Identifying Computer-Generated Text Using Statistical Analysis

Citation for published version:

Digital Object Identifier (DOI):
10.1109/APSIPA.2017.8282270

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published In:
2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference

General rights
Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.
Identifying Machine-Generated Text
Using Statistical Analysis

Hoang-Quoc Nguyen-Son*, Ngoc-Dung T. Tieu‡, Huy H. Nguyen‡, Junichi Yamagishi*†‡, and Isao Echizen*†

* National Institute of Informatics, Tokyo, Japan
† The University of Edinburgh, Edinburgh, United Kingdom
‡ The Graduate University for Advanced Studies, Kanagawa, Japan

Abstract—Computer-based automatically generated text are
used in various applications (e.g. text summarization, machine
translation) and such the machine-generated text significantly
helps our social life. However, machine-generated text may
produce confusing information sometimes due to errors or
inappropriate use of wordings caused by language processing,
which could be a critical issue in president elections or in product
advertisements. Previous methods for detecting such machine-
generated text typically extracts the text fluency, but, this may
not be useful in near future because recently proposed neural-
network based natural language generation results in improved
wording close to human-crafted one.

However, we hypothesize that the habit of human on writing is
still more consistent. For instance, the Zipf’s law states that the
most frequent word in the text written by human approximates
twice the second most frequent word, nearly three times the third
most frequent word, and so forth. We found that this is not
true in the case of machine-generated text. We hence propose a
method to identify the machine-generated text based on such the
statistics – First, word distributed frequencies are compared with
the Zipfian distribution to extract frequency features. Second,
complex phrase features are extracted to show that human-
generated text contains more sophisticated phrases than machine-
generated one. Finally, the higher consistency of the human-
generated text is quantified at both the sentence level using
phrasal verbs and at the paragraph level based on coreference
resolution relationships, which are integrated into consistency
features.

The combination of the frequency, the complex phrase, and
the consistency features is evaluated on a hundred of original English
books and a hundred of translated ones from Finnish. The result shows that our method achieves the better performance
(accuracy = 98.0% and equal error rate = 2.9%) comparing with
a state-of-the-art method using parsing tree feature extraction.
An advantage of this method is that this method can be used for
large collections of text such as books efficiently. Other evaluation
results in two other languages including French and Dutch
showed similar results. They demonstrated that the proposed
method works consistently in various languages.

I. INTRODUCTION

Machine-generated text plays a major role in modern life.
Techniques to generate texts automatically, natural language
generation, partly or entirely may replace humans in vari-
ous applications such as text summarization [1], header cre-
ation [2], machine translation [3], and image description [4].
Further, speech interfaces such as Apple Siri, Google Assis-
tant, and Microsoft Cortana also have the natural language
generation components and may use machine-generated
text as well as text crafted by human.

However, the quality and trustworthiness of the texts are
difficult to be verified. As a result, the information of the automatically
generated contents may be incorrect or inappropriate
compared with the information of the original contents written
by human truly. In worst cases, the machine-generated non-
trusted information may lead readers to misunderstanding.

Moreover, the machine-generated text could either make
customers annoyed in product advertisements or could give
viewers incorrect attitudes in politics. Additionally, more for-
mal writings such as scientific papers written by the machine,
which have been accepted by a few conferences in fact, may
destroy their reputations. We thus need a method to determine
whether a text is written by human or machine.

Numerous researchers have interests in the detection task of
machine-generated text. In the document level, most methods
estimate fluency of text [5] or word similarity quantifica-
tion [6]. In the sentence level, parsing trees are extracted as
discriminative features [7][8]. Our previous method extracted
two features from informal text at the sentence level: a density
feature using an N-gram language model and a noise feature to
be matched unexpected words (misspelling words, translated
error words, etc.) with original forms of words included in
the standard lexica [9]. The drawback of this method is that,
however, these unexpected words are easily recognized and
corrected by advanced assistant tools in formal text (e.g.
books, papers).

Although advanced natural language processing may improve
the naturalness and readability of the machine-generated
text, we hypothesize that the habit of human on writing is still
more consistent. For instance, it is known that word frequency
of human-generated text follows the Zipfian distribution [10],
which is called “Zipf’s law”. Additionally, we see that human-
generated text commonly use more complex phrases than
computer-generated text such as idioms ("long time no see"),

phrasal verbs ("get rid of"), ancient phrases ("thou"),

1https://medium.com/@samim/obama-rnn-machine-generated-political-
speeches-8c1ab18122ef
2https://pdos.csail.mit.edu/archive/scigen/
and cliche phrases ("only time will tell" means "to become clear over time"). Furthermore, the consistency of human-generated text is generally better than that of machine-generated one.

In this paper, we proposed a novel method to detect the machine-generated text using statistical features at the document level. Our contributions are listed below:

- We evaluate the word frequency distribution of the original and machine-generated documents. We find out that the human-generated text nearly follows the Zipfian distribution whereas machine-generated text does not. Therefore, a few parameters related to the Zipfian distribution are extracted from the text known as frequency features.
- We extract complex phrases from the text including idiom, cliche, ancient, and dialect by matching successive lemmas with the four standard complex phrase corpora, respectively. These extracted phrases are used to calculate complex phrases features.
- We also measure the consistency of the document at the sentence level using phrasal verbs and at the paragraph level using coreference resolution relationships. The number of phrasal verbs and coreference resolution relationships are considered as consistency features.
- We combine these statistical features including the frequency, the complex phrase, and the consistency features to create classifiers to determine whether the document is based on either machine- or human-generated text.

We evaluated our proposed method using two-hundred books in English and Finnish from project Gutenberg [11]: the hundred English books are considered as human-generated books. Then, the other hundred Finnish books translated into English by the Google translation service [3] are treated as machine-generated text. In the experiment, we compared our method with a parsing-tree-based feature extraction [6] because the method is strongly relevant to our method. The result shows that our method has achieved higher accuracy (98.0%) and lower error equal rate (2.9%) than the relevant method. We have also performed similar experiments in other languages including French and Dutch, which showed the similar results. These experiments demonstrated that the proposed method works well in various languages.

The structure of the paper is as follows: Section II introduces some of related work. Section III presents frequency feature extraction based on the estimated Zipfian distribution. The complex phrase feature extraction is discussed in Section IV. Thereafter, Section V describes the consistency feature extraction. The classifiers based on the combination of the frequency, the complex phrase, and the consistency features are described in Section VI. In Section VII, the experiments using original and translated books are presented and analyzed. Finally, Section VIII summarizes some main key findings and mentions our future work.

II. RELATED WORK

The detection task of machine-generated text is a well-known research problem. Some of the main methods at the document or sentence levels are summarized as below.

A. Document Level

Y. Arase and M. Zhou proposed a method that distinguishes machine-generated text from human-generated text [5] based on “salad phenomenon.” This salad phenomenon means that each phrase of machine-generated text is grammatically correct, but, when they put together, they are incorrect in terms of collocation [12]. Consequently, the authors estimate the salad phenomenon using an N-gram language model for continuous word sequence cases and using sequential pattern mining for isolated word cases. This method works well not only for the documents but also for sub-document levels such as sentence or phrase. This method is only evaluated on machine-translated text from Japanese to English. These languages completely different with word forms.

Other detection methods designed for larger scales of documents are text-similarity based approaches. For example, C. Labbé and D. Labbè has measured an inter-textual similarity of academic papers [13] using word distributions [6]. This assumption derives from the abundant reduplicated phrase patterns appeared in the machine-generated papers. The technique looks at technical terms and phrases only in corresponding fields (e.g. computer sciences, physics) because the text similarity in the machine-generated papers is nearly uniform in contrast to that of human-generated papers. However, this characteristic is obviously unsuitable for detecting machine-generated text in the general domain.

B. Sentence Level

Many researchers have successfully detected machine-generated text using the parsing trees at the sentence level. For example, J. Chae and A. Nenkova suggested a solution which quantifies the text fluency by extracting the main parsing components [7] such as phrase type proportion, phrase type rate, and head noun modifier. Moreover, they also exploited the use of incomplete sentence including the human-generated headlines and computer-translated errors.

Y. Li et al. also proposed another method using the parsing structure [8]. They showed that the parsing trees of human-generated text are more balanced than those of computer-generated ones. Based on these findings, the authors extracted several features related to the balance such as right-branching nodes, left-branching nodes, and branching weight index. The authors additionally showed that the emotion in the human-generated text is more abundant than in computer-generated one.

In our previous work [9], we extracted word density features using an N-gram language model on both internally limited corpus and huge external corpus. Futhermore, we found that the human-generated text frequently contains particular words such as spoken words (e.g., wanna, gonna) or misspelling words (comin, goin, etc.) whereas machine-generated one
frequently includes unexpected words which are created by mistakes of generators. These distinguishable words were called as noises. We then performed the detection of machine-generated sentences using the density and noise features.

In this paper, we extend the noise features of our previous method further. The previous features consider individual words only by matching each word with the standard lexica. We extend these features for complex phrases including idiom, cliche, ancient, and dialect. Moreover, several complex phrases are separated such as phrasal verbs, so they are not simply identified by the matching. We then propose a method to detect separable complex phrases using parsing tree tags.

To compare the proposed method with previous methods, we adopted the parsing based method suggested by Y. Li et al. [8] that calculates distinct parsing features for each sentence of a document. The average of the sentence features is then used to construct a classifier. The method is compared with our proposed method which combines frequency features, complex phrase features, and consistency features.

III. FREQUENCY FEATURES

We hypothesize that the word distributed frequency of the human-written text often follows Zipf’s law while computer-generated distribution does not. This law asserts that the distribution of the highest frequented words doubles with the occurrences of the second most frequented ones and triples with the third, and so forth. We use this evidence to distinguish the human-generated text from computer-generated text.

Frequency feature extraction is used to estimate how much an input document text t is compatible with the Zipfian distribution. The proposed scheme for extracting the frequency features is shown in Fig. 1:

- **Step 1 (Extracting linear regression line feature):** Each word in t is normalized by their lemmas. The lemma distribution is calculated and is used to estimate a linear regression function \( f = ax + b \) that is matched to the distribution. The slope feature \( a \) presented for the line is finally extracted.

- **Step 2 (Extracting information loss including square root \( R^2 \) and cost value \( C \)):** The quality of the linear regression line \( f \) is evaluated by two standard metrics. These metrics include the standard square root \( R^2 \) and a cost value \( C \) that measures the information loss.

The detail of each step to extract frequency features are described in below.

A. Extracting Linear Regression Line Feature (Step 1)

Due to word variations in English (such as “has,” “have,” “had”), we first need to normalize the original words in the input text t by their lemmas. The Stanford library [14] is used to convert variances to the same lemma here.

The number of lemma frequented distribution \( d_i \) is calculated. We then estimate the compatibility with the Zipfian distribution with the lemma distribution. According to the Zipf’s law, the distribution \( d_i \) of the \( i \)-th most common lemma is proportional to \( \frac{1}{i} \):

\[
 d_i \propto \frac{1}{i}.
\]  

Therefore, the slope \( a \) is considered as a major feature for detecting computer-generated text.

B. Extracting Information Loss (Step 2)

We quantify the information loss of the linear regression \( f \) via two standard metrics including square root \( R^2 \) and the cost value \( C \). The first one is calculated by:

\[
 R^2 = 1 - \frac{\sum_{i=0}^{N-1} (y_i - f_i)^2}{\sum_{i=0}^{N-1} (y_i - \bar{y})^2},
\]

where \( N \) is the number of distinct lemma, \( y_i \) is the distribution of the \( i \)-th lemma, \( f_i \) is the estimated value of \( i \)-th lemma by linear regression line \( f \), and \( \bar{y} \) is the value of \( i \)-th lemma on the mean distribution line \( \bar{y} \). The demonstration of these variables is shown in Fig. 3.

The other metric to quantify the information loss is a cost value \( C \) given in an equation below:

\[
 C = \frac{1}{2N} \sum_{i=0}^{N-1} (y_i - f_i)^2.
\]
IV. COMPLEX PHRASE FEATURES

The complex phrases, which are flexibly and commonly written in the human-generated text, are extracted as complex phrase features (Fig. 4):

- **Step 1a (Extracting idiom phrase feature I):** Idiom phrases are extracted from an input text \( t \) such as “long time no see” or “a hot potato” by matching with an idiom corpus. We use a standard idiom corpus\(^3\) suggested by Wikipedia with about 5000 distinct phrases. The use of idioms in a text may be different from the original idioms due to various word forms. Therefore, all words are standardized by their lemmas before matching. This standardization is also applied for other next steps. Additionally, all features in this section are divided by the number of words \( n \) in \( t \) for normalizing these features with documents with various lengths.

- **Step 1b (Extracting cliche phrase feature L):** Cliche words are commonly used in human-written text than computer-created one. Therefore, all cliche phrases are identified from the text \( t \) to create a cliche feature \( L \). The cliche phrase corpus used in here for matching is inherited from a Laura Hayden’s corpus\(^4\) with about 600 phrases.

- **Step 1c (Extracting ancient phrase feature A):** Other complex phrases known as ancient phrases also often occur in the human text. These archaic phrases are extracted by matching with a commonly ancient phrase corpus\(^5\) with about 1500 words. An ancient phrase feature \( A \) is measured using the extracted phrases.

- **Step 1d (Extracting dialect phrases features D):** Many deviations of English text can be used in similar contexts known as dialect phrases. Such phrases are identified by extracting contiguous lemmas including in a huge Yorkshire dialect phrase corpus\(^6\) with about 4000 phrases.

We only describe in detail of the Step 1a due to the similar of the four steps in this section.

**Extracting idiom phrase features I (Step 1a):** There are many variants of words in texts. Therefore, these words are standardized by their lemmas. In this step, we use Stanford parser library [14] to decide each lemma from separate words in an input text \( t \). Successive lemmas are combined and matched with each phrase in a candidate idiom phrase list. We utilize the standard idiom corpus suggested by Wikipedia\(^3\) as the candidate idiom phrase list. The idiom extract feature \( I \) is the division of the number of extracted idiom phrases and the number of words \( n \):

\[ I = \frac{\text{number of extracted idiom phrases}}{n} \]

---

\(^3\)https://en.wiktionary.org/wiki/Appendix:English_idioms
\(^4\)http://suspenze.net/whitefish/cliche.htm
\(^5\)http://shakespearestudyguide.com/Archaisms.html
\(^6\)http://www.yorkshiredialect.com/Dialect_words.htm
Input document text $t$

**Step 1a:** Extracting phrasal verb feature

**Step 1b:** Extracting coreference resolution feature

---

$\text{Step 1a (Extracting phrasal verb feature $P$):}$ Phrasal verbs includes separable phrasal verbs and inseparable phrasal verbs that are extracted from the input document text $t$. The number of the verbs phrases is divided by the number of words $n$ to create the phrasal verb feature $P$.

$\text{Step 1b (Extracting coreference resolution feature $S$):}$ Text consistency is also expressed via the coreference resolution relationships. Therefore, the number of coreference resolutions is extracted. This number is also normalized with the number of words $n$ for creating the coreference resolution feature $S$.

---

**V. Consistency Features**

Human-written text is frequently more consistency than machine-created one. The consistency is quantified by the phrasal verb feature $P$ and coreference resolution feature $S$ shown in Fig. 5.

- **Step 1a** (Extracting phrasal verb feature $P$): Phrasal verbs includes separable phrasal verbs and inseparable phrasal verbs that are extracted from the input document text $t$. The number of the verbs phrases is divided by the number of words $n$ to create the phrasal verb feature $P$.

- **Step 1b** (Extracting coreference resolution feature $S$): Text consistency is also expressed via the coreference resolution relationships. Therefore, the number of coreference resolutions is extracted. This number is also normalized with the number of words $n$ for creating the coreference resolution feature $S$.

---

**A. Extracting Phrasal Verb Feature $P$ (Step 1a)**

There are two kinds of phrasal verbs including separable or inseparable ones. For instance:

$s_1$ (inseparable phrasal verb): “The terrorists tried to blow up the railroad station.” (meaning: explode)

$s_2$ (separable phrasal verb): “It rained so they called the soccer game off.” (meaning: cancel)

The separable phrasal verbs are not recognized by matching in the same manner as complex phrases detection in Section IV. These verbs can be identified from the parsing-tree tags. Therefore, the Stanford NLP library [14] is used to generate the syntax tree parsing for each sentence in the document text $t$. The number of phrasal verbs is fitted with $PRT$ tag occurrence in these parsings. For example, in parsing tree of the sentence $s_1$ shown in Fig. 6, an inseparable phrasal verb is counted. In another example of the sentence $s_2$, a separable phrasal verb is recognized in Fig. 7.

The phrasal verbs are flexibly used in the human-created text. Otherwise, the machine often generates more simple phrases. Intuitively, machine prefers using uncomplicated phrases “explode” or “cancel” rather than phrasal verbs “blow up” in $s_1$ or “call off” in $s_2$, correspondingly.

The use of parsing tree avoids recognizing non-phrasal verbs such as a verb following by a preposition. For instance, in a
I voted for Nader because he was most aligned with my values,” she said.

Fig. 9. Three coreference resolution relationships demonstrations.

The ratio of the number extracted phrasal verbs and the number of words \( n \) is called as the phrasal verb feature \( P \):

\[
P = \frac{\text{Number of phrasal verbs}}{\text{Number of words}}.
\]

\( P \) = \text{phrasal verb feature}.

**B. Extracting Coreference Resolution Feature \( S \) (Step 1b)**

A human-written text is more consistency than a computer one. The number of coreference resolution relationships demonstrates the text cohesion. These relationships describe expressions referring to the same entity in the text. For instance, three relationships are shown in Fig. 9. The more of coreference resolution relationships, the higher possibility of human-generated text. We used the Stanford NLP tool [14] to extract coreference resolution relationships. The number of the relationships is used to measure the coreference resolution feature \( R \):

\[
R = \frac{\text{Number of coreference resolution relationships}}{\text{Number of words}}.
\]

**VI. COMBINATION**

The proposed scheme combines the frequency-related to Zipf’s law are estimated (Section III). They include the slope of the logistic regression line \( a \), root square \( R^2 \), and cost value \( C \).

**Step 1a (Extracting frequency features \( Q \)):** The frequency features \( Q \) related to Zipf’s law are estimated (Section III). They include the slope of the logistic regression line \( a \), root square \( R^2 \), and cost value \( C \).

**Step 1b (Extracting complex phrase features \( X \)):** All complex phrase features \( X \) including idiom phrase feature \( I \), cliche phrase feature \( L \), ancient phrase features \( A \), and dialect phrase feature \( D \) are extracted (see Section IV).

**Step 1c (Extracting consistency features \( T \)):** phrasal verb feature \( P \) and coreference resolution feature \( R \) are calculated for consistency features \( T \) (c.f. Section V).

**Step 2 (Detecting computer-generated text):** The extracted features from step 1a, 1b, and 1c are combined to determine whether the input text \( t \) is human- or computer-produced text using the best classifier from the popular machine learning classification algorithms.

Detecting computer-generated text (Step 2): The frequency features \( F \) in step 1a, the complex phrase features \( X \) in Step 1b, and the consistency features \( T \) in step 1c are integrated to determine whether the input text \( t \) is a computer-generated text. The features are processed with two popular classification algorithms, logistic regression and support vector machine. The support vector machines were optimized using either the sequential minimal optimization (SMO) algorithm [15] or the stochastic gradient descent (SGD) algorithm. Among the classifiers, the support vector machine optimized by SGD has achieved the highest performance in our experiments.

**VII. EVALUATION**

### A. Individual Features

We collected various books from Project Gutenberg [11], the biggest online free books. These collected books are released from 2003 to 2005. 100 original English books are used as human-generated text. 100 original Finnish books are translated by Google considered as machine-generated text. We evaluated the proposed method on two popular algorithms with 10-fold cross validation to create classifiers. The two classification algorithms involve logistic regression and support vector machine (SVM) used in here. The SVM were optimized using sequential minimal optimization (SMO) algorithm and the stochastic gradient descent (SGD) algorithm that are abbreviated by SVM(SMO) and SVM(SGD), correspondingly. The performance is quantified by accuracy and equal error rate (EER) metrics. The result of evaluation of individual features is shown in Table I.

The result points out the most important feature is ancient \( A \). Although these evaluated books are released in the same periods from 2003 to 2005, the ancient feature \( A \) reaches the best of performance for the all three classifiers. It shows that the translators trend to use uncomplicated words. The SVM(SGD) have the highest performance (accuracy = 89.0%, EER = 10.2%) with the feature \( A \) is used to create the final classifier for other experiments.
**TABLE I**

**EVALUATION OF INDIVIDUAL FEATURES**

<table>
<thead>
<tr>
<th>Group</th>
<th>Individual Features</th>
<th>Logistic Regression</th>
<th>SVM(SMO)</th>
<th>SVM(SGD)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Accuracy</td>
<td>EER</td>
<td>Accuracy</td>
</tr>
<tr>
<td>Frequency features Q</td>
<td>Slope $a$</td>
<td>53.5%</td>
<td>45.0%</td>
<td>58.5%</td>
</tr>
<tr>
<td></td>
<td>Root squared $R^2$</td>
<td>64.5%</td>
<td>26.0%</td>
<td>56.0%</td>
</tr>
<tr>
<td></td>
<td>Cost value $C$</td>
<td>73.5%</td>
<td>28.0%</td>
<td>74.0%</td>
</tr>
<tr>
<td>Complex phrase features X</td>
<td>Idiom $I$</td>
<td>74.5%</td>
<td>26.0%</td>
<td>76.0%</td>
</tr>
<tr>
<td></td>
<td>Cliche $L$</td>
<td>87.0%</td>
<td>33.0%</td>
<td>55.5%</td>
</tr>
<tr>
<td></td>
<td>Ancient $A$</td>
<td>88.5%</td>
<td>12.0%</td>
<td>88.5%</td>
</tr>
<tr>
<td></td>
<td>Dialet $D$</td>
<td>61.0%</td>
<td>38.0%</td>
<td>61.0%</td>
</tr>
<tr>
<td>Consistency features T</td>
<td>Phrases verb $P$</td>
<td>52.0%</td>
<td>47.0%</td>
<td>80.0%</td>
</tr>
<tr>
<td></td>
<td>Coreference resolution $S$</td>
<td>68.5%</td>
<td>31.0%</td>
<td>68.5%</td>
</tr>
</tbody>
</table>

**B. Combination**

We did similar experiments by combining the individual features in three groups: frequency features $Q$, complex phrase features $X$, and consistency features $T$. The result is compared with the most suitable method for books using parsing tree suggested by Y. Li et al. [8]. This method quantifies features for each parsing tree sentence. We adapted the method using the average of these features for the whole book. The result of comparison is shown in Table II.

This result indicates the influence of the group features. The group integration efficiently improves the individual group performances. Most integrations are better achievements than the parsing tree method [8]. The final combination of all features obtains the best performance (accuracy = 98.0%, EER = 2.9%).

**C. Other Languages**

We took the similar experiments in other languages. 100 English, 100 French, and 100 Dutch books are randomly chosen. The French and Dutch books are also translated into English by Google translation [3]. The performances of the proposed method are compared with the parsing tree method [8] shown in Table III.

The Table III shows that our method works well in other languages. Our performances are better than the parsing tree method [8] in both French and Dutch. These results demonstrate the consistency of proposed method with various languages.

**VIII. CONCLUSION**

People often use more sophisticated natural languages than computers. Specifically, the usage of words in the human text has commonly followed rules such as Zipf’s law. People also manipulate complex phrases (e.g., idioms, dialects, cliché, and ancient) more flexible than machines. Furthermore, the consistency of phrases in human text is generally higher than machine one. Therefore, we propose a method to distinguish computer- with human-generated text based on statistical analysis. More specifically, the frequency features are firstly extracted by estimating the word distribution with Zipfian distribution. Second, complex phrase features are calculated by matching successive lemma words in the text with complex phrase corpora. Finally, consistency features are measured with phrasal verbs and coreference resolution relationships. These three group features are combined to create a classifier. The classifier is evaluated with 100 original English books and
100 translated English books from Finnish. The result shows that the combination archived the best performance (accuracy = 98.0%, equal error rate = 2.9%) comparing with the most relevant method on large text [8] which extracts features from parsing trees. The similar performances with French and Dutch prove that the proposed method is high consistent with various languages.

In future work, we will evaluate our method on other kinds of documents such as novels or news. We also enhance the proposed features to segment a document into various parts in which are generated by either people or machines.

ACKNOWLEDGMENT

This work was supported by JSPS KAKENHI Grants (JP16H06302 and JP15H01686).

REFERENCES


