. 1: Preprocessing text

3 Stop-word style markers

The purpose of this characteristic is to find the most frequent words (stop words) of given language in the text and provide style marker values for them.

3.1 Stop-word extraction from a corpus

To obtain list of stop words of a given language, **Sketch engine**'s [14,15] `freq` tool can be used on large corpora. We extracted stop words for Czech, Slovak and English documents:

- for Czech stop words we use `czTenTen` corpus that consists of 5 069 447 935 tokens
- for Slovak stop words we use `skTenTen` corpus that consists of 876 003 720 tokens
- for English stop words we use `enTenTen` corpus that consists of 12 968 375 937 tokens

To extract stop-word lists using Sketch engine, you can use two console commands:

```
command:
        freqs $corpora '[]' 'word 0 tag 0' > word_feq
        freqs $corpora '[]' 'lemma 0 tag 0' > lemma_feq
```

where `$corpora` is a name of the corpus. We used following corpora:

- sk: sktenten

- cs: cztenten12_8
- en: ententen12

The advantage of the Sketch engine is that is already contains corpora for almost every world language. If you do not have access to the Sketch engine and you have own corpora, you can generate stop words with frequencies using bash command line tools or any programming language.

3.2 Style markers extraction

This characteristic can use two input types thus user can choose if the calculations operate with lemmas or tokens (*word* represents a lemma or a token). Below we describe calculations on words but the same calculations can be provided on lemmas too.

We calculate two types of frequencies: the absolute frequency and the relative frequency of stop words that appear in the input text. This characteristic is used to generate three different predominant outputs:

1. relative frequencies of stop words:
$$\frac{count_{stop\_word}(document)}{count_{word}(document)}$$
2. absolute differencies of absolute values of relative frequencies of words from the corpus and relative frequencies of words in the input text
$$\left|\frac{count_{stop\_word}(corpus)}{count_{word}(corpus)}\right| - \left|\frac{count_{stop\_word}(document)}{count_{word}(document)}\right|$$
3. squared values of differencies of absolute values of relative frequencies of words from the corpus and relative frequencies of words in the input text
$$\left(\frac{count_{stop\_word}(corpus)}{count_{word}(corpus)} - \frac{count_{stop\_word}(document)}{count_{word}(document)}\right)^2$$

The first method ignores corpus frequencies and it is suitable for scenarios we are given stop-word lists, but frequencies are counted from an untrustworthy corpus or are unknown. For other situations, we recommend other two variants. The third variant is sensitive to big deviations from corpus frequencies which can be important style marker.

4 Conclusions

The style markers based on stop-word lists are easily implemented. To generate own style markers we recommend to use mentioned methods or their alternatives for other programming languages. Style markers based on stop-word lists are still considered to be very beneficial for the most of algorithms using stylometry, therefore each software solving these problems should implement them.

Acknowledgements This work has been partly supported by the Ministry of Education of CR within the LINDAT-Clarin project LM2010013.

References

1. Stamatatos, E., Fakotakis, N., Kokkinakis, G.: Automatic text categorization in terms of genre and author. Computational Linguistics **26**(4) (2000) 471–495
2. Mosteller, F., Wallace, D.L.: Inference and Disputed Authorship: The Federalist. Addison-Wesley (1964)
3. Holmes, D.I.: Authorship attribution. Literary and Linguistic Computing **13**(3) (1998) 111–117
4. Juola, P.: Authorship Attribution. Foundations and Trends in Information Retrieval **1** (2006) 233–334
5. Grieve, J.: Quantitative authorship attribution: An evaluation of techniques. Literary and Linguistic Computing **22**(3) (2007) 251–270
6. Eder, M.: Style-markers in authorship attribution a cross-language study of the authorial fingerprint. Studies in Polish Linguistics **2011**(Volume 6 Issue 1) (2011)
7. Lui, M., Baldwin, T.: Cross-domain feature selection for language identification. In: In Proceedings of 5th International Joint Conference on Natural Language Processing. (2011) 553–561
8. Lui, M., Baldwin, T.: Langid.py: An off-the-shelf language identification tool. In: Proceedings of the ACL 2012 System Demonstrations. ACL '12, Stroudsburg, PA, USA, Association for Computational Linguistics (2012) 25–30
9. Pomikálek, Jan and Suchomel, Vít: Chared: Character Encoding Detection with a Known Language. In: Aleš Horák, Pavel Rychlý. RASLAN 2011, Tribun EU, 5th ed. Brno (Czech Republic) (2011) 125–129
10. Rychlý, P.: CzAccent - Simple Tool for Restoring Accents in Czech Texts. In Horák, A., Rychlý, P., eds.: 6th Workshop on Recent Advances in Slavonic Natural Language Processing, Brno, Tribun EU (2012) 15–22
11. Michelfeit, J., Pomikálek, J., Suchomel, V.: Text Tokenisation Using unitok. In Horák, A., Rychlý, P., eds.: 8th Workshop on Recent Advances in Slavonic Natural Language Processing, Brno, Tribun EU (2014) 71–75
12. Šmerk, P.: K počítačové morfologické analýze češtiny (in Czech, Towards Computational Morphological Analysis of Czech). PhD thesis, Faculty of Informatics, Masaryk University (2010)
13. Šmerk, P.: Fast morphological analysis of czech. In: Proceedings of Third Workshop on Recent Advances in Slavonic Natural Language Processing, Brno, Tribun EU (2009) 13–16
14. Kilgarriff, A., Baisa, V., Bušta, J., Jakubíček, M., Kovář, V., Michelfeit, J., Rychlý, P., Suchomel, V.: The sketch engine: ten years on. Lexicography **1**(1) (2014) 7–36
15. Kilgarriff, A., Baisa, V., Bušta, J., Jakubíček, M., Kovář, V., Michelfeit, J., Rychlý, P., Suchomel, V.: Statistics used in the sketch engine. (2014)