平成 28 年度　学士論文

# ハンガリアン法による語の相似関係抽出

Analogous word association and its detection by the Hungarian method.

平田　泰樹

Taiki Hirata

北海道大学 工学部 情報エレクトロニクス学科 コンピュータサイエンスコース 知識ベース研究室

Department of Electronics and Information Engineering.
Hokkaido University

2017 年 2 月 10 日

# 目次

# 図目次

# 表目次

# 第 1 章

# 序論

## 1.1 はじめに

SNS などのサービスが普及し、以前にも増して世界中の人々がインターネット上にテキストデータを投稿するようになった。こうして集められるデータは、その人の趣向、動向、人と為りや、世界の動向を知る手掛かりとなるだろう。しかし、インターネット上にあるテキストデータは膨大なものになっており、ユーザーが本当に欲しい情報は見えにくく、扱いづらいものとなってしまっている。膨大なデータを有効活用していくためには、効率的な分類、加工をしていかなければならないが、データのすべてを人手で解析し、まとめることは非常に困難である。そこで、計算機を利用してこのテキストデータを処理することがさまざまな分野で考えられている。
テキストデータを計算機で処理するにあたって、テキストを何らかの方法で数値化する必要がある。そもそもテキストを構成する文字自体は単なる記号であり、また、計算機が文字データを識別するために、文字一つ一つに事前に割り当てられている数値には、記号識別以上の意味がない。
そこで、テキストを計算機で処理するために、いかにして意味を持つ数値で表現し、どのように利用するか、が課題となっている。

## 1.2 研究背景

テキストを、意味を持つ数値で表現する手法について、現在まででいくつかの方法が研究されている。中でも 2013 年に発表された word2vec は、ニューラルネットワークによって学習した単語の意味表現ベクトルで、単語の和、差の計算ができるようになり話題となった。

このベクトルの加算減算が可能になったこと、形態素ベクトルの位置関係に相似性が見られることは、主成分分析、t-SNE などで次元を削減し、可視化したデータから見て取れる。

## 1.3　本研究の目的

本研究では、word2vec により得たベクトルデータを可視化することなしに、語の相似関係を抽出することを目的としている。

## 1.4　本論文の構成

はじめに本章は序論であり、研究背景に関して述べた。
本稿２章では、本研究で用いる単語の意味表現についての説明を述べた。３章では、本研究で用いるベクトルデータを出力する word2vec について述べた。
４章では、word2vec により得られたデータの解析に用いる、梅山氏のグラフノードマッチング問題に関する提案手法と、ハンガリアン法について述べた。
５章で、本研究で行った実験の流れについて説明し、結果を検証した。
最後に６章では、本研究における結論と、今後の展望について考慮していることを述べて最後のまとめとした。

# 第 2 章

# 意味表現の学習

本研究では、word2vec で学習した分散表現ベクトルから語の相似関係を抽出することを目的としている。本章では、単語の意味表現の学習についていくつかの表現方法と、word2vec で取得可能な分散表現ベクトルについてを説明する。

## 2.1 形態素と意味表現

### 2.1.1 形態素

自然言語は、人と人とのコミュニケーションに用いられる道具であり、文字、あるいは音の並びによって構成されている。しかし、一般に文字、或いは音そのものは意味を持たないため、自然言語処理の場面では、最小単位として**形態素**を用いることが多い。
形態素とは、文字列が意味、或いは役割を持つ最小のまとまりである。本研究ではこの形態素に着目していく。

### 2.1.2 形態素解析

先に述べた通り、多くの自然言語処理に関わるタスクでは、形態素を最小単位としているが、日本語などの言語は特に、形態素ごとに区切られた文章ではなく、また、計算機は、形態素のまとまりを認識できない。そこで、自然言語処理をする際に、前処理として**形態素解析**を行うのが一般的である。形態素解析をすることで、入力された文を形態素毎に区切り、品詞などを決定する。

本研究では、京都大学情報学研究科と、日本電信電話株式会社コミュニケーション科学基礎研究所が、共同研究ユニットプロジェクトで開発した **MeCab**[*1] [3] という、オープンソースの形態素解析エンジンを実験データの前処理に用いた。

[*1] http://taku910.github.io/mecab/

### 2.1.3　意味表現

自然言語を計算機で処理するにあたって、計算機が形態素の意味や役割を獲得すれば文や文章など、より複雑な構造を持ったテキストの意味を理解し、それに応じた処理ができるようになることが期待されている。この、計算機で認識するために形態素の意味や役割を何らかの数値で表現したもの、或いは表現することを意味表現という。
この意味表現をどのように与えるかということが問題になる。

## 2.2　辞書

我々は、未知語の意味を知ろうとする時に検索エンジンを利用して意味を調べるなど、何らかの方法で意味が記述されたテキストを参照する。その参照対象をここでは一括して辞書と呼ぶこととする。辞書を計算機に与えることで、意味を教えることはできるかもしれない。日本語 **WordNet**[*2]は、計算機で使うことを想定して開発された辞書である。

### 2.2.1　WordNet

日本語版 WordNet は国立研究開発法人情報通信研究機構（NICT）で 2006 年から 2012 年まで開発されていた大規模な日本語意味辞書である。プリンストン大学で開発された Princeton WordNet と、ヨーロッパの EuroWordNet 協会が推進する Global WordNet Grid に着想を得て開発された。
WordNet では、単語の意味を **synset** と呼ばれるグループで表現しており、各 synset は単語の意味に関する記述と、上位関係や下位関係など、他の synset との関係まで保持しており、synset の情報を用いて単語同士の類似度なども計算できるようになっている。なお、現在の日本語 WordNet の情報規模は以下の通りである。[4]

- synset 数 57,238
- 単語数 93,834
- 語義数：synset と単語のペア 158,058
- 定義文数 135,692
- 例文数 48,276

人手で整備された辞書は単語の意味に関してある程度精密な情報を備えているが、新語や多くの複合語には対応しきれておらず、たとえば、MeCab を用いて日本語版 Wikipedia 全文から抽出された形態素数は 63〜84 万程度であった（数字の幅があることに関しては後ほど説明するが、形態素の基本形で数えた場合と、活用された形そのままの場合の 2 パターンで実験をしたからである。）のに対し、WordNet により意味を保持できている単語は 10 万程度であるから、５〜７倍の単語については意味を理解できないということになる。こうした点から、単語・複合語のすべてとすべての意味を収

[*2] http://nlpwww.nict.go.jp/wn-ja/

録し、あらゆる言語処理の応用に耐えうる辞書を作成することは、非常に困難であることがわかる。

## 2.3　統計的手法

辞書などを作成することで、計算機に単語の意味を明示的に与えることには限界があることがわかった。そこで、現在数多くある、人間が生み出したテキストデータから計算機で自動的に単語の意味を学習・獲得することを考える。以下では、テキストデータを統計的に処理することで獲得する意味表現について述べていく。

### 2.3.1　分布的意味表現

統計的に獲得する意味表現の一つに**分布的意味表現**というものがある。
これは、**分布仮説**

> The Distributional Hypothesis is that words that occur in the same contexts tend to have similar meanings(Harris, 1954). (ACL wiki)

”同じ文脈に出現する単語は、似た意味を持つ。”という考え方によって構築されるものである。[1]

図 2.1: 分布仮設による意味の類推例

上述の例は全く同じ文脈で、似たような料理名が出てくる例文を並べたものである。この時、もしどれかひとつの料理名が全く無知なものであったとしても、それが料理であること、どのような料理であるか、ある程度類推することができるだろう。それは、まさしく同じ文脈に出現する単語であるからに違いない。

この考えに則ってベクトルを作成してみる。例えば、

- 今日の晩御飯は、ジャガイモの入ったカレーライスだ。
- 今日の晩御飯はハヤシライスだ。

という二本の文章を例にして、”カレーライス”と”ハヤシライス”の意味表現を考える。

まず、それぞれの文章を形態素解析すると、

- 今日 \ の \ 晩御飯 \ は \、\ ジャガイモ \ の \ 入っ \ た \ カレーライス \ だ \。
- 今日 \ の \ 晩御飯 \ は \ ハヤシライス \ だ \。

となる（上では \ によって形態素を区切っている）。

これをもとに、”カレーライス”と”ハヤシライス”に関して、同じ文脈に着目した意味表現ベクトルを作成する。同じ文脈に出現する形態素（これ以降文脈語と呼ぶ）をベクトルの各次元とし、注目する単語と何個の同じ文脈でその次元の文脈語が出現しているかを値としてとると、

- カレーライス = [(今日：1),(の：1),(晩御飯：1),(は：1),(、：1),(ジャガイモ：1),(入っ：1),(た：1),(だ：1),(。：1)]
- ハヤシライス = [(今日：1),(の：1),(晩御飯：1),(は：1),(、：0),(ジャガイモ：0),(入っ：0),(た：0),(だ：1),(。：1)]

と表すことができる。

ここで、ある単語 $x_i$ と別な単語 $x_j$ が何らかの文脈で同時に出現していることを**共起**していると言い、何度共起しているかの回数を**共起頻度**と言う。つまり、上記の例で作成したベクトルは、共起頻度を値として持つ**共起ベクトル**であると言える。[5] この方法でベクトルを作成する場合は、学習に用いられるテキストの集まり (**コーパス**) すべてから学習するため、一般に高次元で疎なベクトルが生成される。これは、コーパスに含まれる単語数が数万程度であったり、単語数に対し、単語同士が共起することは一般に多くないためである。

こうして作成される意味表現ベクトルを用いて別な自然言語処理のタスクを行う際、元々のベクトルが高次元で疎なものであるために、学習事例を正しく表現できなくなってしまうなどの問題が発生してしまう。

### 2.3.2　分散的意味表現

分布的意味表現ベクトルが高次元で疎なものであるという点を避けた学習方法として提案されたのが、**分散的意味表現**である。

分散的意味表現の学習手法の基本は以下のようになっている。

1. まず、すべての単語に、任意の固定次元のベクトルを割当て、ランダムな実数値で初期化する。
2. 与えられたベクトルを用いて、何らかの予測タスクを解く。
3. 予測タスクを解いた結果に応じてベクトルの値を更新する。
4. 2 へ戻る。

分散的意味表現を表すベクトルの各次元は、何らかの実数値変数が持つ値と解釈することができる。この実数値変数が持つ特徴はどのような予測タスクを解くか、固定次元数をどのくらいの値にす

るかなどによって変わると予想されるが、単純な共起頻度が値となるわけではなく、また、他の様々な形態素と次元を共有することから、コーパス中で共起することがない形態素同士の関係についても表現しているものと期待されている。

次章では、本研究で用いた形態素の分散的意味表現学習ツール word2vec で実装されている 2 種類の予測タスクについて述べる。

# 第 3 章

# word2vec

**word2vec**[*1] とは、2013 年に Google の Mikolov らが発表した、単語の分散的意味表現学習ツールである。意味表現学習の過程で解くのは、ある文脈内で共起する形態素を予測するタスクである。文脈が与えられ、次に出てくる形態素を予測するモデルを**言語モデル**と言い、これは、与えられた形態素列がどれほどその言語らしいかを評価するモデルである。

言語モデルでは着目形態素の前方にある形態素のみから次に来る形態素を予測しなければならないが、意味表現獲得を目的とする場合、後方の形態素も予測に使うことができる。よって、$x_1,...,x_{i-1}$ と、$x_{i+1},...,x_n$ から $x_i$ を予測する。

この章では word2vec に実装されている 2 つの手法を紹介していく。

## 3.1 実装アルゴリズム

それぞれの手法を紹介していく前に事前に確認しておくが、word2vec では、単語の予測タスクを解く上で、一つの単語に対し、二つのベクトルを設定し、学習を進めている。以下に述べる二つの手法の双方で、同一文脈における単語の共起を、意味表現ベクトルの内積で定義しているためである。

通常同一単語が同一文脈に出現することは稀であり、$x_i$ に着目した時、$x_i$ の文脈で $x_i$ が出現する確率 $p(x_i|x_i)$ を小さくする必要があるが、この確率を導出する計算過程で $x_i^T x_i$ を計算することになる。そして、$p(x_i|x_i)$ を小さくするために、$x_i^T x_i$ を小さくしなければならない。しかし、この後に述べるが $p(x_i|x_i)$ の導出にはソフトマックス関数という関数を利用しており、これはスケール不変な関数で、$x_i^T x_i$ を小さくしても値が変化しない。

こうした問題を避けるために、以下の手法中では、用いる意味表現ベクトルを、同じ単語が対象語として現れるか、対象語と共起する文脈語として現れるかによって使い分けている。

### 3.1.1 連続単語袋詰モデル

**連続単語袋詰モデル** (Continuous Bag-of-Words model, CBoW model) では、**文脈語** (着目形態素と共起している形態素) から着目形態素の出現確率を予測している。

[*1] https://github.com/svn2github/word2vec

図 3.1: CBoW によるベクトル構築イメージ

着目形態素の前後 n（下図では 3）個の形態素を文脈語とし、文脈語の表現ベクトルの平均を、着目形態素 $x_i$ の文脈を代表するベクトル $\hat{x}$ として、与えられた文脈中に、着目形態素 $x_i$ が出現する確率 $p(x_i|\hat{x})$ は、

$$p(x_i|\widehat{x}) = \frac{\exp(\hat{x}^T x)}{\sum_{x' \in v} \exp(\hat{x}^T x')} \tag{3.1}$$

と表せる。ここで $v$ はコーパス中の全形態素からなる語彙集合であり、x' は $v$ 中の単語である。
文脈語の代表ベクトル算出時に語順を無視しているため、同じく語順を無視して文や文書ベクトルを導出する **Bag-of-Words モデル**の拡張とみなすことができる。

式 (3.1) で導出できる確率を学習していく。学習過程については、次のモデルで学習する確率を説明した後に述べる。

### 3.1.2　連続スキップグラムモデル

**連続スキップグラムモデル** (continuous Skip-gram model, Sg model) では、連続単語袋詰モデルとは逆に、与えられた文脈語から着目形態素の出現を予測するものとなっている。

図 3.2: Sg によるベクトル構築イメージ

着目形態素 $x_i$ から文脈語を予測する場合の確率 $p(z_{i-n},...,z_{i-1},_{i+1},...,z_{i+n}|x_i)$ は、着目形態素 $x_i$ が与えられた時、文脈語 $z_{i-n},...,z_{i-1},z_{i+1},...,z_{i+n}$ の出現確立がすべて独立であると仮定して、

$$p(z|x_i) = \frac{\exp(x_i^T z)}{\sum_{z' \in v(x)} \exp(x^T z')} \tag{3.2}$$

と表せる。ここで $v(x)$ はコーパス中で着目形態素 $x_i$ と共起している文脈語の集合であり、z は文脈語ベクトルである。
文脈語の独立性を仮定することで、

$$p(z_{i-n},...,z_{i-1},z_{i+1},...,z_{i+n}|x_i) = p(z_{i-n}|x_i)...p(z_{i-1}|x_i)p(z_{i+1}|x_i)...p(z_{i+n}|x_i)$$

と表すことができ、スキップグラムモデルでの確率計算を簡単にしている。

### 3.1.3　二つの学習モデルの違い

連続スキップグラムモデルでは、一つの着目形態素と一つの文脈語の関係から確率計算を行うので、少ないコーパスからでも一定の精度のモデルを学習できる。これに対し連続単語袋詰めモデルでは複数の文脈語から一つの着目形態素を予測するモデルであるため、複数の文脈語からなる単語列が、コーパス中に複数回出現している必要がある。したがって、連続スキップグラムモデルよりも大きなコーパスデータが必要となる。[5]

## 3.2　モデルの最適化

本節ではいよいよ、前節で導出した確率の最適化について述べていく。
まず、式 (3.1),(3.2) の右辺は**ソフトマックス関数** (softmax function) と呼ばれる関数の形になっており、このソフトマックス関数について述べる。

### 3.2.1　ソフトマックス関数

**ソフトマックス関数** (softmax function) は、N個の実数出力を、確率値に変換するのによく用いられる関数である。

$$softmax(x_i) = \frac{\exp(x_i)}{\sum_{j=1}^{N} \exp(x_j)} \tag{3.3}$$

ソフトマックス関数が持つ特徴として、

- 出力値が 0 から 1 の範囲内
- $\Sigma_{i=1}^{N} softmax(x_i) = 1$ を満たす

というものがあり、これによって出力値を確率値として解釈可能にする。

### 3.2.2　学習の流れ

式 (3.1),(3.2) はどちらも exp 内に内積を含んでいるため、対数をとれば双線形になる、対数双線形な関数である。両式は一方の変数を固定させるともう一方に関して凸になる関数であるため、交互に最適化を行うことができる。

対数を取った確率を、それぞれの変数に関して偏微分し、学習率をかけたものを足すことで、ベクトルを更新していく。

$$
\begin{aligned}
x^{(t+1)} &= x^{(t)} + \eta^{(t)}\left(\frac{\partial}{\partial x}\log(p(z_i|x_i))\right) \\
&= x^{(t)} + \eta^{(t)}\left(z^{(t)} - \sum_{z' \in V(x)} z' p(z'|x)\right) \qquad (3.4) \\
z^{(t+1)} &= z^{(t)} + \eta^{(t)}\left(\frac{\partial}{\partial z}\log(p(z|x))\right) \\
&= z^{(t)} + \eta^{(t)} x(1 - p(z|x)) \qquad (3.5) \\
& \qquad (3.6)
\end{aligned}
$$

# 第 4 章

# 梅山氏の提案手法

ノード数が等しい二つのグラフG，Hを比較した際、両者の違いが最小となるようにノードを対応させる問題を、**ノードマッチング問題**という。本研究では word2vec の出力ベクトル空間上にある任意の二つの形態素集合をそれぞれ、各形態素をノードとみなした重み付き無向グラフと考えて、梅山伸二氏の提案手法 [8] でノードのマッチング問題を解くことで相似関係を抽出する。

本章では、梅山氏の提案手法を紹介する。[8]

## 4.1 グラフの同型性問題

ノード数が等しい二つの重み付き無向グラフG，Hの隣接行列をそれぞれ $A_G$、$A_H$ とし、行・列の並び替えを行う転置行列を $P$ と表す時、両グラフの違いの大きさを下記の式で定義することができる。

$$J(P) = ||PA_GP^T - A_H||^2 \tag{4.1}$$

$J(P)$ の値が最も小さくなる時のノードマッチングが最適なマッチングであり、これを満たす $P$ を求める。

### 4.1.1　スペクトル分解

梅山氏の提案手法で無向グラフのノードマッチングを解くに当たり、まず二つのグラフの隣接行列を固有値分解する。
ここからは、元論文に掲載されていた例と同一のものである、下記の二つのグラフを用いて、操作ステップを確認していく。

図 4.1: グラフG、グラフH

図のグラフから隣接行列 $A_G$、$A_H$ は以下のようになる。

$$A_G = \begin{pmatrix} 0.0 & 5.0 & 8.0 & 6.0 \\ 5.0 & 0.0 & 5.0 & 1.0 \\ 8.0 & 5.0 & 0.0 & 2.0 \\ 6.0 & 1.0 & 2.0 & 0.0 \end{pmatrix}$$

$$A_H = \begin{pmatrix} 0.0 & 1.0 & 8.0 & 4.0 \\ 1.0 & 0.0 & 5.0 & 2.0 \\ 8.0 & 5.0 & 0.0 & 5.0 \\ 4.0 & 2.0 & 5.0 & 0.0 \end{pmatrix}$$

得られた行列 $A_G$、$A_H$ をそれぞれ固有値分解する。この時、固有ベクトルを、対応する固有値の大きさ順に並べたモード行列を $U_G$、$U_H$ とし、固有値を大きい順に対角上に並べた対角行列を $\Lambda_G$、$\Lambda_H$

とすると以下の式のように表せる。

$$A_G = U_G \Lambda_G U_G^T \tag{4.2}$$
$$\Lambda_G = diag(14.25, -0.28, -4.83, -9.14)$$
$$U_G = \begin{pmatrix} 0.614 & 0.141 & -0.182 & -0.755 \\ 0.434 & -0.528 & 0.726 & 0.079 \\ 0.548 & -0.270 & -0.582 & 0.536 \\ 0.366 & 0.793 & 0.371 & 0.369 \end{pmatrix}$$
$$A_H = U_H \Lambda_H U_H^T \tag{4.3}$$
$$\Lambda_H = diag(13.26, -0.77, -3.43, -9.05)$$
$$U_H = \begin{pmatrix} 0.538 & -0.436 & -0.425 & -0.583 \\ 0.344 & 0.880 & -0.018 & -0.327 \\ 0.624 & 0.026 & -0.250 & 0.740 \\ 0.450 & 0.187 & 0.870 & 0.079 \end{pmatrix}$$

$U_G$、$U_H$ のすべての要素の絶対値を取ると、行に各ノード、列に各固有値ベクトルが対応させることができる。

$$\overline{U}_G = \begin{pmatrix} 0.614 & 0.141 & 0.182 & 0.755 \\ 0.434 & 0.528 & 0.726 & 0.079 \\ 0.548 & 0.270 & 0.582 & 0.536 \\ 0.366 & 0.793 & 0.371 & 0.369 \end{pmatrix}$$
$$\overline{U}_H = \begin{pmatrix} 0.538 & 0.436 & 0.425 & 0.583 \\ 0.344 & 0.880 & 0.018 & 0.327 \\ 0.624 & 0.026 & 0.250 & 0.740 \\ 0.450 & 0.187 & 0.870 & 0.079 \end{pmatrix}$$

行に各ノードというのはつまり、ノード数４の重み付き無向グラフを、隣接行列の固有値分解により４次元空間上で重みを距離に反映した四つの頂点に変換したことになる。

$A_G$ と $A_H$ が同型であると仮定すると、式 (4.1) の左辺が 0 になる。よって、

$$PA_G P^T = A_H \tag{4.4}$$

を満たす $P$ が存在する。式 (4.4)、(4.2)、(4.3) より、

$$PU_G \Lambda_G U_G^T P^T = U_H \Lambda_h U_H^T$$

再度、$A_G$、$A_H$ が同型であるという仮定から

$$PU_G = U_H S$$

を満たす行列 $S$ が存在し、

$$P = U_H S U_G^T$$

と表せる。

求めたい $P$ は順列行列であるから、$\hat{P}$ を導入し、

$$\hat{P} = \begin{matrix} \\ i \end{matrix} \begin{matrix} \\ \end{matrix} \overset{\pi(i)}{\begin{pmatrix} & \vdots & \\ \cdots & 1 & \end{pmatrix}}$$

対角行列を $\hat{S}$ とし、$U_H = [h_{ij}]$、$U_G = [g_{ij}]$、$\hat{S} = diag(s_i)$ であることを用いて

$$tr(\hat{P}^T U_H \hat{S} U_G^T) = \sum_{i=1}^{n} \sum_{j=1}^{n} s_j h_{ij} g_{\pi(i)j}$$

ここで、

$$\begin{aligned}\sum_{j=1}^{n} s_j h_{ij} g_{\pi(i)j} &\leq |\sum_{j=1}^{n} s_j h_{ij} g_{\pi(i)j} \\ &\leq \sum_{j=1}^{n} |s_j h_{ij} g_{\pi(i)j} \\ &= \sum_{j=1}^{n} |h_{ij}||g_{\pi(i)j}\end{aligned}$$

よって、

$$\begin{aligned}tr(\hat{P}^T U_H \hat{s} U_G^T) &\leq \sum_{i=1}^{n} \sum_{j=1}^{n} |h_{ij}||g_{\pi(i)j}| \\ &= tr(\hat{P}\overline{U}_H \overline{U}_G^T)\end{aligned}$$

モード行列 $U_G$、$U_H$ は長さ 1 の固有ベクトルでできた行列であり、各要素の絶対値を取った $\overline{U}_G$、$\overline{U}_H$ の行ベクトルも長さは 1 であるから、$\overline{U}_G \overline{U}_H = [x_{ij}]$ とした時、$0 \leq x_{ij} \leq 1$ を満たす。

以上より、任意の順列行列 $P$ は、

$$tr(P^T \overline{U}_H \overline{U}_G^T) \leq n$$

を満たし、$P$ が最適マッチングを導く時、$n$ が最大値となるので、$\overline{U}_H \overline{U}_G^T$ に、ハンガリアン法を用いることで、最適な割り当てを求めている。

参考までに、ハンガリアン法を用いて求められた $P$ は、

$$P = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

### 4.1.2　ハンガリアン法

**ハンガリアン法** (Hungarian method) とは、割当問題の解法として頻繁に用いられるものの一つであり、n 次の正方行列が与えられた時、それぞれの行、列に重複が無いように値を選び出し、最小となる組合せを見つけるといった問題を解くことができる。たとえば、「n 個の仕事を n 人の作業員にやらせる時のコストを考える。同じ仕事を複数人の作業員でこなすことは無く、また、同じ作業員が複数の仕事をこなすこともないことを条件とし、仕事 j を作業員 i がやる時のコストを $c_{ij}$ としたコスト行列 C を入力として、コストが最小になる割り当てを見つける」

グラフのノードマッチングを解く際は、最大となる組合せを見つけることになるが、予め行列の全要素を-1 倍してから最小となる組合せを見つけることにし、最大化問題を最小化問題に置き換えている。

ハンガリアン法は、以下の 6 ステップにより構成される。

1. 各行の最小値を、その行の全要素から引く。
2. 各列の最小値を、その列の全要素から引く。
3. 行列内のすべての 0 を通るように必要最小限数の直線を引き、直線状の要素に印をつける。
4. 直線の本数 $= n$ であれば、値が 0 の要素が最適マッチングの位置であり、終了。
5. 直線の本数 $< n$ であれば、印のないすべての要素の最小値を、印がないすべての要素から引き、印が二つついている要素には同じ値を加算する。
6. 4 に戻る。

作業員に割り当てる際、作業員毎のコストの差が重要であり、差が同じであれば最適な割当ては変わらない点に注目している。

### 4.1.3　まとめ

梅山氏の提案手法で想定されている入力は、ノード数が一致しており、構造がほぼ同型なグラフであり、それ以外の、ノード数が一致しなかったり、グラフの構造が大きく異なっているもの同士のノードマッチングの精度は保証されないことに注意する必要がある。

# 第 5 章

# 実験と考察

本章では、実験の流れについて説明していく。

## 5.1 目的

本研究の目的は、word2vec の出力ベクトルを利用して語の相似関係を抽出することであり、この実験では、一対の単語ペアを入力とし、入力単語それぞれの近傍 n 個ずつの単語で作った二つのグループに梅山氏の提案手法を用いることでグループ内の単語同士の対応関係を出力する。

## 5.2 実験データ

用いた実験データについて説明する。

コーパスとして用いたのは、2016 年 9 月時点での日本語版 Wikipedia 最新記事[*1]であり、ここからテキストデータを抽出し、MeCab を用いて形態素毎に分かち書きした。

次に、word2vec を用いて分かち書きしたコーパスから形態素の意味表現ベクトルを学習した。ベクトル学習のハイパーパラメータは以下の通りである。

- 適用手法：CBoW
- 学習するベクトルの次元数：200
- 文脈窓：8
- 負例サンプリング数：25
- 階層型ソフトマックス：利用しない
- 学習の反復回数：15

文脈窓は、学習する際に使用する、対象語の前後の文脈語数で、今回文脈語は全部で 16 個使っている。

[*1] https://dumps.wikimedia.org/jawiki/20160901/jawiki-20160901-pages-articles.xml.bz2

## 5.3　梅山氏の手法の有効性確認

まず初めに、word2vec の出力ベクトルデータを元に、単語同士の相似関係を梅山氏のグラフノードマッチング手法を利用して抽出できるかどうかを確認した。

表 (5.1) にある、集合 X、集合 Y の二つの単語群を作成した。これらは、男という括りにできる単語集合 (X) と、女という括りにできる単語集合 (Y) であり、それぞれ対になる単語が含まれるように作成した（例：父-母、雄-雌）。

表 5.1: 手法の妥当性確認のためのテストデータ

| 集合 X | 集合 Y |
|---|---|
| 叔父 | 妹 |
| 王 | 祖母 |
| 老人 | 王女 |
| 父 | 雌 |
| 兄 | 老婆 |
| 祖父 | 花子 |
| 弟 | 姉 |
| 息子 | 叔母 |
| 雄 | 娘 |
| 太郎 | 母 |

表 5.2: テストデータのノードマッチング解析結果

| 集合 X | 集合 Y |
|---|---|
| 叔父 | 叔母 |
| 王 | 王女 |
| 老人 | 老婆 |
| 父 | 母 |
| 兄 | 姉 |
| 祖父 | 祖母 |
| 弟 | 妹 |
| 息子 | 娘 |
| 雄 | 雌 |
| 太郎 | 花子 |

これらの集合を元に、word2vec が出力したベクトルをからなる行列 $U_X$、$U_Y$ を作成した。

$$U_X = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} U_Y = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$$

$U_X$、$U_Y$ はそれぞれ、第 4 節で述べた $U_G$、$U_H$ に対応する。

集合Xの単語と、集合Yの単語は、それぞれのグループにおいて、二点間の距離が重みとして乗ったグラフとみて、$U_G$、$U_H$ の代わりに $U_X$、$U_Y$ を用いて、梅山氏の提案手法を適用した。この際、word2vec による出力ベクトル空間において、単語ベクトルの長さが持つ情報が不明瞭であるため、余弦類似度をコスト行列作成に用いた。

グループ X、Y に関して、梅山氏の提案手法によるノードマッチングを適用した結果が表 (5.2) の通りであるから、ある括りで作られた単語グループ同士では、ほぼ同型のグラフ構造が得られ、梅山氏のノードマッチングの適用で対応する単語同士を発見できることと、word2vec の出力ベクトルは単語の相似関係を保持できていることが予想できる。

word2vec の学習がうまくできていることの確認と、学習ベクトルに梅山氏が提案したノードマッチング問題の解法を適用することで相似関係が抽出できることの確認のために、2013 年に T.Mikolov らによって発表された論文に掲載されている下記のグラフ (5.3) で見られる関係の抽出確認を行った。

図 5.1: 連続 Skip-gram モデルで学習した 1000 次元の単語ベクトルを、主成分分析により 2 次元に縮約し、いくつかの国名と首都名をプロットしたもの。首都名などの関係情報を与えること無しに、モデルが自動的に単語の概念を暗黙的に学習したことが見て取れる。[2]

このグラフにある例を用いて、先ほどと同様に単語集合を作りノードマッチングを行った結果が以下の表 (5.3) である これにより、グラフ (5.3) で確認のできる対応関係と同等のものが抽出できてい

表 5.3: 国名と都市名のマッチング結果

| 集合 X | 集合 Y |
|---|---|
| 中国 | 北京 |
| ポーランド | ワルシャワ |
| ギリシャ | アテネ |
| ポルトガル | リスボン |
| ドイツ | ベルリン |
| スペイン | マドリード |
| フランス | パリ |
| ロシア | モスクワ |
| トルコ | アンカラ |
| 日本 | 東京 |
| イタリア | ローマ |

ることがわかった。

グラフ (5.3)[2] において、分散ベクトルの学習モデルは連続 Skip-gram モデルで 1000 次元のベク

トルを学習していたのに対し、本実験では学習モデルに連続 Bag-of-Words モデルを用いて 200 次元のベクトルを学習していた。ハイパーパラメーターやモデルに差があったが、今回の検証実験により、低次元ベクトルにおいても単語の関係性が反映されていると期待できるので、このまま検証を続けていく。

## 5.4　実験１：着目形態素の近傍形態素でノードマッチング

次に、入力された単語対それぞれの近傍単語から取得した単語で作成したグループ同士にノードマッチングを適用することで、単語同士の掃除関係が抽出できるかを実験した。

入力した単語対の近傍 10 個を取得したものが、表 () で、この表に対しノードマッチングを適用した結果が、表 () である。

ここで、近傍の取り方については、着目単語との余弦類似度が高いものを類似度の高さ順に取ってきている。

表 5.4: (王, 女王) の近傍 10 単語

| 集合 X | 集合 Y |
|---|---|
| は王 | 王女 |
| 皇帝 | 王妃 |
| に王 | 国王 |
| 国王 | エリザベス女王 |
| 君主 | ヴィクトリア |
| 大王 | 王 |
| 王妃 | 王子 |
| 伯 | エリザベス 1 世 |
| 聖王 | エリザベス 2 世 |
| 女王 | 王配 |

表 5.5: (王, 女王) の近傍 10 単語でグラフノードマッチング

| 集合 X | 集合 Y |
|---|---|
| は王 | 王 |
| 皇帝 | 国王 |
| 女王 | ヴィクトリア |
| **国王** | **エリザベス女王** |
| に王 | 王妃 |
| 王妃 | 王配 |
| 君主 | 王子 |
| **大王** | **王女** |
| 伯 | エリザベス 1 世 |
| **聖王**[a] | **エリザベス 2 世** |

[a] 徳が深い王

結果を見るに、そもそも形態素解析がうまくできていない部分があったことがわかる。加えて、(王, 女王) の関係にあるような対応関係を構築することが、人手でも難しい集合になってしまっていることがわかる。

他の例でもいくつか試してみた。

いくつかのケースを試してみたところ、一組の単語対の近傍単語同士で作ったマッチング 10 組のうち、与えた単語対の関係と等しいものは平均すると 2 組程度の取得にとどまった。

正答率の低さの原因として考えられることは、そもそも、与えた単語対の関係を作れる単語が各グループに揃わないことが原因として考えられた。そこから、取得する近傍単語の数を増やし、単語対を作る際のバリエーションを増やせるようにして、再度実験を行った。

取得する近傍単語を増やしてノードマッチングを行った結果を確認したが、精度の上昇は見られな

表 5.6: (日本, アメリカ) の近傍 10 単語

| 集合 X | 集合 Y |
|---|---|
| 韓国 | 米国 |
| 日本国内 | イギリス |
| 台湾 | 英国 |
| 中国 | カナダ |
| 欧米 | アメリカ合衆国 |
| 海外 | ヨーロッパ |
| 日本の文化 | オーストラリア |
| 日本国外 | フランス |
| アジア | ドイツ |
| わが国 | キューバ |

表 5.7: (日本, アメリカ) の近傍 10 単語でグラフノードマッチング

| 集合 X | 集合 Y |
|---|---|
| 韓国 | オーストラリア |
| 欧米 | ヨーロッパ |
| 日本国内 | 米国 |
| 海外 | 英国 |
| アジア | イギリス |
| 台湾 | キューバ |
| 中国 | ドイツ |
| 日本国外 | カナダ |
| わが国 | フランス |
| 日本の文化 | アメリカ合衆国 |

表 5.8: (東京, 東京タワー) の近傍 10 単語

| 集合 X | 集合 Y |
|---|---|
| 大阪 | 東京スカイツリー |
| 名古屋 | 名古屋テレビ塔 |
| 関西 | スカイツリー |
| 札幌 | 電波塔 |
| 横浜 | 屋上 |
| 京都 | オカンとボクと、時々、オトン |
| 神戸 | ランドマークタワー |
| 静岡 | 六本木ヒルズ |
| 新潟 | タワー |
| 神奈川 | お台場 |

表 5.9: (東京, 東京タワー) の近傍 10 単語のグラフマッチング

| 集合 X | 集合 Y |
|---|---|
| 大阪 | お台場 |
| 札幌 | 六本木ヒルズ |
| **名古屋** | **名古屋テレビ塔** |
| **横浜** | **ランドマークタワー** |
| 新宿 | タワー |
| 関西 | 東京スカイツリー |
| 神奈川 | オカンとボクと、時々、オトン |
| 神戸 | スカイツリー |
| 静岡 | 電波塔 |
| 京都 | 屋上 |

かった。原因として、与える単語対が近い位置関係にあり、それぞれの近傍として取得する単語が重複したことにより、同一単語でのマッチングが増え、事実上単語対が消失してしまった。

それ以外のケースにおいては、近傍単語の特徴にばらつきがあることがあげられる。

上記の表 (5.13) においては、(東京, 東京タワー) から、(地名, 塔などの名所) といった関係の取得を想定していたが、それぞれから取得した近傍単語を見るに、”東京”の近傍には地名が多く出現し、想定通りであったが、”東京タワー”の近傍は”名所”、”塔”、”高所”、”題材とした映画”など、大きくぶれてしまった。これは、元にした”東京タワー”という単語が、名所、シンボル、建造物、イメージなど多岐にわたる意味を含有していたからだと推測される。

また、近傍の取得数が 10 個の時から出現していた映画タイトルについては、コーパス中の出現回数が 31 回と、他の単語の出現率の 10 分の 1 以下[*2]であり、学習が十分になされなかったためにノイ

[*2] 鉄塔：570 回、東京タワー：610 回出現

表 5.10: (北海道, 札幌) の近傍 10 単語

| 集合 X | 集合 Y |
|---|---|
| 道内 | 函館 |
| 札幌 | 小樽 |
| 十勝 | 旭川 |
| 釧路 | 帯広 |
| 東北地方 | 釧路 |
| 十勝支庁 | 札幌市 |
| 道東 | 北海道 |
| 道南 | 仙台 |
| 札幌市 | 新潟 |
| 道北 | 岩見沢 |

表 5.11: (北海道, 札幌) の近傍 10 単語のグラフマッチング

| 集合 X | 集合 Y |
|---|---|
| 道内 | 北海道 |
| 釧路 | 釧路 |
| 札幌 | 函館 |
| **十勝** | **帯広** |
| 札幌市 | 岩見沢 |
| 道東 | 旭川 |
| 十勝支庁 | 札幌市 |
| **東北地方** | **仙台** |
| 道南 | 小樽 |
| 道北 | 新潟 |

表 5.12: (北海道, 札幌) の近傍 20 単語のグラフマッチング

| 集合 X | 集合 Y |
|---|---|
| 道内 | 北海道 |
| 釧路 | 釧路 |
| 札幌市 | 札幌市 |
| 札幌 | 函館 |
| **十勝** | **帯広** |
| **青森県** | **青森** |
| 根室 | 室蘭 |
| **道東** | **北見** |
| **九州** | **福岡** |
| 十勝支庁 | 旭川市 |
| 釧路市 | 岩見沢 |
| **東北地方** | **仙台** |
| 胆振 | 江別 |
| 函館 | 名古屋 |
| 道南 | 旭川 |
| 空知 | 丘珠 |
| 道北 | 苫小牧 |
| 道外 | 小樽 |
| 留萌 | 新潟 |
| 沖縄県 | 東京 |

表 5.13: (東京, 東京タワー) の近傍 20 単語のグラフマッチング

| 集合 X | 集合 Y |
|---|---|
| **大阪** | **通天閣** |
| 新宿 | サンシャイン 60 |
| 名古屋 | お台場 |
| 日比谷 | スカイツリー |
| 関西 | 日本電波塔 |
| 札幌 | 六本木ヒルズ |
| **横浜** | **ランドマークタワー** |
| 赤坂 | 大阪タワー |
| 静岡 | FCG ビル |
| 神奈川 | ビル街 |
| 福岡 | 東京スカイツリー |
| 麹町 | 名古屋テレビ塔 |
| 神戸 | 屋上 |
| 都内 | オカンとボクと、時々、オトン |
| 埼玉 | 電波塔 |
| 京都 | 銀座和光 |
| 愛知 | テレビ塔 |
| 青山 | タワー |
| 渋谷 | 展望台 |
| 金沢 | 鉄塔 |

ズ的に現れてしまったと認識できる。

## 5.5　実験２：近傍単語をクラスタリングし、クラスター同士のマッチングを見る

先の実験において、入力した一組の単語対それぞれから得た近傍単語同士のマッチングにより、入力単語対と相似な関係の取得は、近傍単語として取得できる単語の特徴に偏りがあることからよい結果が得られなった。

近傍単語の特徴に偏りがあることを考慮して、取得単語数を増やしてみた場合においても、近傍として設定した領域の重なりや、偏りが解消されないといった問題があった。そこで、偏りを解消するために近傍として取得する単語数を増やし、それぞれで同数のクラスターを作成し、クラスター同士のマッチングを考えた。

表 5.14: ”北海道”のクラスターと”札幌”のクラスターの対応関係：それぞれの近傍単語 100 個を、５個のクラスターに分割した。

| クラスター No.(北海道) | クラスター No.(札幌) |
|---|---|
| 0 | 0 |
| 1 | 2 |
| 2 | 1 |
| 3 | 4 |
| 4 | 3 |

以下に、クラスターごとの対応と関係性の相似が見られた単語対の例を挙げていく。

- (北海道:0, 札幌:0) = (後志, 倶知安・岩見沢市・余市),(十勝郡, 十勝)
- (北海道:1, 札幌:2) =なし
- (北海道:2, 札幌:1) =(十勝支庁, 帯広市),(道北, 稚内・名寄市),(空知支庁, 旭川空港・滝川市),...
- (北海道:3, 札幌:4) =(道東, 釧路),(道南, 苫小牧),(札幌, 中島公園),...
- (北海道:4, 札幌:3) =(東北地方, 仙台),(九州, 福岡),(北海道南部, 函館),...

ほぼすべてのクラスター間の対応で相似関係にある単語対を見つけられた。

# 第 6 章

# 結び

本章で、本研究を振り返っての考察と、今後の展望を述べて、研究の結びとする。

## 6.1 実験結果の考察

前章 (5.3) にて梅山氏の手法の有効性と word2vec の学習モデルの妥当性をチェックしたが、この実験を通して、word2vec の出力ベクトルの和差で獲得できる関係性については、今回の実験で利用した、余弦類似度で作成したコスト行列にハンガリアン法を適用する手法により抽出できることが期待される。

また、後に k-means により分割したクラスターを重心に関する、クラスター同士のノードマッチングによってささやかながらも関係性が見て取れた。

本研究ではベクトルの和差によって抽出できる単語の類似性については、ハンガリアン法を用いたノードマッチングの手法により抽出できることが期待される。また、近傍単語同士の対応関係についても、クラスタリングを用いることにより単語対の作成時の制限を緩めることで、ある程度の抽出ができる兆候を見ることができた。

しかし、いずれの観点についても検証が不十分であり、ハンガリアン法を用いたノードマッチングの強みのひとつでもある、回転したグラフや拡大・縮小されたグラフ構造においての同型性も抽出できる利点を word2vec の出力ベクトル空間においても有効活用できるかどうかの検証ができていない。

また、単語対から取得する n 個の単語同士の対応関係の抽出についても、一方で n 個の単語を取得してから、もう一方の n 個の単語の取得仕方を決めるなどの工夫をすることができなかった。

今回確認した 2 点については、あくまでも期待できる程度のものであり、今後の検討を要するものである。

## 6.2 期待する展望

今回の研究の過程で、グラフの同型性判定を行った。word2vec の出力ベクトルにおいても対応関係の抽出が可能であると期待されたことから、単語ベクトルを用いて文グラフを作成し、文書内に存

在する文グラフ同士の同型性をみることで、文構造の要約や検索に応用することが考えられる。

そもそも word2vec を始めとしたツールにより学習される単語の意味表現ベクトルは、要約や検索といった種々の自然言語処理の課題において、その精度向上のための足掛かりとして、意味を持つ最小単位である単語・形態素の数値表現を与えることを目的としているので、こうした処理がまた別な自然言語処理タスクの一助となるならこの上なく喜ばしいことである。

# 謝辞

本研究を進めるにあたって、特別な事情により普通よりも長い期間、丁寧にご指導いただきました原口先生に心から御礼申し上げます。長い期間の在籍も、研究室内でサポートくださいました大久保先生、配属時の事情など御心配戴きました吉岡先生にも、厚く御礼申し上げます。

研究を様々な面でサポートしていただきました先輩方、研究室の皆様にも大変お世話になりました。

未熟な私がここまでやってこれたのは、サポートしていただきました皆様のおかげであり、すべての方々に感謝を申し上げます。

平成二十八年　二月

平田　泰樹

# 参考文献


[1] J.R.Firth. A synopsis of linguistic theory 1930-55. Studies in Linguistic Analysis, pp.1-32, 1957.

[2] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, Jeffrey Dean Distributed Representations of Words and Phrases and their Compositionality.
https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf, NIPS 2013.

[3] MeCab: Yet Another Part-of-Speech and Morphological Analyzer.
http://taku910.github.io/mecab/, 2013.

[4] 日本語 WordNet
http://nlpwww.nict.go.jp/wn-ja/

[5] ダヌシカ ボレガラ, 岡直観, 前原貴憲 機械学習プロフェッショナルシリーズ ウェブデータの機械学習. 株式会社 講談社, 2016.

[6] 宮崎修一グラフ理論入門 基本とアルゴリズム 森北出版, 2015.

[7] 喜多陵. 記述の構造類似性に基づく法的観点と判例のマッチング. Master's thesis, 北海道大学大学院情報科学研究科, 2014.

[8] 梅山伸二 An eigendecomposition approach to weighted graph matching problems. IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, pp.695-703, Sep, 1988.