

ニューラルネットワークによる 自然言語処理 -前半-

前回の補足 : TensorFlow Hub

TensorFlow Hub

<https://www.tensorflow.org/hub?hl=ja>

TensorFlow Hub は、すぐに微調整してどこにでもデプロイ可能なトレーニング済み機械学習モデルのリポジトリです。BERT や Faster R-CNN などのトレーニング済みモデルを、わずか数行のコードで再利用できます。(Webサイトより)

実際のモデル一覧

<https://tfhub.dev/>

Hubはモデル一覧だけなので、チュートリアルがわかりやすい

<https://www.tensorflow.org/tutorials?hl=ja>

前半・後半でやること

前半：

Deep Learning以前の自然言語処理

自然言語をコンピュータ処理するためのいろいろな「お約束」

クラウド上でできること（自分でやるより遙かに賢い）

簡易コーパスの作成

後半：

Deep Learningを利用した自然言語処理

word2vecの利用・作成

BERT

自然言語処理(Natural Language Processing)

自然言語を対象とした情報処理

- 形態素解析
- 構文解析
- 意味解析
- 文脈解析
- 自動要約
- 自動翻訳
- 固有表現抽出

従来は主に辞書や文章（コーパス）を収集し、人手で意味づけを行い、それを元に統計的手法で行うことが多かった

NLPのこれまでと現在

形態素解析エンジンChaSenの開発で品詞分解（分かち書き）が容易にベクトル空間、SVMなどによって分類

Syntacticではなく、Heuristicな手法が試みられていた

近年はword2vecによるWikipediaコーパスやGoogle BERTによって飛躍的に精度が向上。BERTの登場後、XLNet, RoBERTa, ALBERTなどBERTを上回るものも発表された

Google Cloud Natural Language API

Google Cloud Natural Language API

<https://cloud.google.com/natural-language>

事前トレーニング済みモデルにより、感情分析、エンティティ分析、エンティティ感情分析、コンテンツ分類、構文分析などの自然言語理解の機能を使用可能

さらにカスタマイズする場合は、AutoML Natural Languageを使用する

実際にやってみると

伊豆諸島では、これまでの記録的な大雨で、土砂災害の危険度が非常に高い状態が続いている。引き続き土砂災害に厳重に警戒し、うねりを伴った高波に警戒が必要だ。

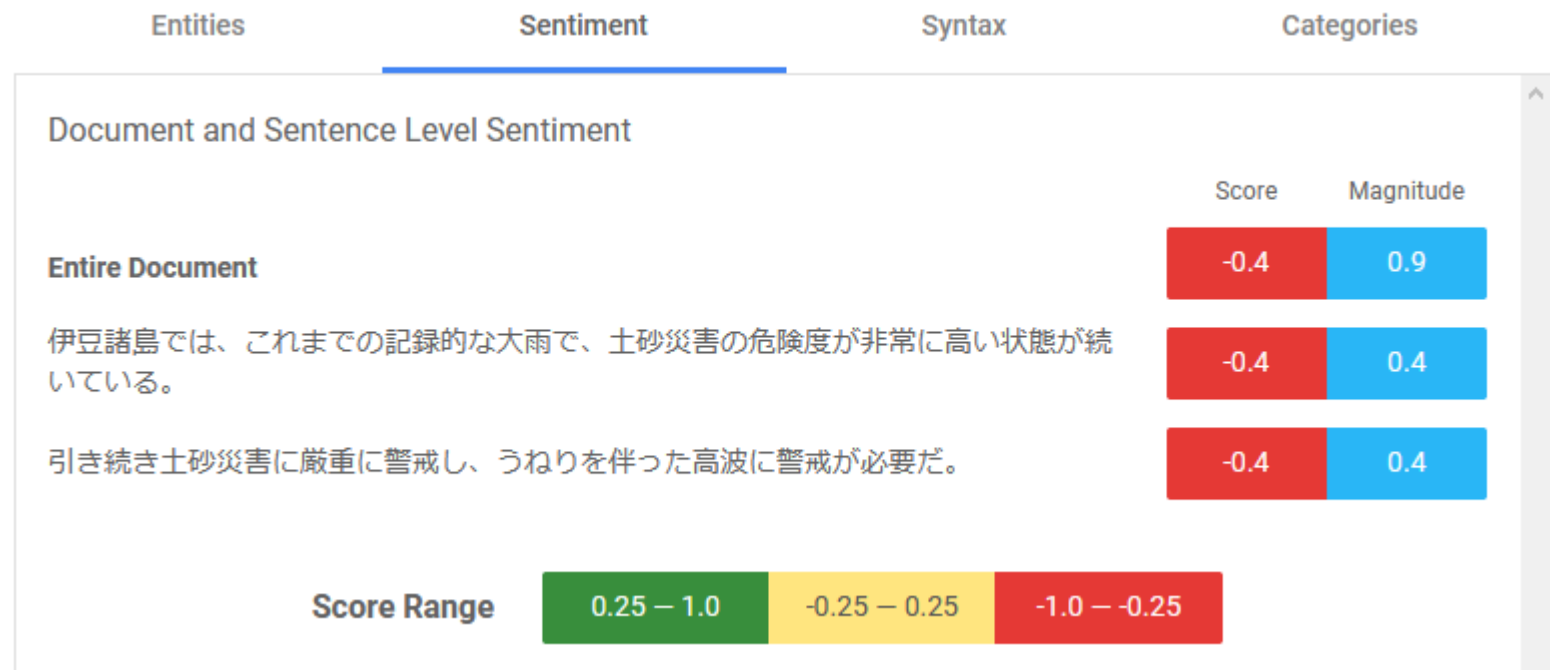
Entities	Sentiment	Syntax	Categories
<p>〈伊豆諸島〉₁では、これまでの記録的な〈大雨〉₃で、〈土砂災害〉₂の〈危険度〉₅が非常に高い〈状態〉₄が続いている。 引き続き〈土砂災害〉₂に厳重に警戒し、〈うねり〉₇を伴った〈高波〉₈に〈警戒〉₆が必要だ。</p>			
1. 伊豆諸島 Wikipedia Article Salience: 0.19	LOCATION	2. 土砂災害 Salience: 0.18	EVENT
3. 大雨 Salience: 0.15	EVENT	4. 状態 Salience: 0.12	OTHER
5. 危険度 Salience: 0.11	OTHER	6. 警戒 Salience: 0.09	OTHER
7. うねり Salience: 0.08	EVENT	8. 高波 Salience: 0.08	OTHER

エンティティ(実在性)
分析

感情分析(Sentiment)

Score: -1から1まで。-1がネガティブ、1がポジティブ

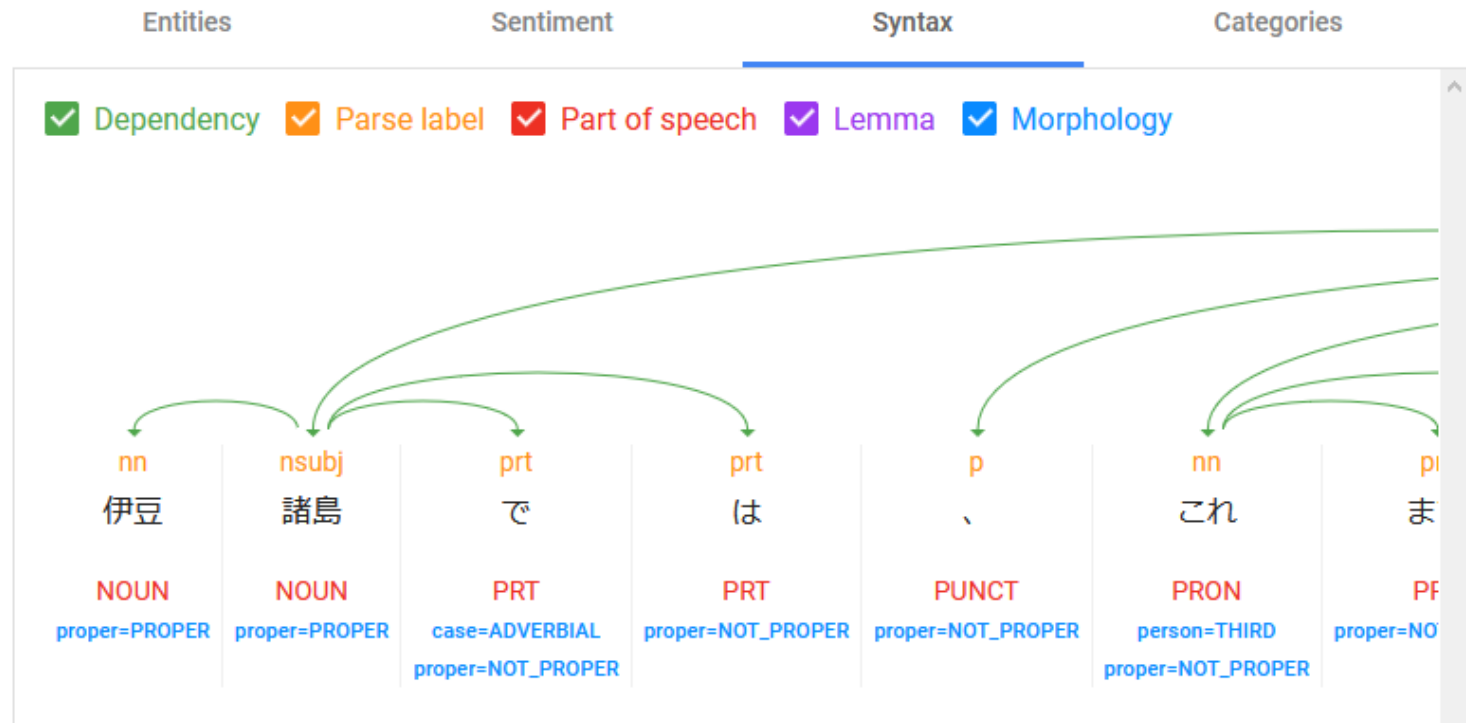
Magnitude: 指定したテキストの感情の強度（ポジティブ、ネガティブの両方）が正の値で示される。Scoreとは異なり、Magnitudeは正規化されていないため、テキスト内で感情が表現されるたびにMagnitudeが増加される。（長い文章ほど高くなる傾向）



構文解析

品詞と構造を分析

NOUN:名詞, PRT:助詞, VERB:動詞, PRON:代名詞, ADV:副詞, ADJ:形容詞
など



コンテンツ分類

日本語は未サポートなので、英語で……
(CNNのニュース文章より)

Try the API

A defiant President Donald Trump resumed public events Saturday with a divisive speech at the White House, where he potentially put lives at risk once again, just nine days after he revealed his own Covid-19 diagnosis.

RESET

[See supported languages](#)

Entities

Sentiment

Syntax

Categories

/News/Politics

Confidence: 0.98

[See a complete list of content categories.](#)

コーパス

自然言語の文章を構造化し大規模に集積したもの
実際に執筆され、使用された文章が主
毎日新聞コーパス(1995)、京都大学コーパス、Yahoo! 知恵袋データ
現代日本語書き言葉均衡コーパス、Wikipediaコーパス、など

国立国語研究所で収集・構築したコーパス

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

特に近年の大規模な日本語コーパスがKOTONNOHA

https://pj.ninjal.ac.jp/corpus_center/kotonoha-project.html

ただし、オリジナル文章には著作権が存在するため、一定の条件の元、主に学術的な用途に限定しているものがほとんど

コーパス横断検索

中納言 ※要登録

<https://chunagon.ninjal.ac.jp/>

あくまで学術・研究目的

申し込んでもすぐには登録されないなので、辛抱強く待ちましょう

形態素解析

文章を品詞分解するための処理

欧米系言語はスペースで単語が区切られているが、日本語は単語が連続している

よく知られたエンジンに、ChaSen, Juman, KAKASI, MeCabなどがある
速度や品質、使いやすさを考慮するとMeCabがダントツ

MeCab: 京都大学とNTT基礎研究所の共同開発

開発の中心である工藤拓氏は、京都大学、奈良先端科学技術大学院を経て現在Google

その他、AWSで利用できるSuidachiなど



<https://aws.amazon.com/jp/blogs/news/published-sudachidict-and-chive-on-aws-open-data/>

形態素解析の例

吾輩は猫である。名前はまだ無い。

吾輩	名詞,代名詞,一般,*,*,*,吾輩,ワガハイ,ワガハイ
は	助詞,係助詞,*,*,*,*,は,ハ,ワ
猫	名詞,一般,*,*,*,*,猫,ネコ,ネコ
で	助動詞,*,*,*,特殊・ダ,連用形,だ,デ,デ
ある	助動詞,*,*,*,五段・ラ行アル,基本形,ある,アル,アル
。	記号,句点,*,*,*,*,。,,。,,。
名前	名詞,一般,*,*,*,*,名前,ナマエ,ナマエ
は	助詞,係助詞,*,*,*,*,は,ハ,ワ
まだ	副詞,助詞類接続,*,*,*,*,まだ,マダ,マダ
無い	形容詞,自立,*,*,形容詞・アウオ段,基本形,無い,ナイ,ナイ
。	記号,句点,*,*,*,*,。,,。,,。

品詞分解するだけの処理を、分かち書きと呼ぶ

構文解析

文の構造を明らかにする処理。特に、係り受け解析がよく行われている
構文解析を行うプログラム構文解析器は、係り受け解析のCaboCha, 構文・
格・照応解析のKNPがある

CaboChaはMeCabと併用でインストールが容易

KNPは形態素解析器にJumanを使用し、コンパイル・インストールにかなり
時間がかかる

Yahoo! Japanのサービス

日本語解析Web APIとしてはかなり老舗（2007年開始）
元々2005年に検索エンジンの外部APIを解放し、徐々に広げていった
<https://developer.yahoo.co.jp/webapi/jlp/>

形態素解析、ルビ振り、係り受け解析、キーフレーズ抽出、自然言語理解など
おおよそのサービスは揃っている

固有値表現抽出

参考スライド：実務で使う固有表現抽出

<https://speakerdeck.com/sansandsoc/eigen-extraction-used-in-practice>

辞書・コーパスが重要

アノテーション（タグ付け）、なんとかルールを見つける

かつてWebメディアに対してアノテーションには、HTMLタグの情報を手がかりにする方法があった（タグに注目するなど）

演習

形態素解析 MeCab

係り受け解析 CaboCha

簡易コーパスの作成

形態素・係り受け・特徴抽出器 GiNZA

NLPにおけるベクトル化と距離

ある文書Aに含まれるすべての文章を形態素解析等で分解し、それぞれの単語の出現回数を数え上げる。文書Bにも同じ処理を行う

この結果を、すべてベクトルに置き換える

例) 単語: 青空 海 幸せ 食事 寝る

$A = \{ 200, 55, 45, 2, 30, \dots \}$

$B = \{ 3, 2, 120, 39, 35, \dots \}$


単語がn個あれば、n次元ベクトルになる

お互いの文書間を、青空と海といった2つの要素で比較する場合、 $A = \{200, 55\}$ と $B = \{3, 2\}$ のユークリッド距離を計算する。もし他に文書Cがあり、 $C = \{4, 10\}$ であれば、当然文書Bと文書Cはこの2つのパラメータにおいて近いと言える

本当に数え上げだけで良いのか？

文書の特徴は、単に単語の出現数だけではないはず
例えば、各単語に適切な重み付けを行い、より特徴が出たベクトルに変換する
重み付けとしてよく使用されるのがTF-IDF

$$\begin{array}{l} \text{青空} \\ \text{海} \\ \text{幸せ} \end{array} \begin{pmatrix} 200 \\ 55 \\ 45 \end{pmatrix} \circ \begin{pmatrix} 0.1 \\ 0.2 \\ 0.6 \end{pmatrix} = \begin{pmatrix} 20 \\ 11 \\ 27 \end{pmatrix} \quad \text{特徴ベクトル}$$

重み付け行列 

重み付けによって、「青空」よりも
「幸せ」の方が大きい値になる

TF-IDF

TF: Term Frequency **単語頻度**

IDF: Inverse Document Frequency **逆文書頻度**

ある文書において、D 個の文、N 個の単語があるとき、単語tがn回現れるとTFは

$$TF = \frac{n}{N}$$

また、単語tを含む文がd個のとき、IDFは以下

$$IDF = -\log_{10} \frac{d}{D} = \log_{10} \frac{D}{d}$$

TF-IDFは上記の積のため

$$TF - IDF: TF \cdot IDF = \frac{n}{N} \log_{10} \frac{D}{d}$$

TF-IDFの特徴

TFは各文書においてその単語がどのくらい出現したのかを表す

一方、IDFは逆頻度のため、よく登場する単語は低い値に、あまり登場しない単語は高い値になる

そのため、その文書において、あまり登場しないが重要な語の抽出手法としてよく用いられている

また、様々な文書や単語についてTF-IDFを計算し、文書間の類似度を計算することができる

TF-IDFの例

夏目漱石の小説での例

吾輩は猫である		こころ	
事	1270.00	私	2695.00
もの	981.00	先生	597.00
君	973.00	事	575.00
主人	932.00	k	529.24
吾輩	816.10	奥さん	388.00
御	636.00	人	388.00
人	602.00	時	375.00

出典: Pythonによるテキストマイニング入門(著: 山内長承, オーム社, 2017年)

N-gram

文章を、N個の文字で分割するテキスト分割方法

N=1: uni-gram, N=2: bi-gram, N=3: tri-gram

例) 今日はいい天気です

N=1: 今,日,は,い,い,天,気,で,す

N=2: 今日,日は,はい,いい,い天,天気,気で,です,す

N=3: 今日は,日はい,はいい,いい天,い天気,天気で,気です,です

文法解析を行わないので、言語の種類を問わず、**良くも悪くも言葉の意味を参照しない**という特徴がある

N-gramの使用例

日本語処理には、bi-gramあるいはtri-gramがよく用いられる
特に、単語間の共起頻度を計算し、文書や言語の特徴を調べる研究によく用いられている。また、全文検索に用いられている
ただし、全文検索エンジンNamazuはN-gramではなく形態素解析。作者の高林哲はSONY CSLを経てGoogle

※共起関係：文字列Xと文字列Yが同時に出現すること

演習

TF-IDF

N-gram