# LexRank を用いた抽出型要約の検証

## 1　はじめに

インターネット上にはニュース記事などの文書データが溢れ，多くの人々が情報源として利用している．しかしそれらの中には冗長な文章が多く存在しているため，簡潔に要約する手法は強く求められている．また Twitter などのように文字制限のあるメディアにおいても要約の必要性は大きい．

これらの要求を満たすためにこれまでに様々な文書要約手法が提案されてきている．自然言語処理を用いた文書要約は生成型要約と抽出型要約の 2 種類の方法に大別される．生成型要約は要約元となる文書の意味を解釈して文章を再構築し，新しい文を生成することで要約する．それに対して抽出型要約は，要約元の文書中に存在する代表文や重要文を抽出することで要約をする．生成型要約をコンピュータでするには難しく，これまでに提案されてきた文書要約手法のほとんどは抽出型要約に属している．しかし抽出型要約だけでは人間の作るような自然な要約文を作ることは難しく，その点から，今後生成型要約の研究が盛んになると考えられる．ただし，生成型要約は文生成の際に抽出型要約で抽出された重要文を使用することもよくあり，抽出型要約の精度向上も非常に重要である．

抽出型要約としてこれまでに LexRank を代表とした多くの文書要約手法が提案されている．LexRank は文書間の類似度から比較的簡単に文の重要度が計算できるので抽出型要約におけるベースラインとしてよく用いられている．

そこで，本稿では抽出型要約のベースラインである LexRank による重要文抽出の性能評価を目的とする．また LexRank に用いる文書ベクトルとして TF-IDF によるベクトルと，Doc2Vec による分散表現ベクトルの二つを使用し，両者のベクトルによる違いも検証する．

## 2　LexRank

LexRank [1] は Erkan らによって提案された PageRank [2] に着想を得た抽出型文書要約アルゴリズムで，以下の 2 つの観点に基づいて重要文が抽出される．

- 多くの文に類似する文は重要な文である
- 重要な文に類似する文は重要な文である

LexRank では，初めに文書間の類似度を計算して，それが閾値 t より大きければ文書ノード間にエッジが存在するものとする．そしてエッジが存在すれば 1，存在しなければ 0 としてグラフを表現する隣接行列を作成する．このとき，類似度計算に TF-IDF により作られるベクトルが用いられる．次に隣接行列の各要素をノードの次数で割り確率行列に変換し，この確率行列を元にマルコフ定常分布を求める．マルコフ連鎖は非周期的で必ず収束することが保証されているため，べき乗法で固有ベクトルを求めることができ，この固有ベクトルが文の重要度になる．

LexRank の擬似コードは以下のとおりである．

**Algorithm 1** Calculate LexRank Scores

```
Input: n 個の文章をもつ配列 S，閾値 t
Output: LexRank Scores を持つ配列 L
Array CosineMatrix[n][n];
Array Degree[n];
Array L[n];
for i = 0 to n do
  for j = 0 to n do
    CosineMatrix[i][j]=idf-modified-
    cosine(S[i],S[j]);
    if CosineMatrix[i][j]< t then
      CosineMatrix[i][j]=1;
      Degree[i]++;
    else if  then
      CosineMatrix[i][j]=0;
    end if
  end for
end for
for i = 0 to n do
  for j = 0 to n do
    CosineMatrix[i][j]=CosineMatrix/Degree[i];
  end for
end for
L=PowerMethod(CosineMatrix,n,ϵ);
return L;
```

# 3 TF-IDF

TF-IDF は文書中の単語の重みを表す手法の一つであり，主に情報検索や文章要約などの分野でよく利用される．

TF(Term Frequency) はそれぞれの単語の文中での出現頻度を表し，$n_{t,d}$ を単語 $t$ の文 $d$ 内での出現回数，$\sum_{s\in d} n_{s,d}$ を文書 $d$ 内のすべての単語の出現回数の和とすると，ある文 $d$ におけるある単語 $t$ の TF は

$$tf(t,d) = \frac{n_{t,d}}{\sum_{s\in d} n_{s,d}} \tag{1}$$

となる．

IDF(Inverse Document Frequency) はそれぞれの単語がいくつの文書内で共通して出現するかを表し，$N$ を全文書数，$df(t)$ をある単語 $t$ が出現する文書の数とすると，ある単語 $t$ の IDF は

$$idf(t) = \log \frac{N}{df(t)} + 1 \tag{2}$$

となる．

これらをかけ合わせたものが TF-IDF となり，ある文 $d$ における単語 $t$ の TF-IDF は

$$tfidf(t,d) = tf(t,d) \cdot idf(t) \tag{3}$$

となる．

よってすべての単語について TF-IDF を求めることで，ある文のベクトルができる．このベクトル間でコサイン類似度を計算することで，文書間の類似度を計算することができる．コサイン類似度の計算は以下の通りである．

$$cos(x,y) = \frac{\sum_{i=1}^{N} x_i \cdot y_i}{\sqrt{\sum_{i=1}^{N} {x_i}^2} \cdot \sqrt{\sum_{i=1}^{N} {y_i}^2}} \tag{4}$$

# 4 Doc2Vec

Doc2Vec [3] は Word2Vec [4] という手法をもとに，文章の分散表現ベクトルを得る手法である．Word2Vec は単語を高次元のベクトルとして表現する定量化手法で，これまでコンピュータで定量的に捉えることの難しかった言葉の意味を直接的に表現できる．また Word2Vec では単語の類似度やベクトル間の加減算が可能で,「王」-「男」+「女」=「女王」などの演算ができる．Word2Vec は同じ文脈にある単語はお互いに近い意味を持っている，という考えをもとに作られており，ある単語が与えられたときに近くに出現する単語を当てるという問題の解をニューラルネットワークに学習させることで単語のベクトルを得ることができる． Doc2Vec では単語ベクトルとパラグラフベクトルをつくり，平均化，結合，推定を経て分散表現が作られる．こうすることで，TF-IDF のような単語レベルではなく意味レベルで文書間の類似度が計算できる．

# 5 実験方法

TF-IDF によるベクトルを LexRank の類似度計算に適用し，重要文を抽出した．要約のテスト文書として Steve Jobs のスピーチ (和訳) [5] と livedoor [6] のニュース記事を選んだ．LexRank は TF-IDF によるベクトルを使用するので，同じ単語やフレーズを多用するスピーチでは重要文が比較的容易にとれると予想して Steve Jobs のスピーチを選んだ．livedoor はインターネット上でニュース記事とそれを 3 文で要約したものを掲載しており，この要約文を評価に利用できると考え選択した．実験では，livedoor の 3 文の要約文を正解文に設定して，LexRank により抽出された要約文と正解文を比較することで，重要文が抽出できるかどうか検証した．

実験の手法は以下の通りである．文書ベクトル生成の際に，文中のすべての単語を用いた手法と名詞のみを用いた手法の 2 通りを検証した．名詞のみを用いた際には，指示代名詞である「こと」,「それ」などは除去した．単語の抽出には形態素解析ツールの Mecab を利用した．また LexRank の上位 3 文を要約文の出力とし，評価は人手でした．

Steve Jobs のスピーチは 166 文で構成され，総出現単語数は 813，名詞の総出現数は 449 であった．livedoor のニュース記事は 10 文から 24 文で構成される記事を 10 件を選んだ．隣接行列を作る際の閾値はすべて 0.2 に設定した．

## 5.1 実験結果

はじめに Steve Jobs のスピーチの要約の結果を示す．要約結果は以下のようになった．

- すべての単語を用いた Steve Jobs の要約文

  1. 私は大学を卒業したことがない.
  2. そして偉大な仕事をする唯一の方法は自分がしていることをたまらなく好きになることだ.
  3. 私は自分がしていることがたまらなく好きだ.

- 名詞のみを用いた Steve Jobs の要約文

  1. 自分がしてきたことが，まだたまらなく好きだった.

  2. たまらなく好きなことを見つけなければならない.

  3. 「ハングリーであり続けろ．愚かであり続けろ」.

次に livedoor のニュース記事の要約結果の一例を示す．livedoor に掲載されている要約文と要約結果は以下のようになった．

- livedoor の要約文

  1. 支持率が低下している安倍内閣の退陣には3つのシナリオが想定できるという

  2. 健康を理由にするか、党内の不満が高まり抑えが効かなくなっての退陣と筆者

  3. 野党再編が起こり、非自民の受け皿ができて政権を失うことも考えられるそう

- すべての単語を用いたニュース記事の要約文

  1. しかし野党の凋落が政権の緩みを生み，国民の潜在的な不信感，不満を広げてきたのです.

  2. 安倍総理の健康不安「結婚 30 周年」記念日に主治医が私邸に駆け付けていた.

  3. 皮肉なことに，自民党の脆弱化は，民進党が想定以上に国民から見放され凋落してきたにも原因があるのです.

- 名詞のみを用いたニュース記事の要約文

  1. そして安倍内閣へ距離を置きはじめたメディアの報道が，次の総選挙に向けた不透明感を党内に広げはじめたのでしょう.

  2. しかし野党の凋落が政権の緩みを生み，国民の潜在的な不信感，不満を広げてきたのです.

  3. 安倍内閣にとっては致命傷になりかねない，安倍総理の健康不安説までもが，すでに流れ始めています.

これらの結果より Steve Jobs のスピーチではすべての単語を用いた場合でも名詞のみを用いた場合でも,「大学」,「卒業」,「たまらなく好き」,「ハングリーであり続けろ」,「愚かであり続けろ」のように繰り返し現れる単語やフレーズを抽出できていることが見てとれる．また名詞のみを用いた場合，上位 3 件目にスピーチの重要文と考えられる「ハングリーであり続けろ．愚かであり続けろ」という文が抽出できており，ある程度は重要文が抽出できているといえる．

一方でニュース記事は，livedoor に掲載されている正解文である要約文と比較した際に，ある程度重要文を抽出できる記事とほとんど抽出できない記事があった．正解文と比較して，似た内容を一部でも抽出できた記事は 10 件中 8 件あり，残りの 2 記事は抽出できなかった．これはニュース記事が短文で，単語の重なりが少なかったために，重要語の抽出が十分にできなかったと考えられる．このことから，LexRank による要約は，スピーチのように文章が十分長く，単語やフレーズが繰り返し現れる文章に適していると考えられる．

## 6 追加実験

先の実験では TF-IDF を用いた文書ベクトルを LexRank の類似度計算に適用した．実験結果から LexRank による重要文抽出は，スピーチなど十分に長く同じ単語を多くもつ文には効果的だが，ニュース記事のように比較的短く単語の重複が少ない文では，うまく重要文を抽出できない場合があるとわかった．そこで単語の意味が顧慮される Doc2Vec による分散表現ベクトルを用いて LexRank の類似度計算をすることで，ニュース記事のような短文でもうまく重要文を抽出できるのではないかと考え，追加実験をした．分散表現を類似度計算に適用し，先の実験結果との違いを確認する実験をした．

要約のテスト文書は先の実験と同じものを使用した．隣接行列を作る際の閾値は 0.2 に設定した．Doc2Vec の学習用データは日本語版 Wikipedia のデータを用いた．Doc2Vec のパラメータは以下の通りである．

表 1: Doc2Vec のパラメータ

| | |
|---|---|
| 次元数 | 100 |
| window size | 5 |
| min count | 5 |
| 手法 | DBoW |

### 6.1 実験結果

以下に Steve Jobs のスピーチの要約結果とニュース記事の要約結果の一例についてまとめる．要約結果は以下のようになった．

- Steve Jobs の要約文

  1. 1 年前に私たちの最高の創造物であるマッキントッシュを出したばかりで私は 30 歳になったばかりだった．

  2. 6ヵ月後私はそれに価値が見出せなかった．

  3. 私はリード大学を 6ヵ月で中退したが，更に 1 年半ほど後に完全に辞めるまで，もぐりの学生として大学に顔を出していた．

- ニュース記事の要約文

  1. ふたつめは、党内での不満が高まり、抑えが効かなくなっての退陣です。

  2. 皮肉なことに、自民党の脆弱化は、民進党が想定以上に国民から見放され凋落してきたことにも原因があると思われます。

  3. 現実的には、党内派閥の力関係の変化で安倍内閣は終わるのかもしれませんが、残念ながら党内派閥は、国民が選挙で選択できません。

分散表現を用いた場合，Steve Jobs の要約結果は先の実験の要約結果よりも重要文抽出の精度が悪いことがみてとれる．これは 分散表現を用いて作ったために，重要文ではない意味的に近い文を重要文としてとらえてしまったことが原因と考えられる．テスト文書の Steve Jobs のスピーチ内で文の意味が近いと考えられる彼の生い立ちについての記述が複数含まれており，実際の要約結果もこの部分が重要文として抽出されていることが見てとれる．

一方でニュース記事については先の実験の要約よりは，正解文である livedoor の要約に近いことが見てとれる．また他のニュース記事においても 10 件中 7 件である程度の性能の向上が見られた．

これらのことから，Doc2Vec の分散表現を用いることで文章が比較的短く，単語の重複の少ない文章からでもある程度は重要文が抽出できることが分かった．

## 7 まとめと今後の課題

本実験では抽出型要約のベースラインである LexRank の実装と，重要文抽出の性能評価をした．実験から TF-IDF によるベクトルを用いた LexRank はスピーチのように同じ単語や同じフレーズが繰り返し出現しやすい文からは重要文が抽出できるが，ニュース記事のようにある程度短文でまとまった，単語の重複の少ない文からは重要文の抽出ができない場合があることが分かった．また追加実験から Doc2Vec の分散表現ベクトルを用いることでニュース記事のような短文においてもある程度は重要文を抽出できることが分かった．このことから，LexRank による重要文抽出は，文章に応じた文のベクトル表現を用いることが重要だと分かった．

今後の課題としては抽出型要約の自動評価手法である ROUGE の実装や，他の抽出型要約手法の実装，長文においても分散表現を利用できるような手法の考案，抽出した重要文の整形による字数制限付き要約手法の考案などが挙げられる．

## 参考文献


[1] Erkan, Gunes, and Dragomir R. Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. Journal of Artificial Intelligence Research 22, pp.457-479. 2004

[2] Page, Lawrence, et al. The PageRank citation ranking. Bringing order to the web. Stanford InfoLab. 1999.

[3] Quoc V, Le and Tomas Mikolov. Distributed Representations of Sentences and Documents, In Proceedings of The 31st International Conference on Machine Learning Vol.14. pp.1188-1196. 2014.

[4] Tomas Mikolov, Greg Corrado, Kai Chen, and Jeffrey Dean. Efficient Estimation of Word Representations in VectorSpace. In Proceedings of the International Conference on Learning Representations. arXiv preprint arXiv:1301.3781. 2013

[5] https://sites.google.com/site/himazu/steve-jobs-speech. アクセス日:2017/7/23

[6] livedoor http://news.livedoor.com/