プログラムの類似度判定

16J5-003 秋野 勇輝

2019 年 4 月 4 日

1 概要

本実験の目的は、複数のプログラムコードの類似点を抽出し、類似の度合いを評価する手法の有効性を確認することである。

2 実験環境

- macOS Mojave 10.14.4
- Python 3.6.5

3 ソースコードの類似度とは

この実験ではソースコードの内容をテキスト情報として比較する。この手法は自然言語分析と内容が重複する点が多い。両者共に字句解析 (tokenize) から文脈判断 (lex) という手順を踏み、テキストの意味情報を抽出する手法が一般的であり、そのためのライブラリも豊富である。しかし、ソースコードについてはコンパイラ (インタプリタ) がその機能を既に持っており、その機能の一部をライブラリとして提供している場合がある (Python#tokenize)。

4 内容

この実験ではプログラムコードに出現する単語を元に類似度判定を試みる。コーディングスタイルの違いを吸収する為に Python の tokenize モジュールを用いて字句解析を行った結果からアルファベットのみを抽出 (記号類やコメントを除外) し、各単語の出現数を記録した。(図 1)

次に 2 つのコードの出現単語リスト同士を比較し、合致する単語と片方のみに含まれる単語の 3 つに分類する。この情報から単語の一致率を求めた。

5 結果

研究室 Wiki の添付ファイル一覧から、作成者が違うが同一の課題の回答例である Python プログラムを入手し検証素材とした。プログラムを実行した結果のログを次に示す。(図 2)

2 ファイルの平均 21.5 単語中 14 単語が一致しており、一致率は 65.1% であった。試しに全く違う機能のコード同士を比較した場合、システム予約後以外はほとんど一致しない結果となった。

問題点として、Python の tokenizer は大雑把な分類機能しか持たず、変数とメソッド名を区別することが難しいことが挙げられる。これにより変数名の仮名化を実装することが難しくなっている。また、システム予約語が判定精度に悪影響を与えている可能性が高いので集計対象から除くことも検討している。

6 まとめ

今回の実験では単語の出現頻度と一致数の可視化を行った。しかし、言い換えやプログラムの個性に影響しづらいシステム予約語への対応が出来なかった。

7 付録

参考文献

[1] 32.7. tokenize — python ソースのためのトークナイザ — python 3.6.8 ドキュメント. https://docs.python.org/ja/3.6/library/tokenize.html. (Accessed on 04/04/2019).

[2] c2nes/javalang: Pure python java parser and tools. https://github.com/c2nes/javalang. (Accessed on 04/03/2019).

[3] パッカー（packer） — マルウェア情報局. https://eset-info.canon-its.jp/malware_info/term/detail/00084.html, mar 2016. (Accessed on 02/25/2019).

[4] 誠岩村, 伊藤光恭, 洋一村岡. 機械語命令列の類似性に基づく自動マルウェア分類システム. 情報処理学会論文誌, Vol. 51, No. 9, pp. 1622–1632, sep 2010.


```
('cv2', 6)
('img_src', 6)
('np', 5)
('neiborhood8', 3)
('import', 2)
('array', 2)
('uint8', 2)
('iterations', 2)
('img_dst', 2)
('numpy', 1)
('as', 1)
('imread', 1)
```

図 1　集計例 (抜粋)

```
[Matched]
('cv2', 6)  ->  ('cv2', 8)
('img_src', 6) ->  ('img_src', 2)
('np', 5)  ->  ('np', 5)
('import', 2)  ->  ('import', 2)
('array', 2)  ->  ('array', 2)
('uint8', 2)  ->  ('uint8', 2)
('iterations', 2) ->  ('iterations', 6)
('numpy', 1)  ->  ('numpy', 1)
('as', 1)  ->  ('as', 1)
('imread', 1)  ->  ('imread', 1)
('IMREAD_GRAYSCALE', 1) ->  ('IMREAD_GRAYSCALE', 1)
('erode', 1)  ->  ('erode', 2)
('dilate', 1)  ->  ('dilate', 2)
('imwrite', 1)  ->  ('imwrite', 1)
 [Source1 only]
('neiborhood8', 3)  ->
('img_dst', 2)  ->
('neiborhood4', 1)  ->
('for', 1)  ->
('i', 1)  ->
('in', 1)  ->
('range', 1)  ->

 [Source2 only]
->   ('img', 12)
->   ('img_dest', 4)
->   ('neigh4', 3)
->   ('neigh8', 3)
->   ('def', 2)
->   ('closing', 2)
->   ('return', 2)
->   ('opening', 2)

 [集計]
合致数= 14
src1_only= 7
src2_only= 8
```

図 2　実行結果