自然言語処理ライブラリ GiNZAの紹介

阿部　泰之

自己紹介

- 阿部 泰之 / Hiroyuki Abe
- twitter / @taki_tflare
- https://tflare.com

・機械学習を利用した新規事業開拓プロジェクトに所属しています。

書いたもの

一から始める機械学習（機械学習概要）

https://qiita.com/taki_tflare/items/42a40119d3d8e622edd2

機械学習で検索するとWikipediaの次に出てきます。

目次

- GiNZAの基本設計
- GiNZAの特徴
- 質疑応答

GiNZAとは

日本語自然言語処理オープンソースライブラリです。

GiNZAの基本設計

- フレームワークにspaCyを採用
- TokenizerにはSudachiPyを使用
- 依存構造解析学習データセットにはUD-Japanese BCCWJを使用
- 固有表現抽出の学習には京都大学ウェブコーパスを使用

https://www.slideshare.net/MegagonLabs/ginza-cabocha-udpipe-stanford-nlp p14より抜粋

GiNZAの特徴

- MITライセンスでモデルを含めて商用利用可能
- pip一行ですべて導入完了
- spacyの豊富な機能セットを利用できる

https://www.slideshare.net/MegagonLabs/ginza-cabocha-udpipe-stanford-nlp p21より抜粋

GiNZAの基本設計

- **フレームワークにspaCyを採用**
- TokenizerにはSudachiPyを使用
- 依存構造解析学習データセットにはUD-Japanese BCCWJを使用
- 固有表現抽出の学習には京都大学ウェブコーパスを使用

https://www.slideshare.net/MegagonLabs/ginza-cabocha-udpipe-stanford-nlp p14より抜粋

spaCy

spaCyは、Pythonで作られた自然言語処理（NLP）用のオープンソースライブラリです。

spaCyは、特に本番利用向けに設計されており、大量のテキスト処理を支援します。

spaCyの特徴

・非破壊的な字句解析
・固有表現抽出
・25以上の言語の字句解析サポート
・8言語の統計モデル
・事前学習済みの単語ベクトル
・品詞タグ付け
・ラベル付き依存構文解析
・統語ドリブンの文分割
・テキスト分類
・構文木および固有表現用のビルトインビジュアライザ
・ディープラーニング

SpaCy - Wikipedia より抜粋
https://ja.wikipedia.org/wiki/SpaCy

字句解析

```
[ ]  for token in doc:
         print(token.text, token.lemma_, token.pos_, token.tag_, token.dep_,
               token.shape_, token.is_alpha, token.is_stop)
```

↓

TEXT	LEMMA	POS	TAG	DEP	SHAPE	ALPHA	STOP
2019	2019	NUM	名詞-数詞	nummod	dddd	FALSE	FALSE
年	年	NOUN	名詞-普通名詞-助数詞可能	compound	x	TRUE	FALSE
10	10	NUM	名詞-数詞	nummod	dd	FALSE	FALSE
月	月	NOUN	名詞-普通名詞-助数詞可能	obl	x	TRUE	FALSE
に	に	ADP	助詞-格助詞	case	x	TRUE	TRUE
福岡	福岡	PROPN	名詞-固有名詞-地名-一般	iobj	xx	TRUE	FALSE
に	に	ADP	助詞-格助詞	case	x	TRUE	TRUE
行っ	行く	VERB	動詞-非自立可能	ROOT	xx	TRUE	FALSE
た	た	AUX	助動詞	aux	x	TRUE	TRUE
。	。	PUNCT	補助記号-句点	punct	。	FALSE	FALSE
天神	天神	PROPN	名詞-固有名詞-地名-一般	iobj	xx	TRUE	FALSE
に	に	ADP	助詞-格助詞	case	x	TRUE	TRUE
泊まり	泊まる	VERB	動詞-一般	ROOT	xxx	TRUE	FALSE
まし	ます	AUX	助動詞	aux	xx	TRUE	FALSE
た	た	AUX	助動詞	aux	x	TRUE	TRUE
。	。	PUNCT	補助記号-句点	punct	。	FALSE	FALSE

固有表現抽出

nlp()にわたすだけで様々な処理が終わっている。

固有表現抽出も終わっている。

```
doc = nlp('2019年10月に福岡に行った。天神に泊まりました。')
for ent in doc.ents:
    print(ent.text, ent.start_char, ent.end_char, ent.label_)
```

```
2019年10月 0 8 DATE
福岡 9 11 LOC
天神 16 18 LOC
```

displaCy機能を使うと以下のようにも表示可能

```
from spacy import displacy

displacy.render(doc, style="ent", jupyter=True)
```

2019年10月 DATE に 福岡 LOC に行った。 天神 LOC に泊まりました。

単語ベクトル

```
token = doc[5]
print(token)
print(token.vector)
```

```
福岡
[ 4.08136368e-01  5.78145385e-01 -7.17741430e-01 -3.60741735e-01
  5.50914645e-01  6.56163990e-01  7.51669943e-01 -4.22361225e-01
 -9.76256669e-01  1.08975410e+00  3.78202170e-01  1.84731460e+00
  1.40827966e+00 -1.56163621e+00  2.35466942e-01 -9.68343377e-01
  9.79801536e-01 -8.44053805e-01  3.93308327e-02 -4.72228289e-01
 -7.29781151e-01  1.99561679e+00  1.50306559e+00  1.41427612e+00
  7.24919140e-01 -9.95452762e-01 -5.65749258e-02 -9.79795873e-01
 -1.71541858e+00 -1.87483896e-02 -3.08172762e-01  8.40885043e-01
  2.25651407e+00  2.25738436e-01  2.51829052e+00  4.08718765e-01
 -2.65633404e-01 -9.65245843e-01 -2.17777006e-02 -1.03491747e+00
 -1.39604747e+00  1.00635104e-01  1.71456492e+00  1.51861691e+00
 -2.03297329e+00  1.70450699e+00  1.74121885e-03  4.81641352e-01
  1.27031684e+00 -2.68649030e+00  1.08589780e+00  1.99129057e+00
  7.02962697e-01  1.74541819e+00 -2.16280431e-01  1.17789090e-01
  4.80242789e-01 -1.77248991e+00  1.17088032e+00  7.52764642e-02
 -8.33266735e-01  1.27608025e+00 -2.02319670e+00 -5.53185761e-01
 -1.57263255e+00 -7.92438507e-01  2.03967378e-01  1.01418507e+00
  5.24024963e-01 -1.12104249e+00 -1.13011837e+00  3.59607369e-01
 -2.02457413e-01  1.61942518e+00 -9.69122112e-01 -2.32909346e+00
 -1.64884973e+00  2.21941397e-01 -5.04148364e-01  1.28840554e+00
 -1.80850017e+00  1.11083317e+00  4.35954705e-02 -8.72822046e-01
 -1.27225840e+00  1.26881897e-01  3.60635698e-01 -3.97512138e-01
  1.67796895e-01  1.19014931e+00 -1.86035669e+00 -1.30738974e+00
 -1.24575150e+00 -5.13757408e-01  1.12325764e+00  8.24675024e-01
  7.92933285e-01  2.05218220e+00 -1.33240998e+00 -6.17824852e-01]
```

spaCyの利点

spaCy をベースとした自然言語処理ライブラリについて、spaCy の API で日本語を処理できるようになった為、比較的小さな手直しで日本語対応させることが可能になった。

例えば、キーフレーズ抽出処理ライブラリである pke は、ストップワード部分の修正をすれば対応できる。

はじめての自然言語処理第5回 pke によるキーフレーズ抽出　を参考に記載
https://www.ogis-ri.co.jp/otc/hiroba/technical/similar-document-search/part5.html

GiNZAの基本設計

- フレームワークにspaCyを採用
- **TokenizerにはSudachiPyを使用**
- 依存構造解析学習データセットにはUD-Japanese BCCWJを使用
- 固有表現抽出の学習には京都大学ウェブコーパスを使用

https://www.slideshare.net/MegagonLabs/ginza-cabocha-udpipe-stanford-nlp p14より抜粋

Sudachi

形態素解析器Sudachi

Sudachiの特徴

1.　豊富な語彙

UniDicをベースにNeologdから大量に固有名詞を追加することにより得た豊富な語彙（280万語を超える登録規模）

2.　正規化表記

・例を上げると、「空き缶，空缶，空き罐，空罐，空きカン，空きかん」などに正規化表記として、空き缶を設定する。
・表記ゆれについてUniDicベースに、新聞でもかき分けがされているものは、別の語句としているなどの改善をしている。

3.　長期にわたる継続的なメンテナンス

開発方針の一つとして長期的なメンテナンスを上げており、新語の取り込み・機械的・人的なチェックにより辞書内容の拡張をしていくとしている

形態素解析器『Sudachi』のための大規模辞書開発　を参考に記載
https://pj.ninjal.ac.jp/corpus_center/lrw/lrw2018/P-1-08.pdf

Sudachiの特徴

以下開発に関わってられる方の記事に「今後10年は継続して更新していく予定」とある

Elasticsearchのための新しい形態素解析器 「Sudachi」

https://qiita.com/sorami/items/99604ef105f13d2d472b

4.　複数の分割単位での出力を追加

一般に形態素解析器はひとつの単位でしか出力できませんが、

SudachiではA,B,Cの3単位での出力が可能です。これはSudachiのシステム辞書に分割情報を付与することで実現しています。

例えば「医療品安全管理責任者」という入力のときには以下3種類の出力が可能です。

A: 医療 / 品 / 安全 / 管理 / 責任 / 者

B: 医療品 / 安全 / 管理 / 責任者

C: 医療品安全管理責任者

Elasticsearchのための新しい形態素解析器 「Sudachi」　より抜粋
https://qiita.com/sorami/items/99604ef105f13d2d472b

Sudachiの辞書

商用利用される形態素解析器としては，OSS として公開されている MeCab2, kuromoji3が大半を占めており，これらで利用可能な辞書としては，IPAdic，NAIST Japanese Dictionary，UniDic，NEologd などがある。
上記辞書には問題がある。
Sudachi辞書は、汎用的な辞書として使用できる大規模かつ高品質の辞書データの構築を目指している。

形態素解析器『Sudachi』のための大規模辞書開発を元に記載
https://pj.ninjal.ac.jp/corpus_center/lrw/lrw2018/P-1-08.pdf

辞書の問題点

辞書	問題点
IPAdic NAIST Japanese Dictionary	長年メンテナンスされていないため辞書内容が最新でない。
UniDic	言語の形態論的側面に着目して規定された短単位で見出し登録されている。 そのため，たとえば語義を取り扱いたい場合や語彙調査をする場合には そのままでは不足が生じる。
NEologd	複数の短単位から成る固有表現が一塊で登録されているため，そのまま検索システムで利用すると再現率が低くなる

形態素解析器『Sudachi』のための大規模辞書開発を元に記載
https://pj.ninjal.ac.jp/corpus_center/lrw/lrw2018/P-1-08.pdf

他の辞書と比べると

	IPAdic	UniDic	NEologd	Sudachi
複数の分割表現	×	×	×	○
固有表現	△	△	○	○
表記の正規化	○	○	×	○
継続的なメンテナンス	×	△	○	○
人手による精査	○	○	×	○

Sudachi ❤ Elasticsearch
https://speakerdeck.com/sorami/sudachi-elasticsearch?slide=36

GiNZAの基本設計

- フレームワークにspaCyを採用
- TokenizerにはSudachiPyを使用
- **依存構造解析学習データセットにはUD-Japanese BCCWJを使用**
- 固有表現抽出の学習には京都大学ウェブコーパスを使用

https://www.slideshare.net/MegagonLabs/ginza-cabocha-udpipe-stanford-nlp p14より抜粋

UD-Japanese BCCWJ

- UD Japanese-BCCWJ は現代日本語書き言葉均衡コーパス (BCCWJ) に付随する係り受け情報などを組み合わせて、UD へと変換、構築した BCCWJ の Universal Dependencie で ある。これは日本語の UD の中でも1980 文章、57,256 文、約 126 万単語を含む最大規模また複数のレジスターを内包したデータセットである。

UD Japanese-BCCWJ の構築と分析
https://pj.ninjal.ac.jp/corpus_center/lrw/lrw2018/P-2-01-E.pdf より抜粋

Universal Dependencies

Universal Dependencies (UD)は、構文解析の後段の処理の共通化や、他の言語のコーパスを用いた言語横断的な学習、言語間の定量的な比較などを可能にするための土台を目指して、多言語で一貫した構文構造とタグセットを定義するという活動である。

日本語 Universal Dependencies の試案
https://www.anlp.jp/proceedings/annual_meeting/2015/pdf_dir/E3-4.pdf

Chris ManningがあげるUDの6つの理念

1. 個々の言語の言語学的分析ができるものでなくてはならない

2. 言語ごとの比較をするのに適しているべき

3. 人間が早く一貫性を保ってアノテーションできる構造であるべき

4. コンピュータにとって高精度で解析できるものであるべき

5. 言語の学習者やエンジニアを含めて誰にとっても直感的な構造であるべき

6. 関係抽出・機械翻訳など、後段の処理で使えるものであるべき

Universal Dependencyの概要
https://pj.ninjal.ac.jp/corpus_center/pdf/2018-06-16-masayu-a-2.pdf

均衡コーパス

言語を分析するための基礎資料として，書き言葉や話し言葉の資料を体系的に収集し，研究用の情報を付与したものをコーパスと呼びます。

https://www.ninjal.ac.jp/database/type/corpora/

ある言語の使用実態をなるべく忠実に反映するようにバランス

良く設計・抽出されたコーパスは均衡コーパスと呼ばれます。

「自然言語処理の基本と技術」　コーパスと辞書

UD-Japanese BCCWJ

- 約126万単語、57256文と世界で3番目ぐらいの規模 新聞、雑誌、書籍、ヤフー知恵袋（Q&A）、ヤフーブログ、白書といったジャンルにまたがって 提供している

UD Japanese-BCCWJ:『現代日本語書き言葉均衡コーパス』のUniversal Dependencies

https://pj.ninjal.ac.jp/corpus_center/pdf/2018-06-16-mai-om.pdf p31より抜粋

GiNZAの特徴

- MITライセンスでモデルを含めて商用利用可能

- **pip一行ですべて導入完了**

- spacyの豊富な機能セットを利用できる

https://www.slideshare.net/MegagonLabs/ginza-cabocha-udpipe-stanford-nlp p21より抜粋

Google Colob

https://colab.research.google.com/drive/1PoChY4uvo5n1FCV9hKGe_hnALUHqu4Bg#scrollTo=o8Lu5Cis6tpv

Google Colobでのインストール

!pip install "https://github.com/megagonlabs/ginza/releases/download/latest/ginza-latest.tar.gz"

import pkg_resources, imp
imp.reload(pkg_resources)

macでのインストール方法

$ python3 -V
Python 3.7.3

$ sudo pip3 install “https://github.com/megagonlabs/ginza/releases/download/latest/ginza-latest.tar.gz"
$ sudo ginza

text = 銀座八丁目はお洒落だ
1　銀座　銀座　PROPN名詞-固有名詞-地名-一般_3compound_BunsetuBILabel=B|BunsetuPositionType=CONT|
SpaceAfter=No|NP_B|NE=LOC_B
2　八　8　NUM　名詞-数詞　NumType=Card3nummod_BunsetuBILabel=I|BunsetuPositionType=CONT|
SpaceAfter=No|NE=LOC_I
3　丁目　丁目　NOUN 名詞-普通名詞-助数詞可能_5nsubj_BunsetuBILabel=I|BunsetuPositionType=SEM_HEAD|
SpaceAfter=No|NP_B|NE=LOC_I
4　は　は　ADP　助詞-係助詞　_　3　case_BunsetuBILabel=I|BunsetuPositionType=SYN_HEAD|SpaceAfter=No
5　お洒落 御洒落 ADJ　名詞-普通名詞-サ変形状詞可能_0root_BunsetuBILabel=B|BunsetuPositionType=ROOT|
SpaceAfter=No
6　だ　だ　AUX　助動詞 _　5　cop　_　BunsetuBILabel=I|BunsetuPositionType=SYN_HEAD|SpaceAfter=No

GiNZAの特徴

- MITライセンスでモデルを含めて商用利用可能
- pip一行ですべて導入完了
- **spacyの豊富な機能セットを利用できる**

https://www.slideshare.net/MegagonLabs/ginza-cabocha-udpipe-stanford-nlp p21より抜粋

Google Colob

```
import spacy
nlp = spacy.load(‘ja_ginza')
doc = nlp('今期は自然言語処理の勉強会を実施しています。')

for sent in doc.sents:
    for token in sent:
        print(token.i, token.orth_, token.lemma_, token.pos_, token.tag_,
token.dep_, token.head.i)
    print(‘EOS’)
```

Google Colob

0 今期 今期 NOUN 名詞-普通名詞-一般 nmod 8

1 は は ADP 助詞-係助詞 case 0

2 自然 自然 NOUN 名詞-普通名詞-一般 compound 4

3 言語 言語 NOUN 名詞-普通名詞-一般 compound 4

4 処理 処理 NOUN 名詞-普通名詞-サ変可能 nmod 6

5 の の ADP 助詞-格助詞 case 4

6 勉強会 勉強会 NOUN 名詞-普通名詞-一般 obj 8

7 を を ADP 助詞-格助詞 case 6

8 実施 実施 VERB 名詞-普通名詞-サ変可能 ROOT 8

9 し 為る AUX 動詞-非自立可能 aux 8

10 て て SCONJ 助詞-接続助詞 mark 8

11 いき 行く AUX 動詞-非自立可能 aux 8

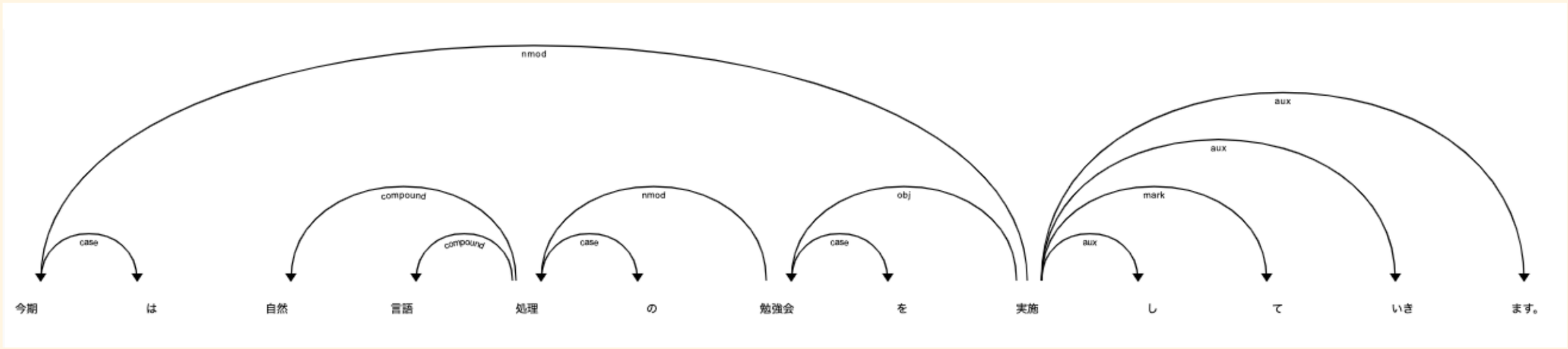
12 ます ます AUX 助動詞 aux 8

13 。 。 PUNCT 補助記号-句点 punct 8

EOS

Google Colob

```
from spacy import displacy
svg = displacy.render(doc, style="dep", jupyter=False)
```



GiNZAの今後

spacy公式言語モデルの提供

・GiNZAの機能のサブセットをspaCyのmasterブランチに統合予定

参考文献

- GiNZAで始める日本語依存構造解析 〜CaboCha, UDPipe, Stanford NLPとの比較〜

https://www.slideshare.net/MegagonLabs/ginza-cabocha-udpipe-stanford-nlp

Universal Dependencies 公開研究会

https://pj.ninjal.ac.jp/corpus_center/20180616.html