Word2Vec を用いた形態素群から推測される関連語の抽出

北海道科学大学 工学部情報工学科 4 年 ○澤野太一
北海道科学大学 大江亮介，川上敬

要　旨

情報科学分野の質問に対する応答システムを実現させるための基礎研究を行う．手法としては，形態素の分散表現を算出することが可能な Word2Vec を用いる．本研究で用いる Word2Vec のモデルは，インターネット上で公開されている Qiita と Wikipedia の本文を学習させた．本稿では，複数の形態素の分散表現から，それらの形態素と関連性が高い単語群から想定した出力が得られるかについて検証を行う．

1. 緒　　言

第一次 AI ブーム, 第二次 AI ブームの流行と衰退を経て, 2006 年に提唱された Deep Learning により，第三次 AI ブームが到来した[1]．この流行の背景には，以前の AI ブームが到来した際よりも, ネットワークの規模が拡大し, 且つ, データ集合を処理する技術の進歩により，種類や形式が多様な非構造化データまたは非定型的データであるビッグデータを獲得・利用できる環境が誕生したという経緯が存在する．

現在訪れている第三次 AI ブームに際し, プログラミングの学習を経て，人工知能の構築に着手せんとする初学者の出現が予測される．しかし，初学者が人工知能を構築するにあたり, どの言語を用い, どのような手法を利用すれば, 自身の想定している人工知能を適切に創造することが出来るかを推定することは，多くの初学者にとって至難を極めると予測される．また，初学者が学習を進めるにあたり，様々なインターネット上の文献や技術書を参考にすると思われるが，それらに記載されている情報が事実とは異なっている，または，既にそれに代わる最新の技術や手法が提案されているといった場合がある．さらに，サーチエンジンを用い，情報を取得しようとした場合には，検索対象となる単語が不適切だったことが原因で適切な検索結果を得ることが出来ない，または，目的の情報を取得するまでに労力を要するといった学習コストを上げてしまう要因が存在する．

こうした学習機会の損失が発生すると，情報科学分野への入門を諦めてしまう人物が増え，情報科学の発展が妨げられてしまうといった事態が発生しかねない．そうした事態を防止するためには，　情報を欲している人物に対して，最適な情報を最短で提供するシステムが必要になる．

そこで本研究では，学習者に対して最適と考えられる情報を短時間で提供するシステムを提案する．本稿では，前記のシステムを構築するにあたっての基礎研究として，Word2Vec を用い，入力した形態素の分散表現から想定した出力が得られるかについて検証した．

2. Word2Vec について

本研究では，形態素の分散表現を求めるための手法として，Google 社が提案した Word2Vec を用いる．

Word2Vec には，Continuous Bag-of-Words（以降，CBOW と記述する）と Skip-Gram の二つのモデルが存在する．

下記では，それぞれのモデルの構成や入出力について説明していく．

2.1 CBOW

CBOW[2][3]とは，入力層・中間層・出力層の三層からなるニューラルネットワークであり，図 1 に示すような構造となる．入力層は注目語 w_i の前後 M 単語からなる単語列 $Words = w_{i-M} \ldots w_{i-2}\, w_{i-1}\, w_{i+1}\, w_{i+2} \ldots w_{i+M}$ となる．中間層では，Word2Vec のパラメータ引数である cbow_mean の値によって，入力層の全単語のベクトルを合計したものを出力するか，全単語のベクトルの平均を出力するかを選択できる．出力層は，中間層の値から注目語となる w_i に対する予測語の確立分布を出力する．

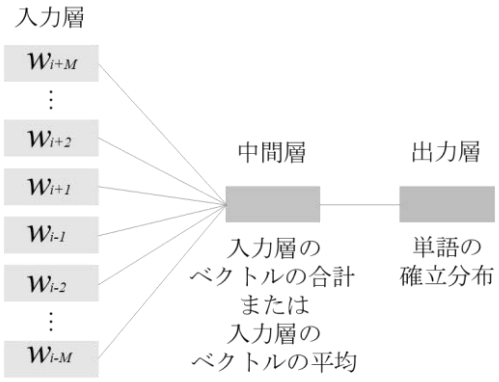

図 1　CBOW の構成

2.2 Skip-Gram

Skip-Gram[2][3]とは，入力層・中間層・出力層の三層からなるニューラルネットワークであり，図 2 に示すような構造となる．入力層には注目語 w_i が与えられる．中間層は重み行列となっていて, w_i の one-hot ベクトルと重み行列をかけることで注目語 w_i の単語ベクトルを出力する．出力層では，中間層で求めた単語ベクトルに出力層の重み行列をかけることで得られた内積を，soft-max 関数に通すことで，w_i と単語列 $Words = w_{i-M} \ldots w_{i-2}\, w_{i-1}\, w_{i+1}\, w_{i+2} \ldots w_{i+M}$ のそれぞれの単語との相関関係を表す確率を出力している．

表 1：コーパス毎のベクトルの和と積の順位

	入力した文章	想定した単語	Wikipedia の和の順位	Wikipedia の積の順位	Qiita の和の順位	Qiita の積の順位
1	文法が平易で可読性が高い汎用の高水準言語	Python	17 位	17 位	244 位	712 位
2	汎用なオブジェクト指向言語	Java	220 位	272 位	247 位	ランク外
3	おすすめの関数型言語	Haskell	6 位	ランク外	3 位	39 位
4	サーバ構築用のプログラミング言語	PHP	146 位	ランク外	397 位	711 位
5	スパコンを構築するにあたって用いられる OS	Linux	23 位	107 位	14 位	21 位
6	スーパーコンピューターで用いられる OS	Linux	32 位	349 位	78 位	322 位

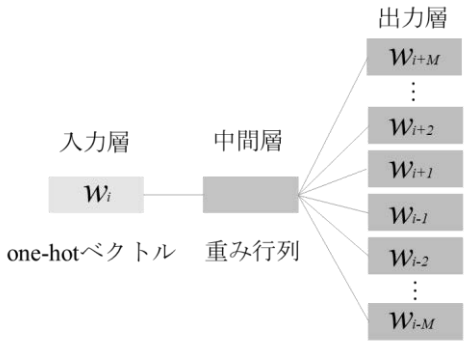

図 2　Skip-Gram の構成

3.　形態素群からの目的語抽出の検証実験

3.1　Word2Vec のパラメータ調整

入力となる形態素群から，想定した単語の出力が得られるか否かについて検証実験を行った．今回の検証で設定した Wrod2Vec のパラメータは，次元数=250，window(参照する前後の単語数)=20, min_count(登場数が n 回未満の単語を削除)=10，sg(モデルの選択．今回は CBOW を選択)=0 とした．また，Word2Vec に学習させるコーパスとして，本稿では Qiita と Wikipedia の本文を用いた．

モデルを選択するにあたって，Word2Vec のパラメータは上記に記したものを用い，CBOW と Skip-Gram のモデルを作成しどちらのモデルを用いた方が想定した結果が得られるかを検証した．結果として，CBOW を用いて作成したモデルの方が，Skip-Gram で作成したモデルに比べ，専門性が高い・低いにかかわらず登場する語(本稿ではこれを汎用語と定義する)を排除できるため，情報科学分野の質問に焦点を当てた本稿では CBOW を選択する．

3.2　実験手順

実験手順として，まず，入力された全ての形態素の中で名詞と判定されたものについて，全名詞との間で分散表現の類似度を計算する．次に，全ての入力名詞に対する類似度に関して，単語ごとの類似度の和を取ったものと，積を取ったものを求める．これらの和と積を大きい順に並べ，各上位 1000 個の単語を出力の候補とする．この時の積は，値の差が明確になるよう，数値を形態素の数の 10 乗としている．また，ベクトルの積を取るということを考慮し，類似度の値が負になった単語は実験データとして用いないこととにする．

3.3　コーパスの選択

Qiita と Wikipedia の内，どちらのコーパスを用いた方が想定とした結果を得られるかを検証した．表 1 の 3 と 4 の結果に注目すると，入力した文章に抽象語や略語が含まれていると，Wikipedia よりも Qitta の本文をコーパスとしたモデルが想定した出力を得られやすいことが判明した．逆に，入力した文章に専門語が多く含まれている場合は，Wikipedia の本文をコーパスとしたモデルが，想定語が上位に出現する．したがって，コーパスの選択は入力文に応じて柔軟に変更すべきである．

3.4　ベクトルの和と積の比較

ベクトルの和と積で単語のランクを決定する際，どちらの方法を用いた方が，想定語が上位に登場するかを検証した．結果を示した表 1 から読み取れるように，ベクトルの積を取る方法よりも，和を取る方法を用いた方が、想定語が上位に出現するという結果になったため，ベクトルの和を取るべきであると結論付けられた．

4.　結　　言

本研究では，入力された文章に形態素解析を行うことで得られる形態素群に対し，Word2Vec を用いることによって得られる分散表現から，入力文への応答となる単語の出力を試みた．

出力の候補となる単語群 1000 個の中に，文章の入力者が想定していた単語は出現している場合も存在したが，汎用語が想定した単語よりも上位に出現するという結果になったため，本システムはまだ実用には至らないという結論に至った．

今後の課題としては，分散表現として出力される汎用語のベクトルの数値を如何にして下げるか，という問題に主眼が置かれる．

参　考　文　献

1)　総務省 ：人工知能(AI)の現状と未来
http://www.soumu.go.jp/johotsusintokei/whitepaper/ja/h28/html/nc142120.html

2)　Tomas Mikolov，Kai Chen，Greg Corrado，Jeffrey Dean
Efficient Estimation of Word Representations in Vector Space.
In *ICLR*，4 page，2013

3)　David Meyer
How exactly does word2vec work?
pp 2-17，2016