卒 業 研 究 報 告 題 目

K-means法による次元削減を用いた文書分類

Classification of Documents using Dimensionality Reduction by K-means Method

指 導 教 員　　出口 利憲　教授

岐 阜 工 業 高 等 専 門 学 校　電気情報工学科

2017E23 瀬尾 慎介

令 和 4 年（ 2 0 2 2 年 ）　　2 月 1 7 日 提 出

Abstract

In this study, a dimensionality reduction method using Word2vec and K-means method is proposed. This method reduces the dimensionality by multiplying the document matrix by the matrix created by the K-means method. In order to verify the effectiveness of this method, experiments of document classification are carried out using the following three dimensionality reduction methods.

1. Latent Semantic Analysis
2. Dimensionality reduction method using hierarchical clustering
3. Dimensionality reduction method using K-means method

The percentage of correct answers for each document classification is conpared. The second method is the dimensionality reduction method proposed in last year's research. The text data used in this study were articles from livedoor news.

As a result of the experiments, the effectiveness of this method over the method of last year's research is confirmed. However, some issues with the calculation time were found. A method to visualize the dimensionality reduction process using the K-means method is also proposed, and is verified its effectiveness. Word Clouds visualized the dimensionality reduction process. As a result of the verification, its sufficient effectiveness is confirmed.

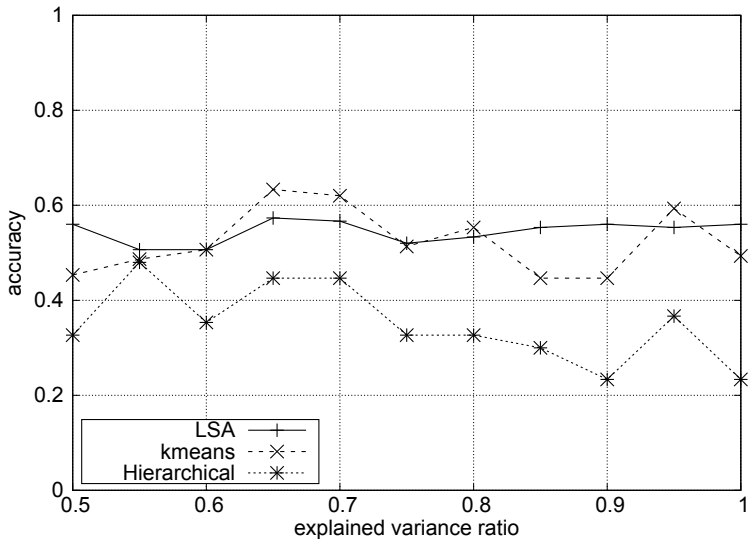

Figure 0.1 A result of the comparison experiments.

目 次

第1章　序論

現代社会は、インターネットの発展によってあらゆる情報が行き交う情報社会と化している。あらゆる情報がデータとして蓄積されるようになり、それが共有されることで我々は様々な情報を入手し、日常をより豊かにすることができるようになった。しかし、その情報量は日々増大し続け、その内容も有益なものばかりでは無く悪意ある情報も増えてきた。その膨大なデータから必要なデータのみを抽出することは多くの労力が必要であり、さらにこれを個人で見極め、正しい情報か判断することは極めて困難である。このような問題を解決するのがデータマイニングという技術である。データマイニングは、膨大なデータから有用なパターンやルールを発見する技術で、いまだ発展途上の分野である。

本研究では、文書間の類似度計算に活用される次元削減という技術において、新しい手法を提案・実装し、その有用性の確認や他手法との比較を行う。提案手法は、単語の分散表現である Word2vec と非階層的クラスタリング手法である K-means 法を用いた次元削減法で、これにより単語の意味を考慮した次元削減が可能になると考えられる。また、単語クラスターの分析により、次元削減工程の理解が期待できる。対象の文書には株式会社ロンウイットが公開している livedoor ニュースコーパスを使用した。このデータは 9 カテゴリに分かれているため、文書分類した際に評価しやすいと考えた。

実験では、次元削減を下の 3 手法で実装し、それぞれで文書間の類似度を導出する。

- 潜在意味解析（LSA）
- 昨年度研究手法（Word2vec+階層的クラスタリング）
- 本研究手法（Word2vec+K-means 法）

潜在的意味解析は従来からの次元削減法で、昨年度研究手法と合わせて比較することで本研究手法の有用性を検証する。検証には昨年度の研究で用いられた正解率を評価指標として使用し、データ数やパラメータを変更して結果を比較する。また、単語クラスターの分析を行い、次元削減の構造から有用性の検討を行う。分析にはワードクラウドを使用し、構造を視覚的に評価する。

これらの検証結果を用いて、最終的に本研究の提案手法の有用性がどのようなものかを検討する。

第2章　テキストマイニング

2.1　テキストマイニング

2.1.1　データマイニング

データマイニングとは、膨大なデータを様々な手法で分析して、有用なパターンやルールを発見する技術である。分析手法には統計学や機械学習などが用いられ、コンピュータを使うことで人では見出せない法則を発見することができる。企業のマーケティング戦略や顧客管理などで活用されており、一般に広く浸透している技術である。

2.1.2　テキストマイニング

データマイニングの中でも、テキストデータに焦点を向けたものがテキストマイニングである。自然言語処理を用いて文章を単語列に分解し、それらの出現頻度や相関関係を分析することで有用なパターンやルールを発見することができる。

英語などの単語の境界が明確な言語に対して、日本語のような境界が曖昧で文法的な揺らぎが大きい言語は解析が困難だったが、自然言語処理の発展により実用的な水準の分析が可能となった。主にSNSのデータやアンケートが対象にされ、流行の予測や課題改善などに活用されている。

2.2　自然言語処理

2.2.1　自然言語[1)]

自然言語とは、人間がコミュニケーションを行うのに用いる自然発生的な言語である。構文や単語の用法に厳格な規則が存在せず、社会的文脈に沿った曖昧な規則が存在すると考えられるものを指す。つまり我々が普段使用している言語のことで、「日本語」「英語」「中国語」等のことである。対義語として人工言語、形式言語が存在し、これにはプログラミング言語や論理式など、構文や意味が厳格に定義された言語が当てはまる。

自然言語は規則が曖昧なため、使用する単語を入れ替えたり、単語の順番を入れ替えたりすることができる。感情でも文章を制御しやすいため、形に縛られない自由な情景表現が可能となっている。また、解釈の自由度が大きいため、話者ごとに異なる解釈が存在する。そのため、話者が直面した状況に応じて解釈を変化させることで他の話者とのコミュニケーションを可能としている。

2.2.2　自然言語の曖昧性

自然言語の曖昧性には、多義性と類義性の二つがある。

多義性とは、一つの単語が複数の意味をもっており、解釈が複数になることをいう。例えば「首」という単語は、頭を支える体の部位としての意味を持つが、その他に解雇の意味も持っている。これを判断するには、文脈からその「首」が持つ意味を判断する必要がある。また、この問題は日本語だけのものではなく、英語でも「light ー光、軽い」のような複数の意味を持つ単語が多く存在する。

類義性とは、複数の単語が同一の意味を持っていることをいう。例として、「太陽」「日」「お天道様」は全て同じものを指す言葉である。しかし、これらをほぼ同義だと認識したうえで判別し、場面によって使い分けることは困難である。

自然言語に存在するこれらの曖昧性は、自然言語をコンピュータで扱う上で大きな壁となっている。

2.2.3　自然言語処理

自然言語処理とは、自然言語をコンピュータに処理させて、データから自然言語への変換や、自然言語のより形式的な表現への変換などを行う技術のことである。機械翻訳や予測変換などに活用されており、一般に広く浸透した技術となっている。

自然言語は前項のように曖昧な規則しか存在しないため、単語間の境界や語義、構文の曖昧性の解析が非常に困難である。そのため、処理には言語学や確率論、統計学、機械学習などを使って形態素解析や構文解析、意味解析、文脈解析等を行うことで、意味を含めた解析が可能となっている。

2.2.4　形態素解析

形態素解析とは、文法的な情報の注釈がない自然言語のテキストデータを、文法や単語の品詞情報などに基づいて形態素（言語で意味を持つ最小単位）に分解し、それぞれの品詞等を判別する技術である。形態素解析を用いることで単語ごとの出現頻度や単語間の相関関係が計算可能となり、テキストデータからコンピュータで解析可能な形式への変換が可能となる。

形態素に分解する際、英語などの単語間が区切られている言語は単語の接辞や変化を調べるだけでいいが、日本語のような区切られていない言語は分解が困難である。その

ため、単語分割のために膨大な辞書データが必要となる。また、辞書データにない単語の処理も困難だったが、固有名詞に強い辞書データの開発により改善されつつある。

形態素解析を行うツールは形態素解析器と呼ばれる。本研究では形態素解析器として「MeCab」を使用し、辞書データに「mecab-ipadic-NEologd」を使用した。

第3章　実験で使用した技術

3.1　文書のベクトル化と類似度計算

3.1.1　文書のベクトル化

テキストマイニングを行う上で、まず最初に文書をベクトル化する必要がある。文書は文字が羅列しているテキストデータであるため、そのままの状態では数学的な計算が困難である。そのため、何らかの方法で文書の特徴量を抽出し、文書を特徴ベクトルに変換する必要がある。

特徴ベクトルは n 個の特徴量 $x_i(i = 1, 2, 3, \cdots, n)$ を縦に並べた n 次元ベクトルで定義される。文書 d_j の特徴ベクトル X_j の定義を以下に示す。

$$X_j = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{pmatrix} \tag{3.1}$$

x_i には文書の特徴量が入り、特徴量数 n はベクトル表現手法ごとで異なるが、主に単語数と等しくなる。また、プログラムによる解析の際は、各特徴ベクトルを更に横に並べた $n \times m$ 行列を作成することが多い。

3.1.2　TF-IDF

TF-IDF とは、TF（Term Frequency）と IDF（Inverse Document Frequency）の二つの指標を基に計算される単語の重要度である。全文書中の全単語と文書間の重要度を計算し、文書行列と呼ばれる単語数 × 文書数の行列を作成する。

TF-IDF は TF と IDF の二つの値で構成される。TF は、単語の出現頻度を表す値である。ある文書に対して出現頻度が高い単語は、その文書と関わりが深い単語であると考えることができるため、重要度が大きくなる。IDF は、ある単語の文書出現頻度の逆数で、つまりその単語の希少度を表す値である。ある文書に対して出現頻度が低い単語でも、ほかの文書でほとんど出現しない場合はその文書特有の単語と見ることができるた

め、IDF が大きくなる。この TF と IDF の積が TF-IDF の値となる。

これらの導出式はいくつかあるが、本研究で使用した計算式を下に示す。

$$tf_{i,j} = \quad (\text{文書 } d_j \text{ での単語 } w_i \text{ の出現回数}) \tag{3.2}$$

$$idf_{i,j} = \log \frac{1 + \text{総文書数}}{1 + \text{単語 } w_i \text{ が出現する文書数}} + 1 \tag{3.3}$$

$$tfidf_{i,j} = tf_{i,j} \times idf_{i,j} \tag{3.4}$$

TF は文書内での単語出現回数を出現頻度としている。単語の出現回数と出現頻度は比例するため、代用することが可能である。IDF は分母分子に 1 を足すことで 0 になるのを防いでいる。また、全文書に出現する単語が存在した場合に、対数計算結果が 0 になってしまうため、全体に 1 を足すことで防いでいる。

3.1.3 cos 類似度

cos 類似度とは、ベクトル同士がどの程度似ているのかを示す指標の一つである。ベクトル同士の類似度計算はいくつかあるが、cos 類似度はベクトル同士のなす角のコサイン値を計算することで類似度としている。コサイン値なので、ベクトル同士がまったく似ていない、つまりベクトル同士が逆方向だった場合に -1 となり、完全に同じ、つまりベクトル同士が同方向だった場合に 1 となるような値になる。

ベクトル同士のなす角の大きさは内積の導出式を変形することで求めることができる。下に cos 類似度の導出式を示す。

$$\cos(\vec{a}, \vec{b}) = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}||\vec{b}|} \tag{3.5}$$

3.2 次元削減

次元削減とは、特徴ベクトルの次元数を減らすことである。文書を特徴ベクトルに変換する際、その特徴量数は膨大になることが多い。前項の TF-IDF も単語数だけ特徴量が存在することになるため、文書数、文字数が多いほど特徴量数が増加する。しかし、特徴量数の増大はデータ理解への壁となり、解析の計算量増大にもつながってしまう。そのため、次元削減によってなるべくデータの意味を保ったまま特徴量を減らすことで、計算量の削減や新たな特徴量の抽出が可能となる。

3.2.1　主成分分析（Principal Component analysis;PCA）[2)]

主成分分析とは、データの分散から全体のばらつきを最も表す主成分を抽出する多変量解析の手法で、データの次元削減に用いられる。n 個の特徴量を持つデータ群を n 次元座標としてプロットしたと仮定したとき、それらのデータのばらつき方は偏ったものになることが多い。そのばらつきが大きい所に軸を作成していくと、データ群を表す新たな座標系を作成することができる。この軸を主成分と呼び、ばらつきが大きい主成分ほどそのデータをよく表していると考えることができる。この新たな座標系へデータを座標変換させることでデータのばらつき方を考慮したベクトルに変換することができる。このベクトルはばらつきが大きい主成分のみを残すことで低次元ベクトルに近似できるため、次元削減することができる。

Figure 3.1 は二次元データを対象とした場合の例である。ベクトル (x, y) で表される二次元データ群をグラフにプロットすると、より分散が大きい軸 Z_1, Z_2 を見出すことができる。この軸を新たな座標系として新たなベクトル (z_1, z_2) を計算することで、データの意味を保ったまま異なるベクトルで表現することができる。次元削減をする際は、ばらつきが小さい軸を無視することで行う。Figure 3.1 では、軸 Z_2 がばらつきが小さいため、データ群は一次元ベクトル (z_1) に次元削減することができる。

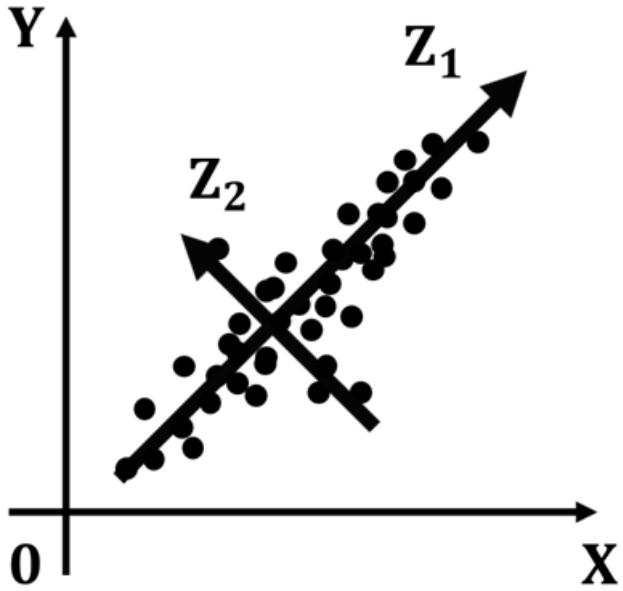

Figure 3.1　Example of two-dimensional data.

具体的な定義式を下に示していく。N 個のパラメータを持つ P 個のデータ $x_p(p = 1, 2, \cdots, P)$ と、主成分を表す K 個の負荷量ベクトル $a_k(k = 1, 2, \cdots, K)$ について、ベ

クトル $z_{k(p)}$ はデータ x_p と負荷量ベクトル a_k の内積によって与えられる。

$$z_{k(p)} = x_p \times a_k \tag{3.6}$$

負荷量ベクトル a_k は単位ベクトルであり、各主成分要素の分散が順に最大化されるように選ばれる。また、主成分の数 K は元のデータのパラメータの数以下になる（$K \leq N$）。この値を N より小さくとることで次元削減することができる。

主成分分析のメリットとして、非常に多くのデータを扱える点がある。通常、データの多さは分析結果の精度向上につながるが、一方で解析の過程で偏りが生じる場合がある。しかし主成分分析はばらつきをいくつも集めて解析するため、データの意味を保持しやすい。そのため、分析結果に偏りが生じにくくなり、信頼できる分析ができる。

しかしデメリットとして、データを一部捨てなければいけない点がある。捨てるデータは本体データにあまり影響がないと思われるものから選ばれるが、本当は重要なデータだった可能性は否定することができない。

3.2.2 潜在意味解析（Latent Semantic Analysis;LSA）[3)]

潜在意味解析とは、文書の潜在的なトピック（話題）を推定することで解析を行い、そのトピック数まで次元削減する手法である。文書行列から特異値分解（SVD）を使ってトピック数 × 文書数の行列を抽出、そこから重要なトピックのみを残すことで低次元ベクトルに近似し、次元削減することができる。

上で用いられた特異値分解とは、任意の実行列を二つの直行行列と特異値からなる対角行列の内積に分解する手法である。特異値分解の式を下に示す。

$$TD = U\Sigma V^T \tag{3.7}$$

この式における U, V^T が左特異ベクトル、右特異ベクトルと呼ばれ、Σ は特異値からなる対角行列である。左特異ベクトルと右特異ベクトルはそれぞれ単語数 × トピック数、トピック数 × 文書数の行列で、トピックと単語、文書の関係性を表している。対角行列はトピック数 × トピック数の行列で、その要素はトピックの重要度を表す。トピックは重要なものから順に並んでいるため、重要なトピックの列、行のみを抜き出すことで次元削減が可能である。

この特異値分解は、前項の主成分分析にも用いられることがある。主成分分析の主成分計算は、突き詰めると固有値問題に行き着く。この固有値問題を解く方法として特異値分解が用いられるため、主成分分析と潜在意味解析は本質的に似ている部分がある。

3.2.3　累積寄与率

次元削減において、削減する次元数の決定は非常に難しい問題である。削減しすぎると必要な情報まで消えてしまう可能性があるため、重要な情報とそうでない情報の線引きが必要になる。その時に多く用いられるのが寄与率、または累積寄与率である。寄与率はデータの重要度を表す値で、主成分分析での主成分、潜在意味解析でのトピックごとの重要度を表す。寄与率の合計が累積寄与率で、すべてのデータでの累積寄与率を 100 % で表す。データ削減後の累積寄与率を 80 %や 90 %などに定め、なるべく少ない次元数でその値になるよう次元削減する。

寄与率 P_n と累積寄与率 C の導出式を下に示す。

$$P_n = \frac{\lambda_n}{\sum_{p=1}^{P} \lambda_p} \tag{3.8}$$

$$C = \sum_{i=1}^{N} P_i \tag{3.9}$$

ここで λ_n は各主成分やトピックの固有値で、分散と対応している。また、N は次元削減後の次元数である。式 3.8 からわかるように、分散の値が大きいトピックであるほど寄与率が大きくなる。

3.2.4　クラスター分析

クラスター分析とは、異なるものが混ざり合う集団から互いに似ているものを集めてクラスターと呼ばれる集団を作り、対象を分類することである。形や色などで分類する場合とは違い、クラスター分析は分類の基準や評価があらかじめ決められていない、教師なしの分類法である。

クラスター分析の手法には、階層的クラスタリングと非階層的クラスタリングの二つが存在する。階層的クラスタリングは似ている対象から順にクラスターに分類していく手法である。二つの対象の距離を計算し、その距離が短い者同士から順にクラスターを作成していく。クラスター同士の距離の計算方法はいくつか存在し、対象のデータに最も適した方法を選択する。代表的な計算方法は以下の 4 つである。

- 最短距離法（最も近いデータ同士の距離をクラスター間の距離とする）
- 最長距離法（最も遠いデータ同士の距離をクラスター間の距離とする）
- 重心法　　（クラスター内のデータの重心をクラスターの座標とする）

- ウォード法（仮に結合した際の分散が最も小さいクラスター同士が結合する）

また、階層的クラスタリングの分類過程は階層的な構造になるため、Figure 3.2 のようなデンドログラム（樹形図）で表すことができる。このデンドログラムの横に閾値で直線を引くことで、指定数のクラスターに分類することができる。

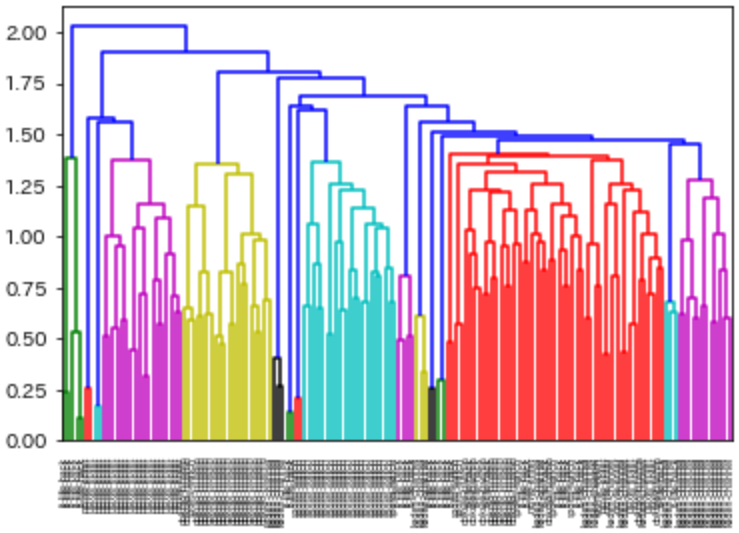

Figure 3.2　Dendrogram.

非階層的クラスタリングは集団全体から、似た対象が同じクラスターに集まるよう分割する手法である。しかし階層的クラスタリングとは異なり階層的な構造を持たず、あらかじめいくつのクラスターに分類するかを決めておく必要がある。そのため事前に計算を行うことができず、また最適なクラスター数を自動的に計算する手法も存在しないため、分析者によって大きく結果が変わることがある。非階層的クラスタリングの手法として、K-means 法があるが、これについては次項に詳しく記載する。

本研究では文書の分類と昨年度研究手法に階層的クラスタリング、次元削減に非階層的クラスタリングである K-means 法を使用している。階層的クラスタリングでのクラスター間距離の測定方法は最も使用されている ward 法を採用した。

3.2.5　K-means 法

K-means 法とは、クラスターの平均を用いてあらかじめ決められた数に分類する非階層的クラスタリングのアルゴリズムである。データ総数 n、クラスター数 k の時、以下の流れで実装される。

1. データ $x_i(i = 1, 2, \cdots, n)$ の中から k 個を選び、クラスターの中心 $V_j(j = 1, 2, \cdots, k)$

の初期値とする。

2. x_i から V_j までの距離を求め、一番近い中心のクラスタにデータを割り振る。
3. 各クラスタの重心を求め、V_j に割り振りなおす。
4. 2,3 を繰り返し、重心の移動が閾値以下になれば処理を終了する。

この実装は本研究で使用したもののアルゴリズムである。初期値の決定方法はいくつか存在するため、違う決定方法で実行すると結果が大きく違うものになる可能性がある。この場合は初期値は自分で決定、もしくはランダムで決定する必要がある。

K-means 法のメリットとして、計算量が少ない点がある。階層的クラスタリングでは、すべての組み合わせの距離を計算する必要があるが、K-means 法はクラスターまでの距離を計算するだけなので、階層的クラスタリングと比べて計算量が少なくなる。しかしデメリットとして、最適なクラスター数や初期値を自分で探す必要がある点がある。K-means 法の計算はクラスターの中心の初期値やクラスター数に依存するため、一度で最良の結果が得られるとは限らない。そのため、初期値をランダムに決定したり、何度か繰り返し実行して最良の結果を採用したりすることが多い。

3.2.6 Word2vec

Word2vec とは、2013 年に Google の研究者トマス・ミコロフ氏によって提案された、単語の分散表現を計算するモデルである。単語分散表現とは、単語をベクトル空間に埋め込むことで空間上の点として捉えることで、単語を実数ベクトルに変換する方法である。この実数ベクトルは数百次元からなり、ベクトル同士の類似度計算による単語の比較や、ベクトルの加減算による単語間の意味の計算まで可能となる。有名な例では、「王」から「男」を引いて「女」を足すことで、「女王」を求めることが可能になる。

3.3 Python

3.3.1 Python

Python はグイド・ヴァン・ロッサムによって創られたインタープリタ型の高水準汎用プログラミング言語である。動的な型付けやガベージコレクションなどの機能を持ち、手続き型、オブジェクト指向型、関数型プログラミングなどの幅広いプログラミングパラダイムをサポートしている。読みやすく、それでいて効率もよいコードをなるべく簡単に書けるようにするという思想が浸透しており、その単純さから初心者や非技術者に

利用される。

3.3.2 MeCab

MeCab 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の共同研究で開発されたオープンソースの形態素解析エンジンである。言語、辞書、コーパスに依存しない汎用的な設計方針を採用しており、C 言語、C++、Java、python 等、数多くの言語で使用することが可能である。また、設定することで様々な辞書を用いることが可能で、本研究では mecab-ipadic-NEologd という、固有表現や新語に強い辞書を用いている。

MeCab によって形態素解析をするときは、オプションを指定することで様々な結果を出力できる。例として以下の文を形態素解析し、形態素、品詞、標準形等を出力した結果を示す。

「すももももももものうち」

すもも　名詞, 一般,*,*,*,*, すもも, スモモ, スモモ

も　助詞, 係助詞,*,*,*,*, も, モ, モ

もも　名詞, 一般,*,*,*,*, もも, モモ, モモ

も　助詞, 係助詞,*,*,*,*, も, モ, モ

もも　名詞, 一般,*,*,*,*, もも, モモ, モモ

の　助詞, 連体化,*,*,*,*, の, ノ, ノ

うち　名詞, 非自立, 副詞可能,*,*,*, うち, ウチ, ウチ

3.3.3 scikit-learn

scikit-learn は Python のオープンソース機械学習ライブラリである。分類、回帰、クラスタリングなどのアルゴリズムモデルやサンプルのデータセットを備えており、容易に機械学習を試すことができる。本研究では TF-IDF、主成分分析、潜在意味解析、K-means などを Python で実装するために使用した。

3.3.4 Gensim

Gensim は自然言語処理に用いられる Python のオープンソースライブラリである。主に潜在意味解析のようなトピックモデルを扱いやすくするために作られたライブラリで、

他に Word2vec のような Word embedding 手法を扱うこともできる。本研究ではトピックモデルを扱う用途ではなく、Word2vec を実装するために使用した。

3.3.5 SciPy

SciPy は高度な科学技術計算をおこなうことが可能な Python のライブラリである。微積分や統計などの計算が可能で、機械学習やデータ分析などに使われる。本研究では文書のクラスタリングや昨年度研究手法の次元削減に使用した。

3.4 提案手法

3.4.1 次元削減行列による次元削減

本研究では次元削減に行列の積を用いた方法を提案する。TF-IDF で作成される文書行列 $\boldsymbol{A}$ は、行数が単語数 n、列数が文書数 m になる。これに対して行数が単語数 n、列数が任意の数 k の行列 $\boldsymbol{B}$ を作成すると、行列 $\boldsymbol{A}$ との積で $k \times m$ 行列が作成できる。この時 $k < n$ にしておけば行列 $\boldsymbol{A}$ の次元削減をしたことになる。具体的な式は下のようになる。

$$\boldsymbol{A} = \left(\begin{array}{c|cccc} Term & d_1 & d_2 & \dots & d_m \\ \hline w_1 & a_{1,1} & a_{1,2} & \dots & a_{1,m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ w_n & a_{n,1} & a_{n,2} & \dots & a_{n,m} \end{array}\right) \tag{3.10}$$

$$\boldsymbol{B} = \left(\begin{array}{c|cccc} Term & C_1 & C_2 & \dots & C_k \\ \hline w_1 & b_{1,1} & b_{1,2} & \dots & b_{1,k} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ w_n & b_{n,1} & b_{n,2} & \dots & b_{n,k} \end{array}\right) \tag{3.11}$$

$$\boldsymbol{C} = \boldsymbol{B}^{\top}\boldsymbol{A} \tag{3.12}$$

w_n は全文書に含まれる単語、d_m は文書、C_k はトピックを表す。$\boldsymbol{A}$ の転置行列に右から $\boldsymbol{B}$ を掛けることで $k \times m$ 行列の $\boldsymbol{C}$ が求められる。この $\boldsymbol{C}$ の要素が、m 個の文書に対する k 個のトピックの重要度になるように次元削減行列 $\boldsymbol{B}$ を作成すればよい。

3.4.2 Word2vec と K-means 法を用いた次元削減行列の作成

次元削減行列は単語 – トピック間の重要度を要素とした行列である必要がある。そのため、単語をクラスタリングし、作成されたクラスターをトピックであると捉えることで単語 – トピック間に関係性を持たせ、次元削減行列を作成する。

本研究では Word2vec と K-means 法を用いて単語のクラスタリングを行う。クラスタリングには対象同士の類似度を計算する必要があるため、単語を Word2vec を用いることでベクトルに変換する。Word2vec は分散表現なので、座標が近い単語同士は意味も似ているものになっている。そのため、この単語ベクトル群をクラスタリングすることで似た意味を持つ単語が集まったクラスターに分類することができる。また、K-means 法によるクラスタリングは、最終的に各クラスターの中心座標が求められる。その座標をクラスターの意味ととらえると、クラスターの座標と単語の座標との距離は、クラスターにおける単語の重要度とも取ることができる。この値を要素とすることで次元削減行列を作成することができる。

次元削減行列の定義を下に示す。

$$b_{n,k} = \begin{cases} D(w_n, V_k) & (w_n \in C_k) \\ 0 & (w_n \notin C_k) \end{cases} \tag{3.13}$$

V_k はクラスター C_k の中心座標、$D(w_n, V_k)$ は 2 点間の cos 類似度を求める関数である。cos 類似度の計算には式 (3.5) を使用した。この計算でユークリッド距離ではなく cos 類似度を使用する理由は、Word2vec での類似度計算方法であることと、類似度を直接表現できるためである。Word2vec は単語間の類似度計算に cos 類似度が使われるため、Word2vec による単語ベクトルを扱うこの計算も本家にならって cos 類似度を使用した。また、ユークリッド距離を使うと、似ている単語の場合に値が小さくなり、似ていない場合に大きくなってしまう。そうなると次元削減行列を作成する際にそのまま使えないため、そのまま使える cos 類似度を重要度とした。

3.4.3 次元削減の計算

式 (3.12) のままで計算すると、単語数が多い文書ほど重要度が大きくなってしまう問題がある。本研究では、文書のクラスタリングにユークリッド距離を用いた階層的クラ

スタリングを使用するため、この問題の影響を受けて精度が下がる恐れがある。そのため、単語数が多い文書ほど重要度を低くする計算が必要がある。そのため、各文書の要素を文書内の単語数で除算することで重要度の調整を行う。これによって文書ごとの差を考慮し、より平等なクラスタリングが可能になる。

次元削減の計算式を下に示す。

$$\boldsymbol{W} = \left(\begin{array}{c|cccc} Term & d_1 & d_2 & \dots & d_m \\ \hline C_1 & c_1 & c_2 & \dots & c_m \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ C_n & c_1 & c_2 & \dots & c_m \end{array}\right) (c_m : \text{文書 } d_m \text{ に含まれる単語数}) \tag{3.14}$$

$$\boldsymbol{C} = \boldsymbol{B}^{\top}\boldsymbol{A} \oslash \boldsymbol{W} \tag{3.15}$$

$\oslash$ はアダマール除算と呼ばれる演算子で、行列の要素同士で除算を行う。$\boldsymbol{W}$ は重要度を調整するための除算を行う行列で、列方向に各文書の単語数を要素としている。。$\boldsymbol{C}$ がクラスター数 $\times$ 文書数の行列なので、$\boldsymbol{C}$ を $\boldsymbol{W}$ でアダマール除算にかけることで各列が対応する文書の単語数で除算されるため、単語数に応じた重要度の調整が行われる。

3.4.4 正解率の計算

分類問題の評価方法として、ラベル付きの対象が正しいラベルに分類されている確率である正解率を求める方法がある。二値分類問題の評価方法の一つで、正が正と判断されたものと、誤が誤と判断された物の割合を表す値である。本来多クラス分類問題では正解率は存在しないが、本研究では正解率 Acc を下のように定義して、文書分類に適用する。

$$Acc = \frac{\text{予想と正解が等しいサンプル数}}{\text{全サンプル数}} \tag{3.16}$$

しかしクラスタリングによる分類は類似度が高い対象同士を集めてクラスターを作成するため、クラスター自身にラベルがついていない。そのため、対象が正しいクラスターに分類されたかどうかは、同じクラスターに属している他対象と比べることでしか判断することができない。

そのため本研究では、ラベルがクラスターに割り当てられるパターンを全て試し、そ

れぞれの場合で仮の正解率を求める。その中で最も高い正解率の場合を採用し、分類の評価とする。例えば J_1、J_2、J_3 の 3 つのラベルでジャンル分けされる文書を、文書分類で 3 つのクラスターに分類したとする。その場合、3 つのクラスターにラベルを割り当てるパターンは 6 通り存在する。この 6 通りのラベルのパターンをそれぞれクラスターに割り当て、式 (3.16) を用いて仮の正解率を 6 通り求める。そして、この中で最も値が大きかった仮の正解率が、分類の正式な正解率となる。

3.4.5　ワードクラウドによる次元削減過程の可視化

本研究では、Word2vec+K-means 法による次元削減の有用性の検証のために、次元削減で用いる K-means 法のクラスターを用いてワードクラウドを作成し、次元削減過程の可視化を実験する。潜在意味解析などの従来手法では、次元削減の過程が人間に理解できないため、次元削減後の値がどのような意味を持つものなのかがわからなかった。しかし、本研究手法は、K-means 法で作成する Word2vec の単語クラスターを用いるため、人間に理解できる値のみで計算されている。そのため、それらの値をより理解しやすいように可視化することで次元削減過程の可視化ができると考えられる。その値の可視化を実装するにあたって、直感的に理解しやすいワードクラウドを使用する。

ワードクラウドとは、文章中の単語の出現頻度の多さに応じて単語を色覚的に図示する手法である。ワードクラウドの例を Figure 3.3 に示す。大きさや色彩、角度などで単語を強調するため、文章を構成する単語が理解しやすい。そのため、この手法を用いると、その文章内の単語構成が直感的に理解できる図を作成することができる。この手法を用いれば各クラスターを構成する単語を可視化し、それぞれのクラスターが持つ意味を直感的に理解できると思われる。

Figure 3.3　Word Cloud.

第4章　実験

4.1　実験の概要

本実験では、以下の3種類の次元削減法で文書分類を行う。

- 潜在意味解析
- Word2vec+非階層的クラスタリング (K-means 法)
- Word2vec+階層的クラスタリング

潜在意味解析は、従来の方法との比較を行うために使用した。また、二つ目の次元削減法は昨年度の研究で用いられた手法で、K-means 法ではなくウォード法を用いた階層的クラスタリングによる方法になっている。これらと提案手法を比較することで、提案手法の優位性を検証する。実験では、潜在意味解析の際の累積寄与率を基に次元の削減数を決定し、それぞれ同じ次元数まで次元削減を行う。その後クラスタリングによる分類を行い、正解率の計算、比較を行う。また、その際に累積寄与率を変化させて実験することで、累積寄与率と精度の相関を検証する。

実験の文書サンプルは、livedoor ニュースコーパスを使用した。このコーパスは NHN Japan 株式会社が運営する「livedoor ニュース」の記事のデータで、事前に 9 つのジャンルに分けられている。本実験ではこの記事をジャンルごとに分類することを目標に行う。記事数やジャンル数を変化させながら文書分類を行い、記事数やジャンル数による正解率の変化を検証する。実験するパターンは以下の 2 つである。

- パターン 1

  ジャンル：3　総文書：30(1 ジャンル 10 文書)
- パターン 2

  ジャンル：3　総文書：90(1 ジャンル 30 文書)
- パターン 3

  ジャンル：5　総文書：50(1 ジャンル 10 文書)
- パターン 4

  ジャンル：5　総文書：150(1 ジャンル 30 文書)

ジャンルと 1 ジャンルごとの文書数を 2 パターンで組み合わせて比較することで、ジャンル数や文書数による正解率の変化を分析する。

また、本研究手法の有用性の検証として、WordCloud による次元削減の可視化を行う。

クラスターごとの単語の重要度を視覚的に図示し、それぞれのクラスターがどのような意味を持つのかを把握することで、次元削減後のベクトルへの理解を深めることができると考えられる。本実験ではこの方法の有用性について考察を行う。

4.2 実験準備

4.2.1 Google Colaboratory

実験を行う環境として、Google が機械学習の教育及び研究用に提供している Google Colaboratory(以降 Colab) を使用した。Colab はインストール不要で Python による機械学習・深層学習の環境を整えることが出来る無料のサービスである。本実験では Word2vec や MeCab を Python で実装するため、Python へのサポートが充実している Colab を使用した。Python のバージョンは 3.7.12 である。

4.2.2 MeCab[4)]

MeCab を Python で実装するため、mecab-python3 を使用した。mecab-python3 は Python で簡単に MeCab を利用できるラッパーで、OS へのインストールではないためコマンドツールとしての利用はできないが、Python のモジュールとして利用することは可能である。

使用する辞書は「mecab-ipadic-NEologd」を使用した。「mecab-ipadic-NEologd」は新語、固有表現に強く、語彙数も多いオープンソースな辞書である。本実験で使用するコーパスはニュース記事なので、固有名詞に強いこの辞書を採用した。導入は開発者である佐藤敏紀氏の GitHub ページより行った。

4.2.3 Word2vec[5)]

Word2vec を Python で実装するためにはモデルの学習、または学習済みモデルが必要である。そのため、本研究では東北大学の乾・岡崎研究室で作られた日本語モデルを使用した。このモデルは学習に Wikipedia の全記事が使われており、固有表現を考慮するように学習されている。モデルは複数あるが、その中でも 2019 年に学習された 100 次元のモデルを Gensim で読み込んで使用した。

4.2.4　テキストデータの取得[6)]

分類対象である livedoor ニュースコーパスは、RONDHUIT のサイトからダウンロードした。このデータは可能な限り HTML タグを取り除いたテキストデータではあるが、それでも必要ないデータが多い。そのため本実験では本文のみを対象とし、タイトルや URL などは事前に削除する処理を行った。

ジャンルは 3 つの場合、家電チャンネル、MOVIE ENTER、Sports Watch を使用し、5 つの場合は追加で独女通信、IT ライフハックの二つを使用した。理由としては、ジャンル同士の内容が比較的離れていることと、それぞれ内容が固まっているからである。livedoor ニュースコーパスはニュース記事をまとめたデータであるため、ジャンルによっては書かれている内容が広い範囲に及ぶものもあった。そのようなジャンルを分類するのは困難だと考えられるため、分類しやすいジャンルを実験の対象にした。

実験に使用する記事は、対象のジャンルの中からランダムに選択して使用した。処理には random モジュールを使用し、記事ファイルへのパスを格納した配列を shuffle 関数でランダムな順にした後、先頭から指定数までの記事を使用した。

4.2.5　テキストデータの前処理

取得したテキストデータに対して、MeCab を用いた形態素解析を行う。ベクトル化を行う際、文書に関わりがない単語があると分類する際のノイズとなってしまう。そのため、内容に関わりが深い単語のみを残すことで、ノイズ防止及びデータの削減を行った。

関わりが深い単語の判断方法として、MeCab で出力される品詞情報を使用した。文章の中にはさまざまな種類の品詞が含まれるが、品詞の中でも助詞、助動詞、記号などは文書の内容にあまり影響がない。そのため、直接影響があると思われる名詞、動詞、助動詞の単語のみを残した。また、「1 つ」や「2 冊」といった数字を含む単語は名詞に分類されているが、文章の内容にはあまり影響がないと思われる上、単語数の増加にもつながる可能性がある。そのため数字を含む名詞も削除し、以上をテキストデータの前処理とした。

4.2.6　TF-IDF[7)]

形態素解析を行ったデータに対して TF-IDF によるベクトル化を行う。Python による TF-IDF は scikit-learn の CountVectorizer、TfidfTransformer を使用する。CountVector-

izer は形態素解析を行ったデータを入力することで Bag of Words と呼ばれる形式の配列が作成される。この配列を TfidfTransformer に入力することで、文書行列の配列が作成される。

また、CountVectorizer で配列を作成した際、`get_feature_names_out` で単語の配列を取得する。この単語列は出力された配列と対応するように並んでおり、のちに次元削減行列を作成する際に使用する。

4.3 次元削減

4.3.1 累積寄与率及び次元削減数 [7)]

本実験では累積寄与率による正解率の変化を見るために、一定間隔で累積寄与率を変えて文書分類を行う。累積寄与率は 50 %〜100 %の 5 %間隔で変化させたものを使用し、その累積寄与率の時の次元数まで次元削減を行う。

累積寄与率の計算は scikit-learn の TruncatedSVD によって行う。この関数は本来特異値分解を計算するものだが、その計算結果に各トピックの寄与率がある。TruncatedSVD に文書行列を入力し、特異値分解を計算した後に `explained_variance_ratio_` を取得することができる。この値を用いて目的の累積寄与率までの次元数を計算していき、累積寄与率 50 %〜100 %ごとの次元を求める。

4.3.2 潜在意味解析による次元削減

潜在意味解析は前項で使用した scikit-learn の TruncatedSVD によって計算する。文書行列と前項で求めた累積寄与率ごとの次元を引数として、それぞれの次元削減行列を作成する。

4.3.3 Word2vec による単語のベクトル化

次元削減行列を作成するために、4.2.6 項で取得した単語の配列を Word2vec を用いてベクトルに変換する。Gensim で Word2vec のモデルを作成して `get_vector` の引数に単語を入力することで対応したベクトル配列を取得する。本実験では 100 次元のモデルを使用するため、100 次元配列を取得する。この配列を列方向に順に結合していき、単語数 ×100 の配列を作成する。

Word2vec で変換する際に、入力した単語のベクトルが存在しない場合がある。その

場合、最初の単語数と最終的に作成する配列の列数が一致しないため、のちの計算でエラーが発生する。そのためベクトルが存在しない単語は別の配列に格納しておく必要がある。

4.3.4 K-means 法による次元削減行列の作成 [7)]

次元削減行列を作成するために、前項のベクトル化した単語の配列を K-means 法によってクラスタリングする。K-means 法を実装するために scikit-learn の KMean を使用する。クラスタリング対象として前項で作成した配列を、クラスターの数として累積寄与率ごとの次元数を引数にすることでクラスタリングを行う。この実行結果として、単語の分類結果と各クラスターの中心座標が取得できるため、これと式 3.13 を用いて次元削減行列を作成する。

4.3.5 次元削減行列による次元削減の計算

次元削減の計算は、式 (3.15) のように行う。式 (3.14) の $\boldsymbol{W}$ は、文書行列で文書ごとの単語数を求めて作成し、行列の積の計算は Numpy の dot で行った。しかし、Word2vec に無かった単語の影響で、次元削減行列の行数と文書行列の列数が合わず、行列積の計算ができない。そのため、4.3.3 項で取得したベクトルが存在しない単語を用いて文書行列から Word2vec に無かった単語の列を削除する。その後、行列の積を計算し、次元削減を行う。

4.4 文書のクラスター分析

4.4.1 文書のクラスタリング

次元削減した行列に対してクラスタリングし、文書の分類を行う。クラスタリングには、SciPy の linkage を用いた階層的クラスタリングを使用する。同じく SciPy の pdist で距離計算を行い、その計算結果とクラスタリング方法を引数に指定することで階層的クラスタリングを行うことができる。本実験ではクラスタリング方法にウォード法を指定した。また、fcluster の引数に linkage の結果とクラスター数を入力することで、指定したクラスター数に分類することができる。

4.4.2　正解率の計算

分類した文書に対して、正解率の計算を以下の手順で行う。

1. クラスターに適当なラベルを割り当てる。
2. 正解率を計算する。
3. 全てのパターンを割り当てるまで 1、2 を繰り返す。
4. 最も値が大きい正解率を、分類の正式な正解率とする。

文書のクラスタリングで使用した sklearn の fcluster は、ラベルに 0 から順の数字を割り当てる。そのため、0 からクラスター数 −1 までの整数をクラスターに割り当てる。クラスターのラベルのパターン作成には、itertools ライブラリの permutations でラベルの全順列を作成することで実装した。また、正解率の計算には scikit-learn の `accuracy_score` を使用した。

4.5　ワードクラウドの作成

ワードクラウドの作成は以下の手順で行う。

1. 本研究手法で文書分類を行う。
2. 分類過程で作成されたクラスターを用いてクラスター数 × 単語数の行列を作成する。
3. 行列の各行を用いて各クラスターのワードクラウドを作成する。

行列の作成にはクラスターの中心座標と各単語間の cos 類似度を使用する。ワードクラウドは本来単語の出現頻度を使うため、重要度が高いほど値が大きくなる cos 類似度を代用で使用する。これによって各行がクラスターに対する単語の重要度を表す行列が作成される。

また、ワードクラウドの作成には、Python の WordCloud ライブラリの `generate_from_frequencies` を使用する。行列の列ベクトルに単語を結び付けた辞書型変数をこの関数の引数にすることで、各クラスターのワードクラウドを作成する。

4.6　実験結果

4.6.1　パターンごとの正解率

それぞれの実験パターンの、累積寄与率に対する正解率のグラフを Figure 4.1〜 4.4 に示す。グラフ全体から、本研究手法は昨年度研究手法と比べて比較的正解率が高く、潜

在意味解析とほぼ並ぶことがわかる。また、グラフの多くが凸凹した横ばいの線になっており、累積寄与率に対する正解率の値はあまり相関性が見られなかった。

また、グラフ間を比較して、以下の分析結果が得られた。

- パターン 1、パターン 2 のジャンル数が 3 つのグループはどの次元削減法でも正解率が高く、全体的に 0.8 付近の値をとっていることがわかる。また、潜在意味解析、本研究手法が昨年度研究手法より正解率が高いことが見て取れる。しかしどのグラフも凸凹しており、正解率が安定していないと見ることができる。
- パターン 3、パターン 4 のジャンル数が 5 つのグループはジャンル数が 3 つのグループと比較して正解率が低くなっていることがわかる。次元削減法での比較はジャンル数が 3 つの場合と同様だが、正解率の値が全体的に 0.6 以下になっている。しかし、グラフの凸凹がジャンル数が 3 つの場合より小さく、比較的安定していることがわかる。
- パターン 1、パターン 3 の文書数 10 のグループと、パターン 2、パターン 4 の文書数 30 のグループを比較すると、文書数 30 のほうが正解率が低くなる傾向がみられる。しかしその差もあまり大きくないため、文書数と正解率の間に相関はあまりみられないと思われる。

以上のことから、正解率と関係性があると考えられるのはジャンル数で、累積寄与率、文書数は関係性が低いと考えられる。また、ジャンルごとの文書数を増やすことでどの次元削減法でも正解率が安定すると思われる。

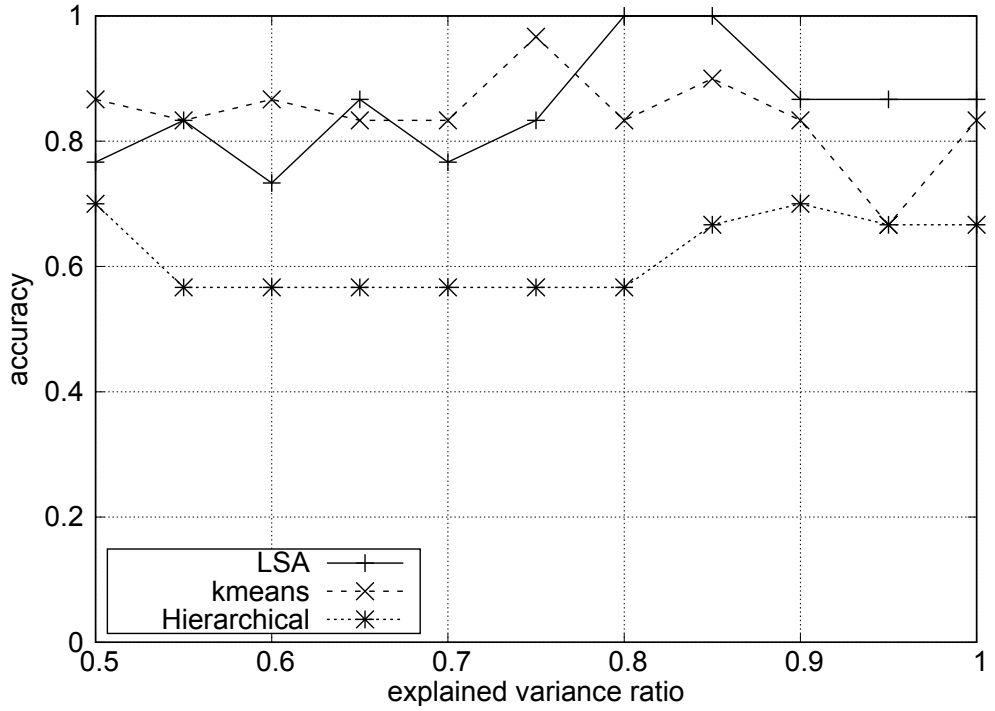

Figure 4.1 Pattern1(jenre:3 document:10).

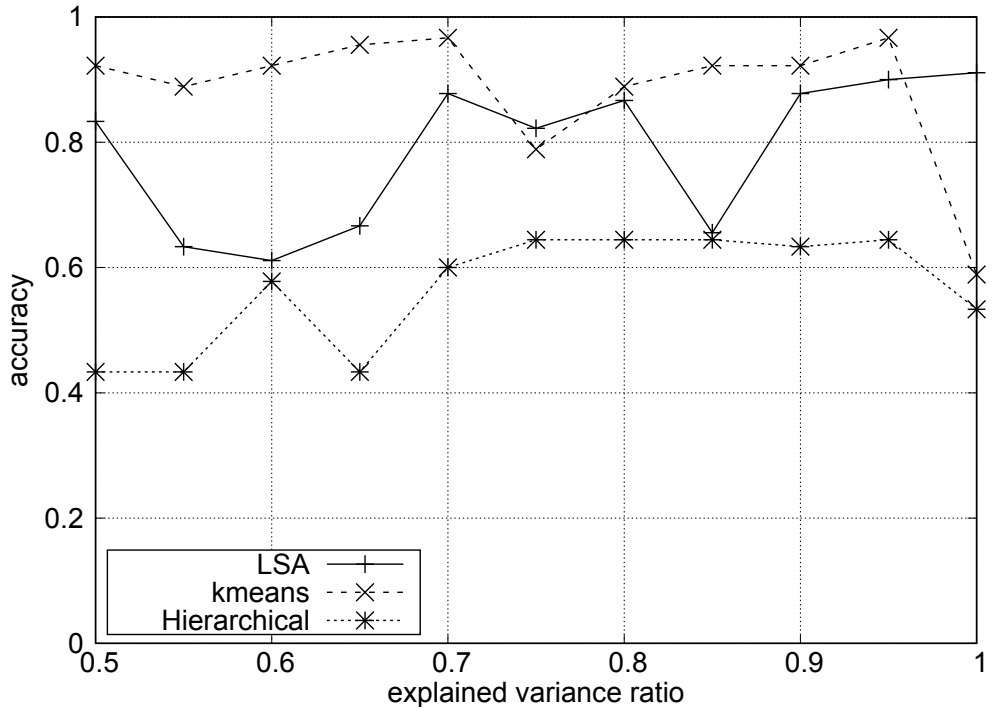

Figure 4.2 Pattern2.(jenre:3 document;30).

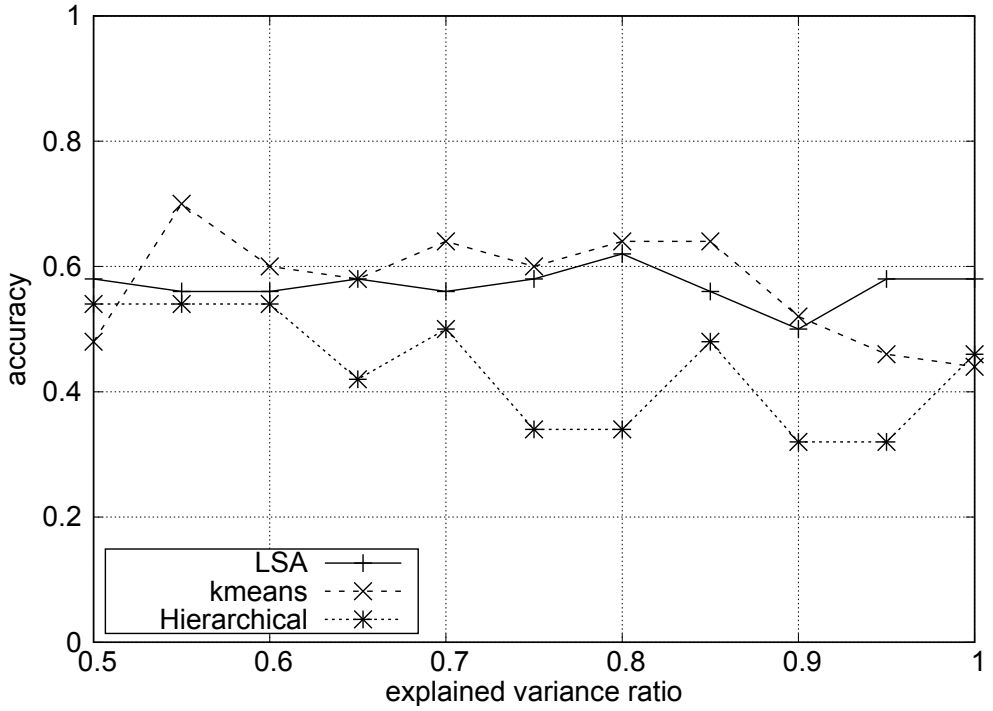

Figure 4.3　Pattern3(jenre:5 document:10).

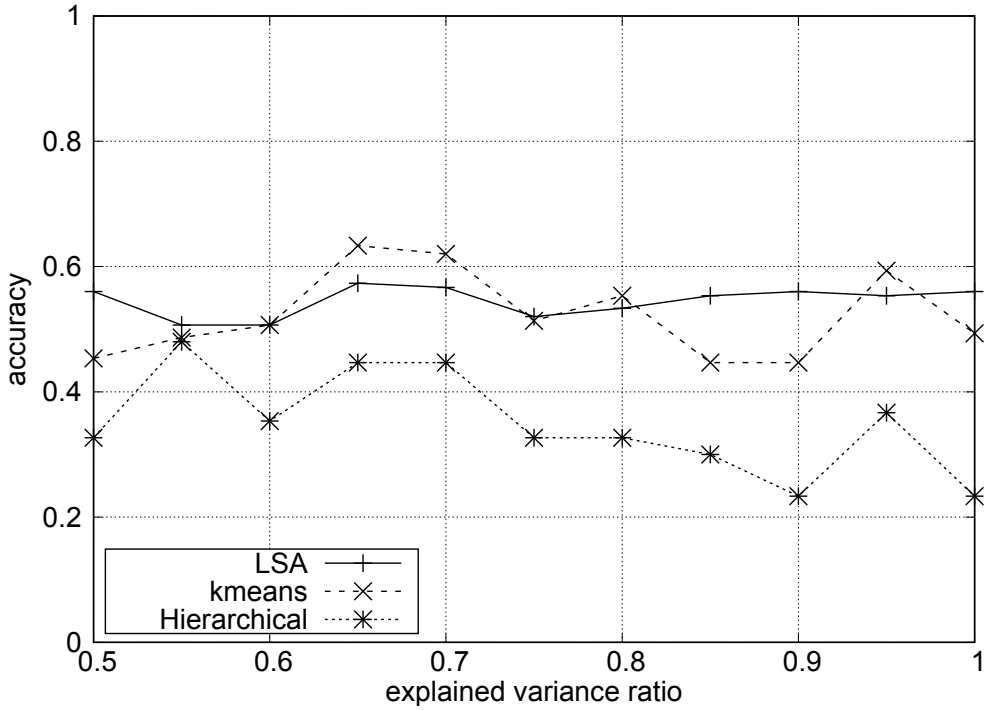

Figure 4.4　Pattern4(jenre:5 document:30).

4.6.2　計算時間の比較

4.6.1 項の実験の際、計算時間に大きな変化を感じたため、次元削減方法ごとの計算時間を検証した。累積寄与率は 4.6.1 項と同様、ジャンル数を 5、文書数を 30 で文書分類を行い、それぞれの次元削減法のクラスタリングを始める時から次元削減行列の作成までの時間を計測する。時間計測にはシステム CPU 時間とユーザー CPU 時間の合計を計測する、time ライブラリの `process_time` を使用した。

実験で得られた、累積寄与率と計算時間のグラフを Figure 4.5 に示す。昨年度研究手法と潜在意味解析は累積寄与率にかかわらずほぼ一定の計算時間に抑えられている。しかし、本研究手法は全体的に他手法より計算時間が多い上、累積寄与率と比例するように増加していることがわかる。

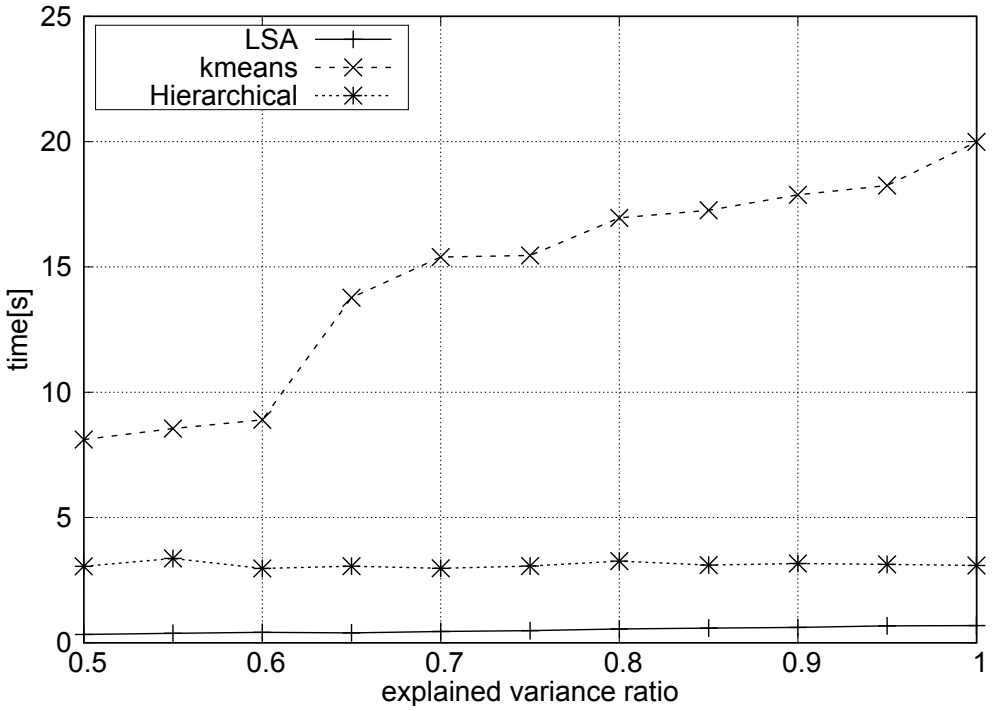

Figure 4.5　Culculation time.

4.6.3　ワードクラウド

パターン 4 の文書分類を、次元削減後の次元数を 10 に設定して行い、ワードクラウドを作成した。作成したワードクラウド画像を Figure 4.6〜 4.15 に示す。

それぞれのワードクラウドを見ると、どれも類似した意味の単語が強調されていることがわかる。また、ワードクラウドの内容は野球、英語、感情など、それぞれ異なったものになっていることも見ることができる。例えば Figure 4.6 は「開幕投手」「日本人選

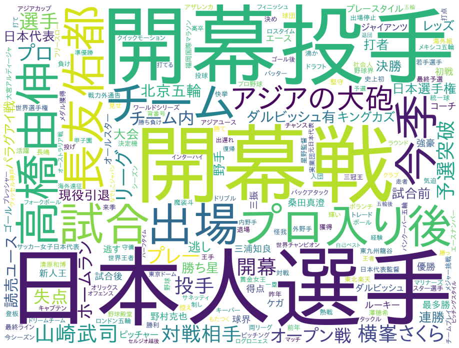

Figure 4.6　Label0.

Figure 4.7　Label1.

Figure 4.8　Label2.

Figure 4.9　Label3.

Figure 4.10　Label4.

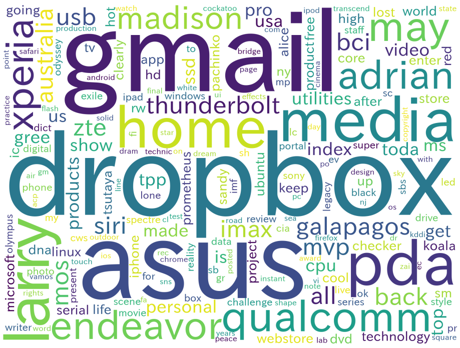

Figure 4.11　Label5.

Figure 4.12　Label6.

Figure 4.13　Label7.

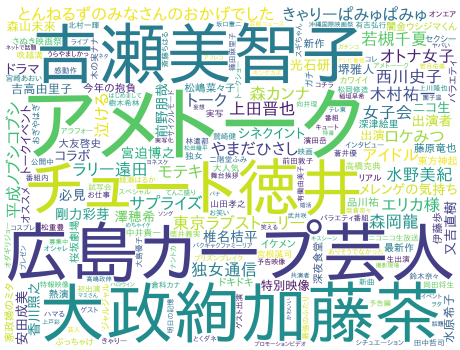

Figure 4.14　Label8.

Figure 4.15　Label9.

手」などの野球関係、Figure 4.7 は「美奈」「さゆり」などの女性名といったように、それぞれのワードクラウドで特色を見ることができる。それぞれのワードクラウドの内容は下のようになる。

- label0：野球関係（開幕投手、日本人選手、開幕戦、高橋由伸　など）
- label1：女性名（美奈、さゆり、久美、美加　など）
- label2：業務関係（取り組み、企業、自主的、実態　など）
- label3：カタカナ外人名（ジャック、ロイド、クリス、サラ　など）
- label4：接続詞（そのため、そのもの、一方、それ　など）
- label5：英単語（dropbox、gmail、asus、pda　など）
- label6：PC 関係（web ブラウザ、モバイルルーター、デバイス、省電力　など）
- label7：家事などの日常関係（保冷剤、お湯、歯磨き、食洗器　など）
- label8：芸能関係（吉瀬美智子、アメトーク、広島カープ芸人、大政絢　など）
- label9：感情関係（自分、羨ましかっ、思い、気持ち　など）

4.7　考察

4.7.1　各手法の比較

4.6.1 項及び Figure 4.1〜 4.4 より、本研究手法は全てのグラフで潜在意味解析とほぼ等しい正解率を出していることがわかる。その一方で、階層的クラスタリングを用いた昨年度研究手法より精度が高くなったことも読み取れる。この結果は、Word2vec と階層的クラスタリング、非階層的クラスタリング（K-means 法）の相性によるものだと考えられる。

Word2vec の N 次元ベクトル空間には、単語が空間中に広がるように配置されており、似た意味の単語は近くに配置される。そのため、Word2vec で似た単語を集めるには、周

辺空間の単語を集める必要があると考えられる。しかし、階層的クラスタリングは近い単語から順にクラスタリングしている方法である。この方法でも結果的に周辺空間の単語が集められるが、空間的な考えに基づいたクラスタリングではない。それに対してK-means法は初期値の座標を中心に周辺の単語を集めていく処理になる。そのため階層的クラスタリングよりも確実に似た意味の単語を集めやすく、クラスターが偏りづらいと考えられる。

4.7.2 正解率

4.6.1項及びFigure 4.1〜 4.4より、どの次元削減手法においても正解率はほぼ同様の変化になることが読み取れる。

4.7.3 計算時間の比較

4.6.2項及びFigure 4.5より、本研究手法は他手法より大幅に計算時間が長いことがわかる。理論的には、K-means法の計算量はデータ量Nのとき$O(Nk)$、階層的クラスタリングはN^2になるはずなため、この結果は予想外であった。

プログラムを確認して計算時間が多い部分を調べたところ、scikit-learnのKMeansによる処理が非常に長いことがわかった。そのためこの関数について調査を行ったが、計算時間が長くなる原因は特定できなかった。原因の予測だが、クラスターの中心座標の収束がうまく行われず、最大反復回数まで計算してしまっていることが原因ではないかと思われる。

4.7.4 ワードクラウドの有用性

4.6.3項及びFigure 4.6〜 4.15より、K-means法を用いたワードクラウドはクラスターごとの特色をはっきりと判別することができ、次元削減の解析方法として有用であると思われる。潜在意味解析などの次元削減法では各トピックの意味が理解できなかったが、この方法であれば次元削減の構造や意味が視覚的に理解できるため、次元削減後のベクトルへの理解も深まると思われる。

第5章　結論

本研究は、Word2vec と K-means 法による次元削減法を提案し、昨年度研究手法や潜在意味解析との比較によってその有用性を検証した。また、ワードクラウドを用いた次元削減過程の可視化を行い、新たな有用性を検証した。

他手法との比較は、livedoor ニュースの記事の分類を潜在意味解析、昨年度研究手法、本研究手法の 3 つの次元削減法で行う実験で検証した。分類結果は正解率で評価し、累積寄与率 − 正解率間のグラフを作成し、比較を行った。その結果、本研究手法は昨年度研究手法より正解率が高く、潜在意味解析とほぼ同等の精度を得ることができた。また、精度は分類するジャンル数が多いほど減少し、文書数が多いほど安定した値になることがわかった。よって、本研究手法は分類精度について従来手法である潜在意味解析とほぼ同等の有用性があることが言える。しかし、手法ごとの計算時間を計測した結果、本研究手法は他手法より多く時間がかかり、次元削減後の累積寄与率と比例して増加することがわかった。このことから、本研究手法は次元削減後の次元数が低い場合に限り、潜在意味解析とほぼ同等の有用性があるといえる。

ワードクラウドを用いた次元削減過程の可視化は、K-means 法のクラスターを用いてクラスター数 × 単語数の行列を作成することで各クラスターのワードクラウドを作成して検証した。その結果、各クラスターにそれぞれはっきりした意味を見出すことができた。また、意味が重複したクラスターも確認されなかったため、ワードクラウドによって次元削減過程は十分に理解できると思われる。よって、ワードクラウドを用いた次元削減過程の可視化は十分に有用性が確認できるといえる。

以上のことから、精度では潜在意味解析と同等だが、潜在意味解析ではできなかった次元削減過程の可視化ができるため、本研究手法には独自の有用性があるといえる。しかし、有用性のほかに課題もいくつか見受けられる。まずは、正解率の評価の妥当性に不安がある点である。本研究で使用した評価方法は昨年度の研究である程度の妥当性が確認されているが、それ以上に信頼性がある評価方法を検証できなかった。クラスター内のデータ構成を考慮した評価指標を考案できれば、更に信頼性がある検証ができると思われる。また、本研究手法の計算時間が非常に多かった点も大きな課題である。理論上は昨年度研究手法より計算時間は短くなるはずなので、原因を特定できれば有用性の向上につながると考えられる。

謝辞

最後に、本研究を進めるにあたり、ご多忙中にも関わらず多大なご指導をしていただきました出口利憲先生、また、共に勉学に励んだ同研究室のメンバーに厚く御礼申し上げます。

参考文献

1) 自然言語, Wikipedia.
https://ja.wikipedia.org/wiki/自然言語

2) 加納学, 主成分分析, 京都大学大学院工学研究科化学工学専攻プロセスシステム工学研究室, 1997.
http://manabukano.brilliant-future.net/document/text-PCA.pdf.

3) 【技術解説】潜在意味解析 (LSA) ～特異値分解 (SVD) から文書検索まで～, MIERUKA AI, 2019.
https://mieruca-ai.com/ai/lsa-lsi-svd/

4) Toshinori Sato, mecab-ipadic-NEologd : Neologism dictionary for MeCab, 2020.
https://github.com/neologd/mecab-ipadic-neologd（2021 年 9 月 6 日アクセス）.

5) 鈴木正敏, 日本語 Wikipedia エンティティベクトル.
http://www.cl.ecei.tohoku.ac.jp/ m-suzuki/jawiki_vector/（2021 年 9 月 7 日アクセス）.

6) ダウンロード - 株式会社ロンウイット.
https://www.rondhuit.com/download.html（2021 年 9 月 6 日アクセス）.

7) scikit-learn: machine learning in Python.
https://scikit-learn.org/stable/

8) 長谷川翔海, Word2vec を利用したクラスター分析による文書の分類, 岐阜工業高等専門学校電気情報工学科卒業研究報告, 2020.
http://www.gifu-nct.ac.jp/elec/deguchi/sotsuron/hasegawa.pdf

9) 遠藤大介, 単語間の距離を利用したクラスター分析による次元圧縮, 岐阜工業高等専門学校電気情報工学科卒業研究報告, 2020.
http://www.gifu-nct.ac.jp/elec/deguchi/sotsuron/endo_ad2.pdf