

5 AI を使った自然言語処理の研究

10 番 昆 佑也

指導教員 佐々木 建

1. はじめに

私は,Google の翻訳機能を使うことがしばしばある.便利さを感じる反面,実際に翻訳した結果として脈絡がない文が表示されるケースが多く見られる.

機械学習関連の書籍などを見ると,「自然言語処理」について触れているものが多く,卒業研究の機会に,「自然言語処理」での様々な処理方法を調べてみたいと思い,卒業研究のテーマとして取り組むことにした.

2. 開発環境

今回の研究で使った環境を次の表 1 に示す.

OS	Linux(Ubuntu)
使用言語	Python3.7
使用ソフトウェア	word2vec 2.0 TextGenerator 0.1 MeCab 0.996 wp2txt 0.9.1

表 1 開発環境

3. 研究概要

当初は,「自然言語処理」の研究をするにあたり,様々な文献から,「MeCab」,「word2vec」及び「TextGenerator」が必要であることがわかり,研究の環境を整備した.しかし,そのツールを使うためには「wp2txt」も必要であることがわかり,研究の環境に追加していき,研究を進めた.使用した各々の技術については以下に示す.

3.1. 「MeCab」・「wp2txt」

Wikipedia の日本語版のデータをダウンロードしたものを「wp2txt」を使用し,XML 構文を取り除いたコーパス (テキストデータ) にし,それを「MeCab」により形態素解析で「分かち書き」処理をした.「分かち書き」とは,文節の切れ目ごと

に余白 (空白など) を入れることであり,その例を図 1 に示す.

図 1 テキストデータを「分かち書き」した例.

Wikipedia の表示されているデータから XML 構文要素を取り除きテキストデータにした.

3.2. 「word2vec」

前述の「MeCab」で「分かち書き」したファイルを,学習用のバイナリファイルに変換した後,「word2vec」で意味の近い単語をベクトル数値化した.(意味の近い単語ほどベクトル値を高く示す.)ベクトル値表記した例を図 2 に示す.

図 2 ベクトル値表記例.「榛名山」と入力して得られた,各山のベクトル値.

3.3. 「TextGenerator」

前述の Wikipedia コーパスを「TextGenerator」で解析し,自動的に文章生成をした.文章生成した例を図 3 に示す.

```

aiuser@ubuntu16:~/TextGenerator$ python GenerateText.py 5
#'''君の瞳''' (6分後に差し迫っており、野田事件があってもいい、Thayetkanという農村
食べることもある。1981年8月まで、セントジョージ病院の法律家であるというのは上記「支給
島県」[[静岡県清水区]: ネチズンが選んだ。水澤英洋(みらい文庫)週刊みらいエンパワメン
BN978-4797357165(ncid=BA8414020X|ref=NurekiVassilyev1995)}
aiuser@ubuntu16:~/TextGenerator$ python GenerateText.py 5
*北海道1位[[北陸大学]](2年1月7日に東京美術)2008-20095歳以上(総合得点420点以上(新
になった。*[[ホルヘ・テラロサ]]==脚注==*[[中露国境となった。
aiuser@ubuntu16:~/TextGenerator$ python GenerateText.py 5
合アレン]][[ゴム通り入口]からの応援により、彼は、千葉県出身のグループ展を開催して
00足キャンペーンを開始。===2019年2月20日に千葉地検松戸支部は、日本は、ビールニーの
すべての研究(中川淳司編)、吉松忠敬(卓蔵の息子に生まれた鹿嶋リン(凛)。
aiuser@ubuntu16:~/TextGenerator$

```

図3 テキストデータを「TextGenerator」で解析し、文章生成した例。

4. 結果・考察

4.1. 作業手順

今回の研究における作業手順を以下に列挙する。

4.1.1. Wikipedia のコーパスの作成

Wikipedia のダンプデータを「wp2txt」を使い XML 構文の部分を取り除き、Wikipedia コーパス (言語を分析するための資料) を作成した。

4.1.2. 「word2vec」でのベクトル数値化

前述で作成した Wikipedia のコーパスを「MeCab」を使い、「分かち書き」(文節、単語を空白で区切る)処理をした。

さらに「分かち書き」したファイルを「word2vec」を使い、ベクトル数値化をし、意味が近い単語ほどベクトル値が高いことを確認した。

4.1.3. 「TextGenerator」による文書の自動生成

「Wikipedia のコーパスの作成」で作成したコーパスを「TextGenerator」で解析し、自動的に文書を生成することを確認した。

4.2. 研究結果・考察

今回の研究では、主に「『word2vec』でのベクトルの数値化」と「『TextGenerator』による文書の自動生成」を主体するものだった。

```

/distance wiki.bin
to break): 比較
ulary: 31017
-----
Word          Cosine distance
-----
瑞鳳          0.661887
摩耶          0.643617
榛名          0.623237
神通          0.616512
鈴谷          0.615213
龍鳳          0.613246
鬼怒          0.606994
橋立          0.594036
ラミリーズ   0.589568
阿武隈       0.585657
沖鷹         0.573672
韓崎         0.571047
キアサージ   0.567656
菊月         0.566896
迅鯨         0.565987
那智         0.564780

```

図4 意味の近い単語のベクトル値が高く表示されている。

図4のように「『word2vec』でのベクトルの数値化」を実行したところ、書籍の例と同じように、意味の近い単語ほどベクトル値が高くなっていることが確認できた。様々な短い文章のデータを使用しても同じ傾向があることを確認した。

「『TextGenerator』による文書の自動生成」は同じ Wikipedia のコーパスのデータを使用して3回実行してみたが、3回ともすべて脈絡のない文章になってしまうことを確認した。本来であればもう少し機械学習要素を取り入れて脈絡がある精度の高い文章にする予定だったが、「wp2txt」のインストールが「Ruby」という環境に依存するため思っていたよりも時間がかかってしまうことも実際にわかり、最終的には、「wp2txt」での作業にとっても時間がかかって時間が足りなくなってしまった。

```

aiuser@ubuntu16:~$ wp2txt --input-file ./jwiki-latest-pages-articles.xml.bz2
[DEPRECATION] This gem has been renamed to optimist and will no longer be supported. Please switch to optimist as soon as possible.
WP2TXT is spawning 1 threads to process data
Preparing ... This may take several minutes or more ... Done.
jwiki-latest: 5% |oo | ETA: 01:44:18

```

図5 「wp2txt」での作業進行度(作業時間 時:分:秒)

5. 参考文献

自分で動かす人工知能 著:中島能和(なかじま よしかず) 出版:(株)インプレス

Python による AI・機械学習・深層学習アプリの作り方 著:クジラ飛行機、杉山陽一、遠藤俊輔 出版:ソシム(株)

Python によるスクレイピング&機械学習 著:クジラ飛行機 出版:ソシム(株)

Wikipedia (日本語版) 全文データをテキストファイルへ変換

URL:<https://qiita.com/oyaryo/items/203837d6f23a5495f2bb>

Wikipedia コーパスを気軽に使いたい人へ

URL:<http://jabberwocky.hatenablog.com/entry/2016/11/12/061709>