Grammar Checker Features for Author Identification and Author Profiling

Overview

Hypothesis
The number and the types of grammatical and stylistic errors serve as indicators for a specific author or a group of people.

Challenges

- How to analyse the text and identify grammatical and stylistic errors?
- How to transform these errors into a representation suitable for the machine to fulfil the task?

Approach

- Prepossessing based on open-source NLP tools
- Open-source grammar checker as input
- Transformation into binary features & feature vectors
- Supervised Machine Learning & Information Retrieval techniques

The whole system is available as open-source: https://www.knowminelabsvn.sourceforge.net/pan2013.html

Feature Extraction

LanguageTool Features
The central component of our authorship identification and profiling system is a component to detect grammatical and stylistic errors within text, which has already been reported as suitable features for the task of author identification [1, 2]. Here we employ the opensource tool LanguageTool [3], which is a style and grammar checker. It works for 20 different languages and can be easily be extended to include additional rules. To illustrate the output of the LanguageTool library an example is depicted in figure 1, where two different types of errors are detected, where the example is directly taken from the PAN 2013 authorship identification data-set.

Figure 1: Example for a short snippet of text which contains 2 errors according to the LanguageTool. For the second missed violation, LanguageTool suggests: “Consider using a past principle here: ‘machined’.”

Basic Statistics Features

- Lines: Number of lines, number of characters per line, max line length, ...
- Sentence: Number of paragraphs, number of characters per paragraph, ...
- Document: Number of tokens, number of stop words, ratio of capital letters, ...

Vocabulary & Stylometric Features

- Ratio of alphabetic characters, ratio of white-space characters, ...

Stem Suffix Feature

The suffix of the words, which would be remove by a stemmer, i.e. the Snowball stemmer.

Slang Word Feature

Features generated out of the slang words contained within the text, where there are three lists of such words: Internet slang words, swear words and common smileys.

Sentence Structure Features

Features generated out of the Stanford Parser, which generates a parse tree and typed dependencies for the grammatical roles. There the depth of the parse tree and statistics of the types dependencies are taken as features. These features are by default disabled, as the parser component takes a considerable time to compute.

Author Identification

Feature Spaces
Each set of features is transformed into a feature space. For each feature space the features of all documents are combined into a single meta feature space. The binary features of the meta feature space are then the comparison of the reference documents and the text document: i) more than minimum, ii) less than maximum, iii) within minimum and maximum, and iv) about mean, which integrates the standard deviation.

The grammatical features are interpreted as sample of a probability distribution, each document being a single sample. This input data is smoothed and pair-wise compared between the reference documents and the text document. For the comparison the Kolmogrov-Smirnov test is used. Here the binary features are: i) same distribution for close matches, and ii) about the same distribution for less close matches.

Classification
For the final decision the binary of the meta feature space are combined.

Results
Additionally to the Pan 2013 data set we also report on the results from a evaluation data set generated out of the Pan 2012 data.

Author Profiling

Feature Spaces
For author profiling we just use two feature spaces (for the submitted system): i) output of the style and grammar checker and ii) word frequencies.

Algorithmic Approaches
We provide two main approaches: i) Language Models, and ii) the k-NN classification algorithm. Our system allows to play with any combination of algorithms and features spaces, where the algorithms are applied in sequence and the first algorithm which provides a score (ignoring ties) is taken as final result.

Language Model
Training: For each group (genres, age groups) a single Language Model is build from the training documents for each feature group.
Classification: Iterate over all features

k-NN Classifier
Training: Each document is treated a single instance, with the gender and age group of the author stored alongside.
Classification: Combine the similarity score from the top 3 nearest neighbours.

Results
We report the results on the training data set, which has been randomly split into 70% used for training and 30% for testing:

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Language</th>
<th>Age</th>
<th>10s</th>
<th>20s</th>
<th>30s</th>
<th>Gender</th>
<th>Male</th>
<th>Female</th>
</tr>
</thead>
<tbody>
<tr>
<td>k-NN + Figures</td>
<td>English</td>
<td>0.583</td>
<td>0.542</td>
<td>0.271</td>
<td>0.612</td>
<td>0.503</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Language Model + Grammar</td>
<td>English</td>
<td>0.565</td>
<td>0.531</td>
<td>0.271</td>
<td>0.649</td>
<td>0.576</td>
<td></td>
<td></td>
</tr>
<tr>
<td>k-NN + Figures</td>
<td>German</td>
<td>0.546</td>
<td>0.527</td>
<td>0.270</td>
<td>0.648</td>
<td>0.503</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Language Model + Grammar</td>
<td>German</td>
<td>0.565</td>
<td>0.531</td>
<td>0.270</td>
<td>0.648</td>
<td>0.576</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

References