特 別 研 究 報 告 題 目

情報量最大化クラスタリングによる
次元削減を用いた文書の類似度計算

Document Similarity Calculation Using Dimensionality Reduction by Invariant Information Clustering

主　　査　　出口 利憲　教授

副　　査　　堀内 咲江　講師

岐 阜 工 業 高 等 専 門 学 校　　専攻科　先端融合開発専攻

2024Y14 相撲　美月

令 和 8 年（ 2 0 2 6 年 ）　　1 月 2 8 日 提 出

Abstract

In this study, an unsupervised learning clustering method is proposed, utilizing Word2Vec and Invariant Information Clustering. This method is an unsupervised learning method in which a neural network is trained to maximize the mutual information of a word vector and a vector that is a slightly transformed from it at random. To verify the effectiveness of this method, experiments were conducted to calculate document similarity using the following four dimensionality reduction method.

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

Methods 2 and 3 are dimensionality reduction techniques proposed in previous research. The text data used in this study are articles from Livedoor News. The results obtained by each dimensionality reduction method were visualized using a dendrogram. To evaluate the clustering results, the number of words contained in each cluster was analyzed and visualized using WordCloud. In addition, to evaluate similarity calculations, document classification was performed and the effectiveness of each method was evaluated by comparing classification accuracy.

Experimental results confirmed the effectiveness of clustering using Invariant Information Clustering over traditional dimensionality reduction methods.

目 次

第1章　序論

急速な情報化により膨大なデータが活用される一方、SNS の普及で信頼性の低い情報も拡散し、情報リテラシーが重要視されている。そのため、統計学や人工知能を活用したデータマイニングが注目されており、特に自然言語処理を活用したテキストマイニングは、文章から法則性や相互関係を見出す技術として活用されている。しかし言語の意味を正確に捉える難しさからさらなる研究が必要とされている。このテキストマイニングにおける文書分類という分野では、文書間の類似度計算を効率化するために次元削除が用いられる。特徴ベクトルの次元を減らすことで、計算量を抑えつつ、データの意味を保ちながら新たな特徴量を抽出できる。従来の次元削減手法である潜在的意味解析は、単語の出現回数などに基づき次元を削減する手法であるが、単語の意味が十分に考慮されないという課題がある。これに対し、Word2Vec とクラスタ分析を組み合わせ、単語の意味を考慮した次元削減が提案されている。そこで 2019 年に、クラスタ分析における、クラスタが一つにまとまる又は消失する現象と、クラスタの特徴量が他のクラスタと類似する現象を解決する教師なし学習手法として情報量最大化クラスタリング（IIC）が新たなクラスタ分析手法として提案された。特に IIC は画像分類の分野で顕著な成果を挙げており、ランダムな変換を加えた画像ペアを用いて、そのペアの確率変数の相互依存の尺度（相互情報量）を最大化するようにニューラルネットワークを学習させる。本研究では、この手法を単語のクラスタリングに応用することで、単語の意味を考慮して次元削減し、文書間の類似度を計算する。

本研究の目的は、文書間の類似度計算における次元削減において、新たな手法として IIC の利用を提案・実装し、その有用性について検討することである。また従来の次元削減手法との比較を行い、提案手法の有効性を確認する。テキストデータの対象には、株式会社ライブドアが提供する Livedoor ニュースに現在掲載されている記事と、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。前者は「スポーツ」「食べ物」「教育」「産業」「気象」「海外」の 5 つのジャンルからいくつか記事を抽出したものを使用し、後者は 9 種類のニュース記事が計 7367 本収載されており、各ジャンルからニュースを取得し使用した。これらはジャンルに分かれているため、類似した単語の評価がしやすいと考えた。

実験では情報量最大化クラスタリングと、潜在的意味解析（LSA）、ward 法での階層

的クラスタリング、K-means 法での非階層的クラスタリングを用いた場合の次元削減の結果を比較する。結果はデンドログラムにて可視化する。クラスタリングの際のクラスタの生成結果の評価のため、各クラスタの単語数と WordCloud による可視化を行い、結果を検証する。その後類似度計算の評価のため、文書分類を行い、正解率を比較し検証する。また使用した文章、クラスタ数による結果についても比較し、検証する。

第2章　自然言語処理

2.1　自然言語処理手法

2.1.1　データマイニング

データマイニングとは、データベースに蓄積された大量のデータから統計学や人工知能によって、情報の傾向を見出す技術である。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。[1)]

2.1.2　テキストマイニング

テキストマイニングとはデータマイニングの一種であり、対象が文書データに限られている。自然言語解析の手法を使って、文章を単語（名詞、動詞、形容詞等）に分割し、それらの出現頻度や相関関係を分析することで有益な情報を抽出する。ビッグデータの活用においても、テキストマイニングは非常に重要な要素となる。

2.1.3　形態素解析

形態素解析とは、自然言語処理（NLP）の一部であり、自然言語で書かれている文を、文法や品詞の情報をもとに形態素に分解し、一つ一つの品詞や変化などを判別していくことである。これにより単語の出現頻度の計算や特定の品詞のみを抽出するといった処理が可能となる。「形態素」は言語学の用語であり、意味を持つ表現要素の最小単位のことである。

テキストマイニングにおいて形態素解析が行われる理由は、多くのテキストマイニングでは単語を入力値として与えて処理するため、日本語は英語とは異なり、文章がどこで区切れるか分かりにくいためである。形態素解析を行うためのツールは形態素解析器と呼ばれ、いくつかの形態素解析器はオープンソースで公開されている。本研究では MeCab というオープンソースのソフトウェアを使用した。

2.2　自然言語

2.2.1　自然言語と人工言語

自然言語は、人間が日常的に使用する、意思疎通を行うための言語である。例として、日本語や英語、中国語などが挙げられる。対して人工言語は人間がコンピュータに処理

させるための言語である。例として、C 言語や Python といったプログラミング言語などが挙げられる。自然言語においては、同一の文章でも読み手によって複数の解釈がある場合、伝える際に曖昧な表現を使用しても、相手が解釈することで伝わる場合がある。しかし人工言語では、解釈は一通りしか認められない。命令に対して複数の解釈があると、コンピュータの処理が毎回変化してしまうのを防ぐ為である。

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である。

2.2.2 自然言語の曖昧性

自然言語の曖昧性（Ambiguity）とは、言語表現や文脈が複数の解釈や意味を持つ状態を指す。言葉や文章が一意に解釈できない状態が生じることで、曖昧性が発生する。

曖昧性には多義性と類義性の二つの種類がある。多義性とは、ある単語が複数の意味を持っており、可能な解釈が広がること、すなわち単語の意味が完全に一つに定義できないことを指す。多義性の例として「潰れる」という単語を上げる。「箱が潰れる」という文では、外部からの力を受けて、もとの形が崩れることを表しているが、「あの店は潰れてしまった」では経営・生活などが成り立ってゆかなくなるという意味で使われている。類義性では異なる単語が同じ意味を示す性質のことを指す。「用意」と「準備」といったような、読み書きが異なる場合においても似た意味を持つ単語が例として挙げられる。

自然言語の曖昧性は、言語理解や機械翻訳、情報検索、テキストマイニングなどの分野で課題となっている。これを解決するためには、文脈や周囲の情報を考慮して意味を推測する手法やアルゴリズムが必要とされる。

第3章　学習

3.1　機械学習

3.1.1　機械学習とは

機械学習とは、Machine Learningとも呼ばれ、コンピュータに大量のデータを読み込ませ、一定のルールやパターンを見つけ出し学習させることで、同じような課題に直面した際に、以前学習したルールやパターンを用いることで、良い推測や判断を行うことができるデータ解析技術である。機械学習を支える技術の一つがニューラルネットワークである。またこのニューラルネットワークの学習能力を高める一つの手法がディープラーニングである。

3.1.2　教師あり学習

学習手法のうち最も代表的なものが教師あり学習である。人間が事前に正解のデータ（ラベル）を入力し、その正解のデータと比較して正しいかどうかを判断させる。コンピュータにとって判断基準が明確であるため、「正しいか、間違っているか」の問題を解決するのに適している。教師あり学習のアルゴリズムで代表的なものは、「回帰」と「分類」である。

3.1.3　教師なし学習

教師なし学習は、教師あり学習とは異なり、正解のデータを教えずに学習を行う。大量のデータを学習させることでデータの特徴やパターンを学習する。データ内に存在する未知のパターンを見つけたいときに適している。教師なし学習のアルゴリズムで代表的なものは、クラスタリングである。[2)]

3.1.4　特徴量

機械学習の活用のうえで重要な概念の一つが特徴量である。特徴量とは対象となるデータの特徴を数値にして表したものである。自然言語処理の場合、テキストデータにおける特徴量とは、ある単語の出現頻度や重要度の指数、単語を数値化したベクトルなどのことを指している。これらは非構造化データとも呼ばれ、エクセルといった「列」と「行」の概念を持つ構造化データに比べて、取得・解釈・利用が難しい。近年のインターネット

のさらなる普及により、これらの非構造化データを上手く活用する技術が重要度を増している。そこで非構造化データには、ニューラルネットワークを適用することが多い。[3)]

3.2　ニューラルネットワーク

ニューラルネットワークとは、人間の脳内にある神経細胞（ニューロン）とそのつながり、つまり神経回路網を人工ニューロンという数式的なモデルで表現したものである。ニューロンは信号を受け取った後、次へ情報を伝えるためにシナプスを作る。その際シナプスの結合強度によって、情報の伝わりやすさが変わる。

ニューラルネットワークで、伝達された情報は、Figure 3.1 に示す入力層、隠れ層、出力層の順に処理される。入力層は、人工ニューロンが最初に数値の情報を受け取る層のことである。その後受け取った情報を、次の隠れ層に転送する。隠れ層は中間層とも呼ばれ、入力層から情報を受け継ぎ、取り込んだ複雑なデータを選別し、学習によって扱いやすい状態に変換する層である。隠れ層の数は決まりがなく、層が多い程、複雑な分析が可能となり、隠れ層が３層以上あるニューラルネットワークを用いた機械学習の手法やその周辺の研究領域のことをディープラーニングと呼ぶ。出力層では、隠れ層で処理された信号が送られ、隠れ層での計算の最終結果を伝達する。あるニューロンから次のニューロンへの出力過程において、入力された数値を特定の方法で変換し、その結果を出力する関数を活性化関数と呼ぶ。ニューロンや中間層が増えるほど、分析の柔軟性や結果の表現力は向上する反面、データ量やパラメータ数の増加により、メモリ使用量や演算量は増加する [4)]。

活性化関数にはシグモイド関数、ソフトマックス関数、ReLU 関数などがある。例としてソフトマックス関数を挙げる。これは入力データ内の複数の値を 0.0〜1.0 の範囲の確率値に変換する関数であり、データがどのクラスに属するか判断するような分類問題に用いられる。この関数によって出力される複数の値の合計は常に 1.0 となる。ソフトマックス関数はニューラルネットワークにおいて、入力値ベクトルの各ベクトルに対する確率値に相当する出力値ベクトルに変換する役割を持っている。ここでソフトマックス関数は以下に示す式 (3.1) で表すことができる。y_i は i 番目の出力、x_i は i 番目の入力のことである。分母はすべての入力信号の指数関数の和、n はデータ数を表す。[5),6)]

$$y_i = \frac{e^{x_i}}{\sum_{k=1}^{n} e^{x_k}} \tag{3.1}$$

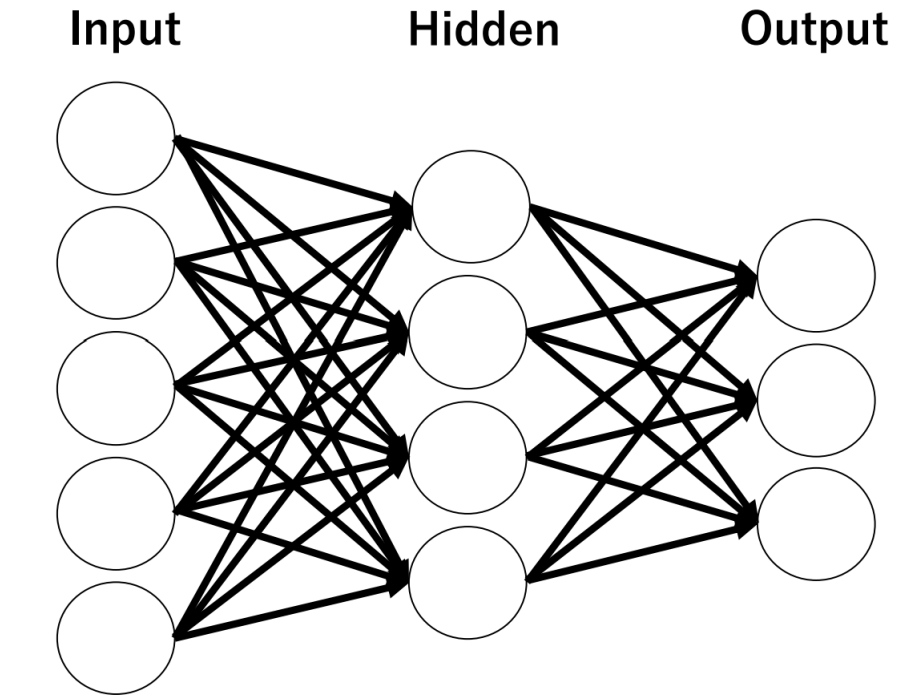


Figure 3.1: Network layer.

第4章　実験で使用した技術・手法

4.1　単語のベクトル化と類似度計算

4.1.1　分散表現

分散表現とは単語を高次元のベクトルとして表す技術である。コンピュータが演算できるのは数値のみであるため、単語の意味という概念的な要素を数値に置換することで、意味概念を計算することが可能になる。分散表現を効率的に活用するためのツールとして Word2Vec が挙げられる。

4.1.2　Word2Vec

Word2Vec とは、ニューラルネットワークの重み学習を利用した単語の意味をベクトル表現化する手法である。2013 年に Google のトマス・ミコロフ氏らによって開発・公開された。Word2Vec を利用して単語をベクトル化すると、次のような計算ができる。

- 単語同士の類似度計算
- 単語同士の加算・減算

具体例について以下の式 (4.1) を用いて説明する。Word2Vec によって生成されたベクトル空間上には「king」、「man」、「queen」、「woman」という単語が存在するとする。これらの単語はベクトルとして実際に値を持っているため、単語同士で以下のような計算をすることができる。

$$「\text{king}」 - 「\text{mam}」 + 「\text{woman}」 = 「\text{queen}」 \tag{4.1}$$

式 (4.1) は空間上にある単語の足し引きによって導出されているため、ほかにも近い意味のものが存在すればいくつか導出することも可能となる。こうした操作は、Word2Vec による学習を済ませたモデルを用いることで、実際に行うことができる。[2)]

4.1.3　Word2Vec による単語のベクトル化

Word2Vec では、学習済みモデルに対して単語を指定することで、指定した単語のベクトル表現を取得する事が出来る。これにより、単語がベクトル空間上のどこの位置しているのかを数値として受け取ることが可能となる。数値型であるため、COS 類似度を用いた単語間の類似度計算や、単語間の距離をもとにしたクラスタ分析が可能となる。

4.1.4 TF-IDF

TF-IDF とは、文書内の単語の重要度を示す手法の一つである。TF（Term Frequency）と IDF（Inverse Document Frequency）の二つの指標を基に計算され、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 とは、文書における単語の出現頻度を表す値である。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ため、重要度は大きくなる。IDF は、単語の逆文書頻度と呼ばれ、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を特徴づけする重要な単語とみることが出来るため、IDF 値は大きくなる。TF-IDF はこの TF と IDF の積で求められる。これらの導出式はいくつかあるが、本研究では下記の式 (4.2)〜(4.4) で計算を行った。

$$tf_{i,j} = \text{文書 } d_j \text{での単語 } w_i \text{の出現回数} \tag{4.2}$$

$$idf_i = \log \frac{1 + \text{全文書数}}{1 + \text{単語 } w_i \text{が出現する文書数}} + 1 \tag{4.3}$$

$$tfidf_{i,j} = tf_{i,j} \times idf_i \tag{4.4}$$

4.1.5 COS 類似度

COS 類似度とは、二つのベクトルの類似性を表す指標である。ベクトル間の COS 値を求めることで、ベクトル同士がどの程度同じ方向を向いているかが求められる。COS 類似度の値は、1 に近いほどベクトル同士が類似しており、0 に近いほど類似していないことを示す。本研究では、ある二つの単語ベクトルのなす角度の近さ、すなわち単語の類似度を求めるために使用した。

2 つのベクトルを $\vec{a}$ と $\vec{b}$ とすると、COS 類似度をもとにした距離は以下の式 (4.5) で導出される。[2)]

$$\cos(\vec{a}, \vec{b}) = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}||\vec{b}|} \tag{4.5}$$

4.2　提案手法

4.2.1　情報量最大化クラスタリング（IIC）の位置づけ

情報量最大化クラスタリング（IIC）は、教師なし学習におけるクラスタリング性能の向上を目的として提案された手法である。近年、多くの場面で非常に優れた性能を発揮する深層学習手法が使用され始めている。しかし深層学習モデルを実際に学習する場合、大量のラベル付きデータが必要となる。これが深層学習の実応用を制限している。そこでラベルを必要としない教師なし学習手法が注目されている。ここで教師なし学習はデータのみから特徴量を上手く得ることが重要である。教師なし学習の代表的なものとしてK-means 法などのクラスタリングが挙げられる。しかし従来の教師なし学習では、クラスタの特徴量が他のクラスタと似通ること、学習データのノイズに左右されることにより、クラスタが一つにまとまってしまう、本来あるべきクラスタが消失するという問題が指摘されている。上記に述べた問題点を解消する手法として情報量最大化クラスタリング（IIC）が 2019 年に提案された。半教師あり学習に IIC を適応した場合、教師あり学習の精度を超えるという結果も報告されている。[7)]

4.2.2　情報量最大化クラスタリング（IIC）の概要

情報量最大化クラスタリング（IIC）は、正解ラベルを必要としない教師なし学習手法である。IIC は主に画像分類の分野で成果を上げており、ある元画像に一般的なランダム変換を加えたペアとなる画像を作成し、このペアの相互情報量を最大化するようにネットワークを学習させる。IIC は汎用的な手法であり、データの相互情報量が計算できれば、様々なタスクに使用することができる。

IIC の大きな特徴は、相互情報量の最大化とオーバークラスタリングの二つである。[7)]

4.2.3　相互情報量

相互情報量とは二つの確率変数の相互依存の尺度である。数字が大きければ大きいほど、二つの情報が影響を及ぼしているため、一つの情報を見ただけでもう一つの情報を判別しやすい。元の画像とノイズを加えた画像は、同じオブジェクトを含む異なる画像である。元の画像と、ペアとなる画像の共通点を探すことで、オブジェクトの表現できる特徴量が得られる。

ここで相互情報量は以下に示す式 (4.6) で定義することができる。

$$I(X;Y) = \sum_{x \in D_X} \sum_{y \in D_Y} P_{X,Y}(x,y) \log \frac{P_{X,Y}(x,y)}{P_X(x) P_Y(y)} \tag{4.6}$$

ここで $P_{X,Y}$ は同時分布確率、$P_X(x)$ と $P_Y(y)$ はそれぞれ X と Y 周辺確率分布関数である。

本研究では、ニューラルネットワークに単語ベクトルを入力する。次に、単語ベクトルをランダムに少し変換したベクトルも入力する。これらの入力から得られる出力ベクトル間の相互情報量を計算し、それが最大となるように学習を行う。[8)]

4.2.4 オーバークラスタリング

オーバークラスタリングは学習時のみに全結合層の最終層の数を増やす手法のことである。テスト時には、学習時に増やした余分なクラスタ出力は使用されないが、正解クラスタ数よりもクラスタ数が多くなるように学習することで、ノイズの影響と未知のクラスタに対処できるようにする。

4.3 次元削減

4.3.1 潜在的意味解析（Latent Semantic Analysis;LSA）

LSA は、文書データには潜在的なトピックが存在すると推定し、そのトピック数まで次元を削減することで、分類を効果的に行う技法である。文書行列から特異値分解（SVD）を使い、「トピック数 × 文書数」の行列を抽出して、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。

ここで特異値分解とは、任意の実行列が二つの直交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。以下に特異値分解の式 (4.7) を示す。

$$TD = U \Sigma V^{\top} \tag{4.7}$$

この式における U,Σ,$V^{\top}$ はそれぞれ行列を表しており、右辺は文書行列 TD を 3 つの積で表したものである。U は左特異 (ターム) ベクトル、Σ は特異値を含むベクトル、$V^{\top}$ は右特異 (文書) ベクトルと呼ばれる。左特異ベクトルは、情報が重要なものから並んでいるため、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。

4.3.2 クラスタ分析

クラスタ分析とは、異なるものが混ざり合う集団から、似た性質を持つ集団クラスタに分類する手法のことである。教師なし学習の分類手法であり、データの傾向をつかむことができる。形や色などで分類する場合とは違い、クラスタ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。

クラスタ分析には分類が階層的になる階層的クラスタ分析と、あらかじめクラスタ数を指定して分類する非階層的クラスタ分析がある。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に分類していく手法である。対象となる二つの距離を計算し、その距離が近い者同士から順にクラスタを形成していく。階層的クラスタ分析は、クラスタ間の距離測定の方法にいくつか種類があり、対象データに最も適したものを選択する。本研究では ward 法を採用した。ward 法は二つのクラスタを融合した際に、同クラスタ内の分散と他クラスタ間の分散の比を最大化するようにクラスタを形成していく方法である。

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に集まるよう分割する手法である。クラスタ数を事前に計算することができず、また最適なクラスタ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 法が挙げられる。K-means 法とは、クラスタ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。K-means 法はクラスタ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い。

4.4 Python

4.4.1 Python とは

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。

4.4.2 Google Colaboratory

Google Colaboratory とは Google が提供するサービスであり、ブラウザから Python を実行できる。機械学習で必要な外部ライブラリ（NumPy など）もインストール済みであるため、簡単に実行環境を構築できる。また環境構築なしで、GPU を使用することができる。GPU とは Graphics Processing Unit の略称である。GPU は大量の演算を並列かつ高速に処理することができる性質を持っている。

4.4.3 MeCab

MeCab とは、京都大学と日本電信電話株式会社 (NTT) が共同開発したオープンソースの形態素解析エンジンのことである。MeCab は日本語に対応しており、日本で使用される形態素解析エンジンの中でもメジャーである。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。MeCab では、品詞等の情報が記録された辞書を用意し、形態素解析を行うことができる。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。[9)]

「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

上記の結果より、MeCab によって文は意味を持つ最小単位である形態素に分割され、それぞれに品詞や標準形が付与されていることが分かる。例えば、「すもも」や「もも」は名詞として解析され、「も」や「の」は助詞として適切に分類されている。このように、語の繰り返しを含む文に対しても、文脈に応じた形態素分割と品詞付与が行われていることが確認できる。

本研究では、Python から MeCab を呼ぶ出すことで形態素解析を行った。MeCab の辞書には、標準の IPA を導入した。

4.4.4 TF-IDF の計算

Python では、scikit-learn ライブラリに用意されている TfidfVectorizer 関数を利用して TF-IDF の計算が行える。TfidfVectorizer では、文字列のリストを入力として与え、いくつかのオプションを指定することで式 (4.2)、式 (4.3) の通りに TF-IDF が導出される。

4.4.5 PyTorch

PyTorch は Facebook 社が開発した Python 向けのオープンソース機械学習ライブラリである。PyTorch は可読性の高さやデバックのしやすさで近年大きく人気を伸ばしているライブラリである。PyTorch は NumPy に近い操作性を持ち、「Define-by-Run」型の動的な計算グラフで設計されているため、柔軟性が高く、複雑なネットワークであっても比較的容易に実装することが可能である。また GPU を使用できるため、大規模なシステムやデータセット、複雑なモデル構築にも対応することができる。本研究では、GPU での計算、多次元配列を扱うためのデータ構造である Tensor 型への変換、ニューラルネットワーク構築の際のデータ構造やレイヤーを定義する活性化関数や損失関数の定義、パラメータ最適化アルゴリズムの実装などに使用した。[10)]

4.4.6 Gensim

Gensim は自然言語処理に用いられる様々なトピックモデルを実装した 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、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。

4.4.7 SciPy

Scipy は Python のための数値解析ソフトウェアであり、Numpy に基づいた機能を有している。今回の実験においてクラスタ分析の過程で使用する階層的クラスタリングはこの SciPy の ward 法の linkage を用いて行う。

Figure 4.1: WordCloud.

4.5 クラスタの生成結果の評価

4.5.1 各クラスタの単語数

単語集合を IIC を用いてクラスタリングし、次元を削減した場合の、単語の分類結果を評価するために、各クラスタに分類された単語の総数を調査し、表にまとめた。また比較対象として、階層的クラスタ分析、K-means 法によって得たクラスタでも単語数を調査した。

4.5.2 WordCloud による単語集合の表現

WordCloud とは、文章中の単語の出現頻度に応じて単語を視覚的に図示する手法である。本来この手法は、文章中の単語の出現回数が多い程、その文字が強調されて表現されるため、その文章内の単語構成を視覚的に表現できる手法である。また大きさや色彩、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。WordCloud の例を Figure 4.1 に示す。[11)]

WordCloud を作る際、generate 関数にある文章を渡すと、その中のある単語の出現回数でサイズを決める。しかし実際には、出現回数ではなく、他の別の要素で文字のサイズを調整したい場合がある。例として、TF-IDF で重みを付けた値を使いたい場合や、トピックモデルのトピック別出現確率のようなもともと割合で与えられたデータを使いたい場合などである。このような場合、“単語:頻度”の辞書（dict）を作成し、generate_from_frequencies

関数（もしくは fit_words）に渡すと実行することができる。本研究では、IIC によって指定したクラスタ数へ単語を分類している。このクラスタリングの際にそのクラスタになる確率（推定確率）を単語頻度として使用し、“ 単語：推定確率 ” という辞書を作成することによって、分類された単語が、同一クラスタ内の他の単語とどれほど類似しているかを表す指標としている。

また比較対象として、LSA、階層的クラスタ分析、K-means 法によって得たクラスタでも WordCloud を作成している。階層的クラスタ分析、K-means 法の場合は、単語ベクトルとその単語が属するクラスタの中心座標の COS 類似度で数値化し、“ 単語：COS 類似度 ” という辞書を作成し、より類似している単語を強調している。LSA の場合、SVD を実行した際の $V^\top$ には「トピック − 単語行列」が格納される。これは各トピックの内容を把握できる行列であるが、各単語ごとに 0 以上のトピックの数値のみを残し、それ以外を 0 とする。この値により “ 単語：数値 ” という辞書を作成し、より類似している単語として強調している。これらの手法によって、各クラスタに含まれる単語集合を WordCloud として可視化し、意味的に関連性の高い単語が同一クラスタ内で強調されているかを確認することで、クラスタ生成結果の妥当性を検証している。

4.6 次元削減行列による文書間類似度計算の評価

4.6.1 デンドログラム

階層的クラスタリングによって形成されたクラスタの構造は、Figure 4.2 に示すデンドログラム（樹形図）によって視覚的に判断が可能となる。デンドログラムにおいて、図の末端の方で結合しているデータほど類似性の高い関係であるといえる。デンドログラムの横に閾値で直線を引くことで、指定数のクラスタに分類することができる。その例を Figure 4.3 に示す。Figure 4.3 では、破線で示した距離の閾値以下で結合された要素を同一クラスタとして扱うことで、クラスタ分割が行われている。

4.6.2 正解率

今回は、次元削減した行列に対して COS 類似度を用いて、各文書間の類似度計算を行った結果の可視化に、デンドログラムを用いた。

文書分類での分析の評価方法として、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。正が正と判断されたものと、誤が誤と判断

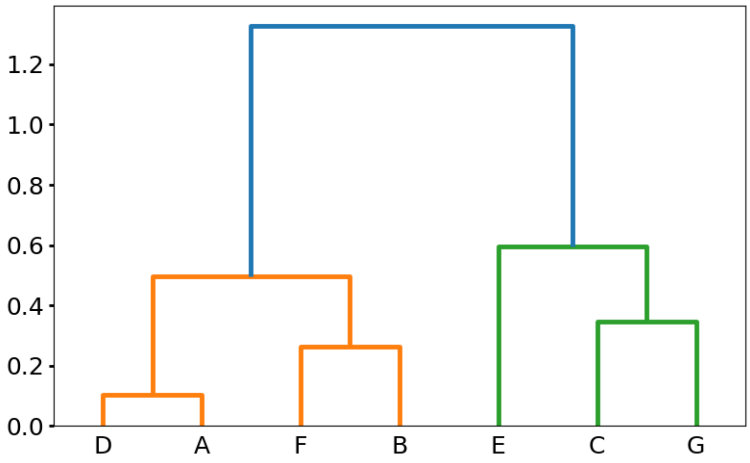


Figure 4.2: Dendrogram.

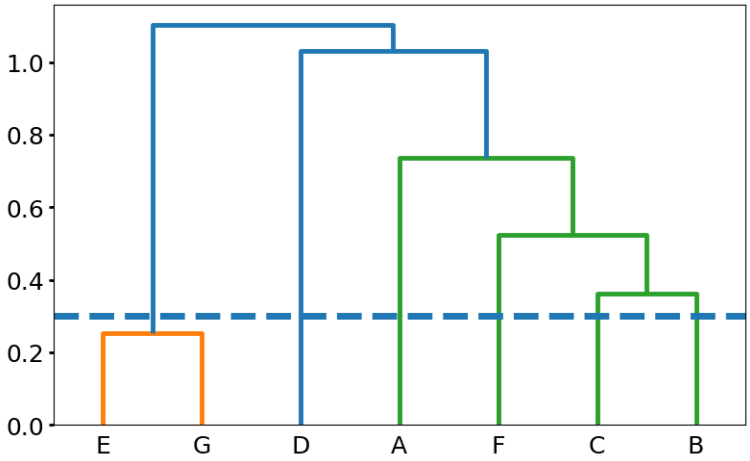


Figure 4.3: Dendrogram with a threshold line.

された物の割合を表す値であり、二値分類問題の評価方法の一つである。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 を式 (4.8) のように定義して、文書分類に適用する。

$$Acc = \frac{\text{予想と正解が等しいサンプルの数}}{\text{サンプルの総数}} \tag{4.8}$$

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を作成するため、クラスタ自身にラベルが付与されない。そのため、対象が正解のクラスタに分類されたかを判断するには、同じクラスタに属している他対象と比べる必要がある。

そのため本研究では、ジャンル数分に分割されたクラスタに対して、ラベルの割り当てを総当たりで調べ、最大の正解率を算出し、分類の評価とする。

第5章　実験

5.1　実験の概要

本実験では、Livedoor ニュースの記事を対象として、以下の四種類の次元削減法で文書の類似度計算を行う。

- 潜在的意味解析（LSA）
- Word2Vec ＋階層的クラスタリング（ward 法）
- Word2Vec ＋非階層的クラスタリング (K-means 法)
- Word2Vec ＋情報量最大化クラスタリング (IIC)

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、二つ目と三つ目の次元削減手法は従来の研究で用いられた手法で、ward 法を用いた階層的クラスタリングと、K-means 法を用いた非階層的クラスタリングによる方法である。これらと提案手法である情報量最大化クラスタリングを比較することで、提案手法の優位性を検証する。

実験では TF-IDF 値を用いて、文書行列を作成する。また単語のベクトル表現に対して IIC を行い、各クラスタへ属する確率値（帰属度）が最大のクラスタに単語を分類することにより「「単語数×クラスタ数」」の行列を作成する。比較手法として、Ward 法による階層的クラスタリングおよび K-means 法による非階層的クラスタリングを用い、同様に単語の分類を行う。クラスタリングした際の結果を比較するため、WordClouds にて単語の分類結果を可視化し、各クラスタに分類された単語数を調査した。

これらによって得た行列から、「文書数×クラスタ数」へ次元削減を行う。この行列に対して COS 類似度を用いて、各文書間の類似度計算を行う。これらの結果はデンドログラムによって可視化する。類似度計算の評価のため、文書分類を行う。類似度を基に ward 法を指定した階層的クラスタリングをもとに、実験パターンごとに割り当てられたジャンル数分までクラスタの分割を行う。分割されたクラスタに対してラベルの割り当てを総当たりで調べ、最大の正解率を算出する。加えて、LSA による次元削減についても、WordCloud による可視化、デンドログラムおよび正解率による評価を行う。

この比較により、IIC を用いた次元削減が従来手法と比べ、どのように機能しているかを確認することが本実験の最終的な目的である。

実験の文書データは、Livedoor ニュースに現在掲載されている記事と Livedoor ニュースコーパスから取得した [12),13)]。文書データを二つから取得し、比較することで、文書

による結果の大きな相違がない事を検証するためである。 実験では、分析対象であるニュースのジャンル数、記事数を変化させ、結果を出力した。これは、ジャンル数や作品数といったテキストデータの情報量に変化をつけることで結果にどのような影響が及ぼされるか確認するためである。扱った文書データは以下に示すとおりである。

Livedoor ニュースに現在掲載されている記事

- 実験パターン 1

  ジャンル数：5 作品数：10（各ジャンル 2 作品）
- 実験パターン 2

  ジャンル数：5 作品数：15（各ジャンル 3 作品）
- 実験パターン 3

  ジャンル数：5 作品数：20（各ジャンル 4 作品）
- 実験パターン 4

  ジャンル数：5 作品数：25（各ジャンル 5 作品）
- 実験パターン 5

  ジャンル数：6 作品数：12（各ジャンル 2 作品）

Livedoor ニュースコーパス

- 実験パターン 6

  ジャンル数：9 作品数：18（各ジャンル 2 作品）
- 実験パターン 7

  ジャンル数：9 作品数：27（各ジャンル 3 作品）
- 実験パターン 8

  ジャンル数：4 作品数：12（各ジャンル 3 作品）
- 実験パターン 9

  ジャンル数：4 作品数：20（各ジャンル 5 作品）

本実験では、行った実験パターンが非常に多いため、結果の比較が行いやすいデータに注目して検証する。

5.2 実験準備

5.2.1 実験環境の構築

本実験の環境を構築するために、以下に示す項目を行った。

- Python, MeCab の導入
- Word2Vec の学習済みモデルの取得
- ニュース記事の収集
- Livedoor ニュースコーパスからの文書データの取得

5.2.2 MeCab の導入

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ある。

5.2.3 Word2Vec の学習済みモデルの取得

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。しかし学習には膨大な時間を必要とする。そのため、本実験では配布されている学習済みモデルを取得し、使用することとした。取得した Word2Vec の学習済みモデルは東北大学の乾、岡崎研究室にて作られた「日本語 Wikipedia エンティティベクトル」というモデルである。このモデルは、人名や地名といった固有表現の情報も含めた上でモデルを作成するために、日本語 Wikipedia の全記事本文から学習が行われている。本実験では、2017 年に学習された 200 次元のモデルを gensim で読み込んで使用した [14]。

5.2.4 テキストデータ取得

今回文章分類の対象となるデータは、Livedoor ニュースに現在掲載されている記事と Livedoor ニュースコーパスの二つから取得した。

一つ目は株式会社ライブドアが提供する livedoor ニュースで現在掲載されている文書であり、以下の６つのジャンルから複数のニュースを任意に集めた。

- スポーツ（野球）
- 食べ物
- 産業
- 教育
- 気象
- 海外

二つ目は Livedoor のニュースとして以前掲載されていた文書であり、株式会社ロン

ウィットが収集し配布したデータである。この文書データは可能な限り HTML タグを取り除いて作成したものであり、９ジャンルに分類されている。そのうち、以下に示す７ジャンルからニュースを複数個取得し、テキストデータとしている。

- dokujo-tsushin
- it-life-hack
- kaden-channel
- livedoor-homme
- movie-enter
- smax
- sports-watch

今回の実験の目的は似ている単語集合のクラスタを生成することであるため、対象となる文書データにジャンルが存在すると類似度の高い単語が多く含まれていることで、クラスタリングが容易になり、評価しやすいと考え、これらの文書データを用いる。

5.2.5　テキストデータの形態素解析

取得・収集した文章に対して、MeCab を使用することで形態素解析を行った。形態素解析後は必要な品詞のみを残す作業を行うよう設定した。本実験では、いくつかのニュースから単語を大量に抽出し、その単語同士の類似度を調べたいため、動詞や形容詞などを含めると、結果が評価しづらくなると考え、名詞のみを抽出するように設定し、それ以外の単語は削除した。

また MeCab では数値も名詞として抽出されるが、本研究の特性上必要ないため、削除した。

これらの作業を行った結果、今回実験に使用した各文書データに含まれる単語の種類は以下の数となることが分かった。

- 実験パターン 1（単語数）･･･ 3541 語
- 実験パターン 2（単語数）･･･ 5241 語
- 実験パターン 3（単語数）･･･ 6659 語
- 実験パターン 4（単語数）･･･ 7883 語
- 実験パターン 5（単語数）･･･ 3884 語
- 実験パターン 6（単語数）･･･ 4733 語

- 実験パターン 7（単語数）… 3108 語
- 実験パターン 8（単語数）… 2285 語
- 実験パターン 9（単語数）… 4030 語

さらに文章中には重複する単語が多く存在する。重複する単語は同一ベクトルであるため削除した。

5.2.6 Word2Vec による単語のベクトル化

単語ベクトル行列を生成するために、第 5.2.5 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 のモデルを利用して、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 200 次元のモデルを使用するため、200 次元配列を取得する。この配列を列方向に結合していき、「単語数× 200」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ことがある。そのため本実験では単語ベクトルが存在する単語のみベクトル化を行った。

5.3 情報量最大化クラスタリング

5.3.1 PyTorch のデータセット作成

前節で生成した配列をもとに、学習に用いる PyTorch のデータセットを作成する。

まずデータセットをテンソル化する前処理を定義する。前項で作成された単語ベクトル配列は Numpy の ndarray 型であるため、torch.Tensor 関数によりテンソル型へと変換するように記述する。

次に PyTorch のモジュールである torch.utils.data.Dataset を使用し、データをテンソル化してラベルと合わせて返す Dataset を定義する。親クラス「torch.utils.data.Dataset」を継承して子クラス「MyDataset」を定義する。__init__で引数の処理をしておき、__len__で引数の長さ、__getitem__でインデックス番号を指定した際に、対応する data と label を返すように定義する。本研究では教師なし学習であるため、label は必要ない。しかし torch.utils.data.Dataset を使用するため、label は np.zeros() でどのデータに対しても 0 を設定している。

最後に学習のためにデータをバッチサイズに分割してイテレータを返す DataLoader を定義する。第 1 引数は先程取得した Dataset を渡し、第 2 引数「batch_size」に 1 回の訓

練またはテスト時に一気に何個の data を使用するか、第 3 引数「shuffle」に data の参照の仕方をランダムにするかを指定する。本実験では、batch_size=128、shuffle=False を指定した。

5.3.2 IIC モデル定義

IIC のモデル定義では第 4.2.3 項で示したオーバークラスタリングを使用する。最終出力層では分類したいクラスタ数のものと、オーバークラスタリングのものを使用し、損失関数もその両方の出力を計算して、総和を使用する。オーバークラスタリングするネットワークで微細な変化を捉えることで、通常の期待したいクラス分類の性能もアップさせたいためである。

実験では OVER_CLUSTERING_RATE（オーバークラスタリング率）は 100、OUTPUT_LINEAR（ユニット数）は 400 とし、NUMBER_OF_CLUSTERS（クラスタ数）の値を適宜変化させた。モデルの構築には、PyTorch のメインパッケージである torch に定義されている nn パッケージを使用し、torch.nn.Module クラスを継承し、NetIIC クラスを作成した。クラス内では第 3.2 節で説明した層（レイヤー）の定義を__init__関数で行う。forward というメソッドで、引数としてデータ（x）を受け取り、出力層の値を出すまでのネットワーク（順伝播）を記述している。

PyTorch で全結合層を定義するためには、torch.nn.Linear を用いる。nn.Linear は入力データに線形変換を適用するクラスである。引数は（インプットされたユニット数、アウトプットするユニット数）とする。全結合層の役割は、隣接する 2 層間の全てのニューロンユニット間において、単純な「線形重みパラメータによる線形識別的な変換」である。今回全結合層は 2 層である。一層目の引数は（入力ベクトル数、ユニット数）として（200, OUTPUT_LINEAR）とした。入力ベクトル数は単語ベクトルの 200 次元ベクトルに対応させた。この OUTPUT_LINEAR は変数であり、変化を与えることによってユニット数を増加させる。活性化関数 (Activation) として ReLU 関数を採用し、正の値はそのまま出力され、負の値は 0 となる線形変換を行う。ReLU 関数は、torch.nn.functions に含まれており、こちらは慣習的に F と読み込まれる。二層目の引数は（OUTPUT_LINEAR, NUMBER_OF_CLUSTERS）とした。NUMBER_OF_CLUSTERS はクラスタ数であり、実験の際に変化を与えることで、クラスタリング結果を比較する。ここでの活性化関数には第 3.2 節で例として挙げた softmax 関数を用いて、入力値ベクトルの各要素を 0.0〜

1.0 の範囲の確率値に相当する出力値ベクトルに変換する。この関数によってデータがどのクラスに属するか判断する。同様にオーバークラスタリング用に二層目の引数を（OUTPUT_LINEAR, NUMBER_OF_CLUSTERS × OVER_CLUSTERING_RATE）にしたものを定義する。OVER_CLUSTERING_RATE はオーバークラスタリング率であり、変化を与えることによって全結合層の最終層の数を変化させる。[15] 最終的に出力層では、クラスタ数のものと、クラスタ数にオーバークラスタリング率をかけたものの二つが出力される。

5.3.3 重み、損失関数、相互情報量

モデルの重みに初期値を設定する際は torch.nn.init を使用した。torch.nn.init.normal_() は、平均 (mean) に 1、標準偏差（std）に 0.02 の正規分布から初期化するように設定した。torch.nn.init.constant_() では定数（val）を 0 に設定し、初期化した。

予想データと正解データの出力の間にどのくらい誤差があるのかを評価する損失関数を定義する。作成した予測モデルの精度を評価する際に損失関数は使われ、値が小さければ小さいほど正確なモデルである。今回は（元のベクトルデータ、少し変えた単語ベクトル）のペアを入力として受け取り、出力が少し変えた単語ベクトルとどれだけ離れているのかを計算する。相互情報量を最大化したいが、損失にするために、マイナスをかけ算して、最小化問題に置き換える。また、相互情報量の計算に係数項を加え、よりクラスがバラつきやすいようにしている。最終的な loss はクラスタリングと Overclustering の平均となり、この値で勾配計算を行う。

損失を最小限に抑えるための最適化関数を定義する。pytorch の optim というモジュールには様々なパラメータの更新手法があり、簡単に誤差逆伝播法を行ないパラメータを更新していくことができる。今回は勾配を記憶する「Momentum」と、勾配の二乗を記憶する「Adagrad」の項により構成される、torch.optim.Adam を使用する。

5.3.4 ペアの生成

IIC は教師なし学習であるため、教師ラベルは使用しない。IIC における入力には、対象のデータとデータを適当に変換したもの二つを使用する。実験では、元のベクトルデータに、全ベクトルデータの標準偏差に基づくノイズを加えて、ペアのデータを生成した。ネットワークにはそれぞれを入力し、それぞれの出力を得る。

元のデータと、ペアのデータを入力した際のそれぞれの出力の第 4.2.3 項で示した相互情報量が最大となるように学習を行う。

5.3.5 学習

学習時には model.train() を実行し、ネットワークを学習モードにする。epoch とはデータを一通り使用する 1 試行のことを意味し、今回は 1000 を設定した。pytorch のスケジューラーは epoch ごとに学習率を更新する。学習率は、scheduler の CosineAnnealingWarmRestarts を用意し、変化させている。学習率を変化させ、小さい値から急激に大きくしたときに局所解から抜け出し、大域的な極小解へとパラメータを学習させやすくする工夫である。ミニバッチ学習のため、1epoch の間に少しずつデータを使用して学習を進め、全データを一通り使用したら 1epoch 終了となる。

5.3.6 テスト（分類）

テスト時には model.eval() を実行して、テストモードに切り替える。結果を把握しやすいように、ミニバッチサイズ 1 のテスト用の DataLoader を用意しなおして、1 つずつ推論し、結果を格納する。

5.4 TF-IDF

TF-IDF の計算には Python を用いる。計算は、4.1.4 項に示したように、scikit-learn ライブラリの TfIdfVectorizer 関数に形態素解析を行ったテキストデータを引数として渡すことで行う。計算終了後、TfIdfVectorizer 関数のメソッドである get_feature_names で計算に使用された単語のリストを抽出しておく。この単語リストは出力された行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。

5.5 次元削減

5.5.1 潜在的意味解析による次元削減

第 4.3.1 項に示したように、 sklearn.decomposition.TruncatedSVD に TF-IDF 値を与えることで、潜在的意味解析による次元削減を行った。次元数は実験パターンごとに、任意に決めたクラスタ数とした。

5.5.2　階層的クラスタリングによる次元削減

クラスタ分析による次元削減は以下の手順で実行した。

1. 抽出した単語のベクトルを基に、linkage と fcluster 関数で階層的クラスタリングを行う
2. 分類されたクラスタの単語ベクトル値を基に、各クラスタの中心値を求める
3. 各単語のベクトル値と各クラスタ中心との類似度を、式 (4.5) を基に計算する
4. この類似度を、クラスタ数×単語数の行列にまとめ、分類されていないクラスタは 0 とする
5. 5.4 節にて求めた行列と、4. の行列を掛け合わせることで「文書数×クラスタ数」の行列を生成する

5.5.3　K-means 法による次元削減

K-means 法による次元削減は以下の手順で実行した。

1. 抽出した単語のベクトルを基に、scikit-learn の KMean 関数でクラスタリングする
2. 分類されたクラスタの単語ベクトル値を基に、各クラスタの中心値を求める
3. 各単語のベクトル値と各クラスタ中心との類似度を、式 (4.5) を基に計算する
4. この類似度を、クラスタ数×単語数の行列にまとめ、分類されていないクラスタは 0 とする
5. 5.4 節にて求めた行列と、4. の行列を掛け合わせることで「文書数×クラスタ数」の行列を生成する

5.5.4　IIC による次元削減

IIC を実行すると、各単語の各クラスタへ属する確率値が算出されるが、そのうち確率値が最も大きいクラスタに各単語を分類する。この値を各文書が含む単語がどのクラスタにどれほど属しているかを表す帰属度として使用し、分類したクラスタ以外の帰属度は 0 とする。これにより「単語数×クラスタ数」の行列を作成する。TF-IDF によって得た「文書数×単語数」の文書行列と、IIC によって得た「単語数×クラスタ数」の行列をかけ合わせることで、次元削減した文書行列が求められ、「文書数×クラスタ数」の行列を生成する。

5.6　クラスタの生成結果の評価

5.6.1　各クラスタの単語数

単語集合を各手法でクラスタリングした際の、クラスタの分類結果をもとに、各クラスタの単語数を調査する。結果は実験パターンごとに棒グラフにて可視化する。LSA は一つの単語が複数トピックに分類されることがあるため、今回は IIC と階層的クラスタリング、K-means 法の結果のみをまとめた。

5.6.2　WordCloud の作成

WordCloud の作成は以下の手順で行う。

1. 各手法で単語分類を行う。
2. 分類過程で生成されたクラスタを用いて、クラスタ数×単語数の行列を作成する。
3. 行列の各行を用いて、各クラスタの WordCloud を作成する。

行列の作成には、階層的クラスタリング、K-means 法ではクラスタの中心座標と各単語間の COS 類似度を使用する。各単語の 200 次元のベクトルを NumPy の mean 関数で平均化し、中心座標を求めた。WordCloud 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。情報量最大化クラスタリングでは各単語の各クラスタへ属する確率値が算出されるが、そのうち確率値が最も大きいクラスタに各単語を分類している。よってこの最大値を、各クラスタに分類した際のよりそのクラスタに類似している重要値として代用する。LSA では $V^\top$ に格納される値について、各単語ごとに 0 以下の場合を 0 に変換し代用する。それぞれ“ 単語：COS 類似度 ”の辞書、“ 単語：推定確率 ”の辞書型変数を作る。

WordCloud の作成には、Python の WordCloud ライブラリ 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、各クラスタの WordCloud を作成する。クラスタ内に単語が一つも無い場合は、no_word と表示する。

5.7　文書のクラスタ分析

5.7.1　文書間の COS 類似度

次元削減された行列に対して、pdist 関数を用いることで COS 類似度を計算した。この値は、次に行うクラスタ分析用に計算を行った。

5.7.2　クラスタ分析

前項で計算した文書間の類似度をもとにクラスタ分析を行った。SciPy の linkage を用いた階層的クラスタリングを使用する。距離の測定はユークリッド距離 (euclidien) を使用した。クラスタリング方法には第 4.3.2 項で説明を行った ward 法を指定する。linkage 関数のパラメータ method を ward にすることで ward 法でのクラスタリングとなる。

5.7.3　デンドログラムの出力

類似度の計算結果に対して、4.6.1 項にあるように、linkage 関数と dendrogram 関数を使用することで、デンドログラムを出力した。

5.8　正解率

5.8.1　クラスタ分析結果の取得

また、fcluster の引数に linkage の結果とクラスタ数を入力することで、実験パターンごとに割り当てられたジャンル数までクラスタの分割を行う。

5.8.2　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 からクラスタ数 − 1 までの整数をクラスタに割り当てる。クラスタ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。

第6章　実験結果

6.1　実験結果

6.1.1　各クラスタの単語数

単語をクラスタリングした際の、クラスタの分類結果より、各クラスタの単語数を調査した結果を、棒グラフにて可視化する。視覚化のためにカラーリングしているが、同一色のクラスタに同じジャンルの単語が含まれているとは限らない。この内容の確認は、WordCloud による可視化にて行う。

実験パターン１～５での結果を Figure 6.1(a)～6.1(e) に、実験パターン６～９での結果を Figure 6.2(a)～6.2(d) に示す。

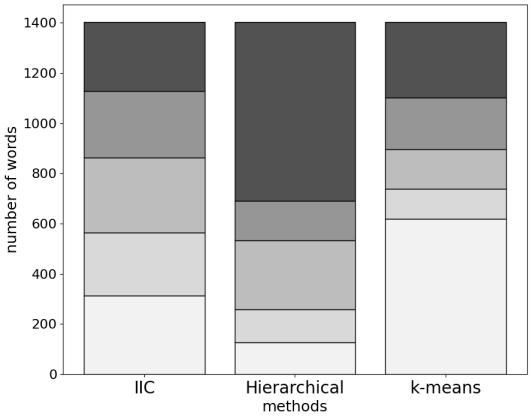
パターン１の結果である Figure 6.1(a) をみると、IIC は単語数がおおよそ均等に分類されたのに対して、階層的クラスタリング、K-means 法は一つのクラスタの単語数が多く偏りがあることが分かる。パターン３、５の結果である Figure 6.1(c), Figure 6.1(e) をみると、K-means 法では単語数が極端に少ないクラスタが見られることが分かる。全体を通して、IIC での各クラスタの単語数がおおよそ均等であり、階層的クラスタリングと K-means 法では単語数の偏りが見られることが分かる。

パターン６の結果である Figure 6.2(a) をみると、IIC は単語数がおおよそ均等に分類されたのに対して、階層的クラスタリング、K-means 法は一つのクラスタの単語数が多く偏りがあることが分かる。パターン７の結果である Figure 6.2(b) は IIC でも単語数の偏りが見えるが、階層的クラスタリングと K-means 法に比べると偏りが少ないことが分かる。全体を通して、IIC での各クラスタの単語数がおおよそ均等であり、階層的クラスタリングと K-means 法では単語数の偏りが見られることが分かる。

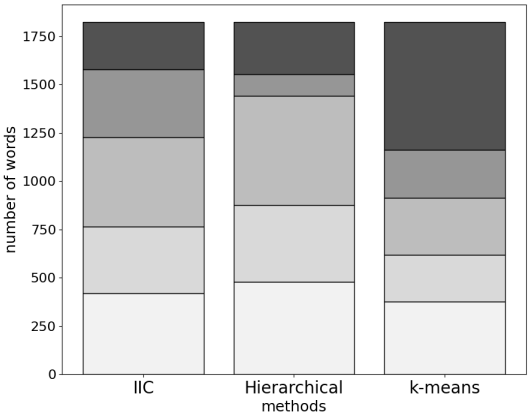
6.1.2　WordCloud

実験パターン１～５のうち、6.1.1 項にて各クラスタの単語数に大きく差があった実験パターン３について、WordCloud にてクラスタ内の単語を確認する。実験パターン３における IIC での WordCloud の結果を Figure 6.3(a)～6.3(e) に、LSA での結果を Figure 6.4(a)～6.4(e) に、階層的クラスタリングでの結果を Figure 6.5(a)～6.5(e) に、K-means 法での結果を Figure 6.6(a)～6.6(e) に示す。

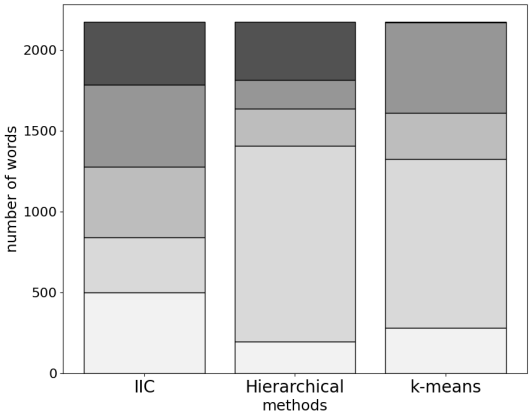
それぞれの WordCloud での結果を見ると、IIC である Figure 6.3(a) はカタカナや「セッ



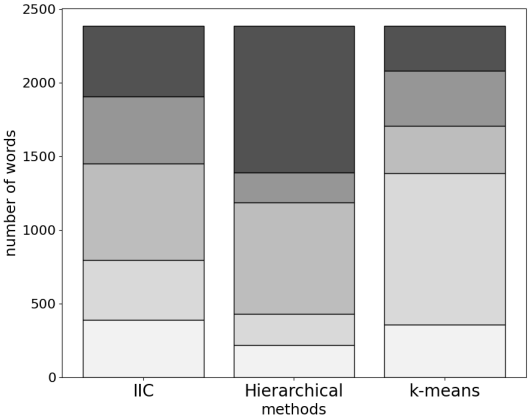
(a) Pattern 1.



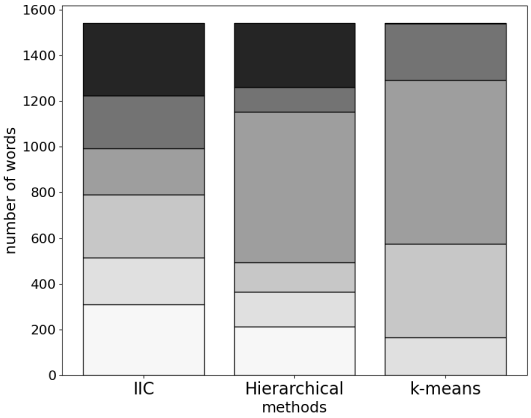
(b) Pattern 2.



(c) Pattern 3.

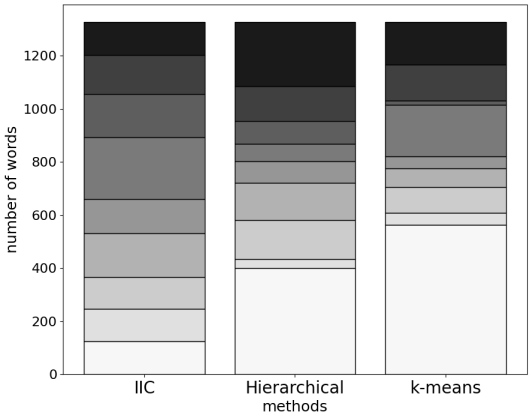


(d) Pattern 4.

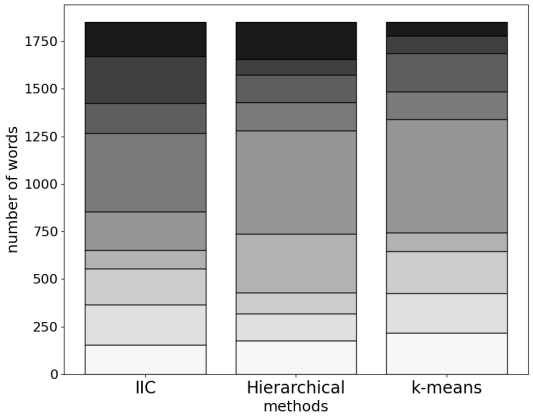


(e) Patternr 5.

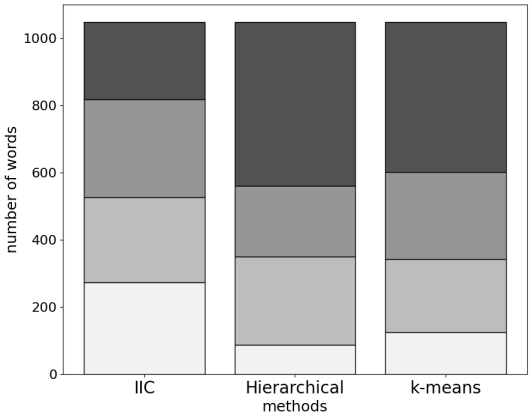
Figure 6.1: Number of words, pattern 1 to 5.



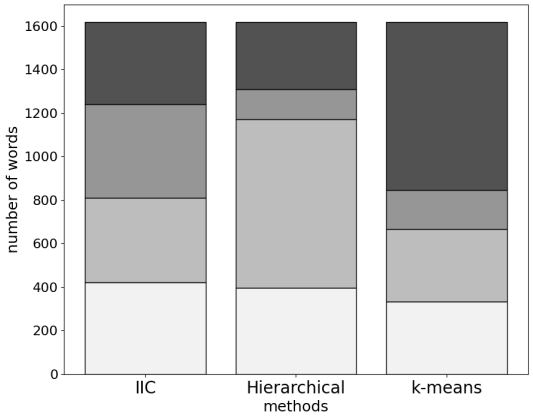
(a) Pattern 6.



(b) Pattern 7.



(c) Pattern 8.



(d) Pattern 9.

Figure 6.2: Number of words, pattern 6 to 9.

ト」「シリーズ」「ポイント」といったスポーツ関連、Figure 6.3(b) は一番強調されている単語ではないが「季節」「水分」「気温」「夜間」といった気象ニュースに用いられる単語、Figure 6.3(c) は食べ物関連、Figure 6.3(d) は強調されている単語ではないが「受賞」「研究」「日本」「平均」「先輩」といった教育関連、Figure 6.3(e) は地名といった類似した意味の単語が含まれていることがわかる。

LSA である Figure 6.4(a) と Figure 6.4(c) は産業関連で非常に類似した分布に、Figure 6.4(b) は「気温」「猛暑」「台風」といった気象ニュースに用いられる単語、Figure 6.4(d) は「英語」「授業」「教員」といった教育関連と「台風」「猛暑」といった気象関連の単語、

(a) Cluster 0.

(b) Cluster 1.

(c) Cluster 2.

(d) Cluster 3.

(e) Cluster 4.

Figure 6.3: IIC, clusters:5.

(a) Cluster 0.

(b) Cluster 1.

(c) Cluster 2.

(d) Cluster 3.

(e) Cluster 4.

Figure 6.4: LSA, clusters:5.

(a) Cluster 0.

(b) Cluster 1.

(c) Cluster 2.

(d) Cluster 3.

(e) Cluster 4.

Figure 6.5: Hierarchical Clustering, clusters:5.

(a) Cluster 0.

(b) Cluster 1.

(c) Cluster 2.

(d) Cluster 3.

(e) Cluster 4.

Figure 6.6: K-means, clusters:5.

Figure 6.3(e) は食べ物関連といった単語が含まれていることがわかる。

階層的クラスタリングである Figure 6.5(a) は強調されていない単語ではスポーツ関連、Figure 6.5(b), Figure 6.5(c) は類似度の低い単語が多く、Figure 6.5(d) は「嵐」「火災」「崩れ」といった気象関連、Figure 6.5(e) は「東芝」「アップル」「インテル」といった産業関連の単語が含まれていることが分かる。全体的に見ると、強調されている単語には類似性の低い単語が多い。

K-means 法である Figure 6.6(a)〜Figure 6.6(d) は「人気」「話題」「ピーク」や、「入社」と「所属」や、「歴史」と「ブーム」など関連している単語もあるが、全体的に類似性の低い単語が多く、一つのクラスタ内に複数のジャンルの単語が集まっていることが分かる。Figure 6.6(e) に関しては「no_word」のみとなり、単語が分類されなかった。

実験パターン６〜９のうち、Figure 6.2 にて各クラスタの単語数に大きく差があった実験パターン９について、WordCloud にて同様に確認する。IIC での結果を Figure 6.7(a)〜6.7(d) に、LSA での結果を Figure 6.8(a)〜6.8(d) に、階層的クラスタリングでの結果を Figure 6.9(a)〜6.9(d) に、K-means 法での結果を Figure 6.10(a)〜6.10(d) に示す。

それぞれの WordCloud での結果を見ると、IIC である Figure 6.7(a) は「話題」「心境」「ブログ」「最近」といった単語、Figure 6.7(b) は「娘」「披露」「結婚」「出席」といった単語、Figure 6.7(c) は名前やひらがな、Figure 6.7(d) は「企業」「メーカー」「年代」「先輩」といった類似性の高い単語が含まれていることがわかる。

LSA である Figure 6.8(a) は「こと」「映画」「絵本」「監督」「子ども」といった単語、Figure 6.8(b) は「デジタル」「スマホ」「android」といった単語、Figure 6.8(c) は「監督」「映画」「ロード」といった単語、Figure 6.8(d) は「絵本」「子ども」「手当」「支給」といった類似性の高い単語が含まれていることがわかる。Figure 6.8(a) に関しては他の 3 つのクラスタで強調されている単語が強調されていることが分かる。

階層的クラスタリングである Figure 6.9(a) は「沖」「列島」、Figure 6.9(b), Figure 6.9(d) は類似度の低い単語が多く、Figure 6.9(c) は「駅」「線」といったが含まれていることが分かる。全体的に見ると類似性の低い単語が多いことが分かる。

K-means 法である Figure 6.10(a)〜Figure 6.10(d) は「抵抗」「反発」「反対」や、「メートル」と「センチ」や、「主演」と「公演」や、「区」と「行政」など関連している単語もあるが、全体的に類似性の低い単語が多く、一つのクラスタ内に複数のジャンルの単語が集まっていることが分かる。

(a) Cluster 0.

(b) Cluster 1.

(c) Cluster 2.

(d) Cluster 3.

Figure 6.7: IIC, clusters:4.

(a) Cluster 0.

(b) Cluster 1.

(c) Cluster 2.

(d) Cluster 3.

Figure 6.8: LSA, clusters:4.

(a) Cluster 0.

(b) Cluster 1.

(c) Cluster 2.

(d) Cluster 3.

Figure 6.9: Hierarchical Clustering, clusters:4.

(a) Cluster 0.

(b) Cluster 1.

(c) Cluster 2.

(d) Cluster 3.

Figure 6.10: K-means, clusters:4.

Table 6.1: Accuracy of Livedoor News.

Genres	Documents	IIC	LSA	Hierarchical	K-means
		Accuracy[%]			
5	10	80.0	100	70.0	80.0
	15	85.0	95.0	85.0	60.0
	20	95.0	95.0	85.0	60.0
	25	80.0	96.0	64.0	64.0
6	12	75.0	75.0	75.0	66.6

6.1.3 正解率

実験パターン１～５での各手法での正解率を Table 6.1 に示す。IIC はオーバークラスタリング率が 100、ユニット数が 400 のものを用いて実験を行った。クラスタ数はジャンル数と同数とした。パターン１～４では「スポーツ・食べ物・産業・教育・気象」のジャンルから文書を抽出した。パターン５では「スポーツ・食べ物・産業・教育・気象・海外」のジャンルから文書を抽出した。

Table 6.1 より、文書数によっては IIC と並ぶが、全体を通して LSA による次元削減を用いた場合の正解率が高いことが分かる。また文書数が増えるほど IIC の正解率が高くなり、他手法の中でも特にクラスタ分析の正解率が落ちる結果となった。クラスタ分析手法である、階層的クラスタリングと K-means 法と比較すると、IIC が優位であることが分かる。

実験パターン６～９での各手法での正解率を Table 6.2 に示す。IIC はオーバークラスタリング率が 100、ユニット数が 400 のものを用いて実験を行った。クラスタ数はジャンル数と同数とした。パターン６、７では「dokujo-tsushin・it-life-hack・kaden-channel・livedoor-homme・movie-enter・smax・sports-watch」のジャンルから文書を抽出した。パターン８、９では「dokujo-tsushin・kaden-channel・movie-enter・sports-watch」のジャンルから文書を抽出した。

実験パターン６～９では実験パターン１～５に比べると、全体的に正解率が低く、文書数が増えると正解率が低くなる結果となった。また IIC ではおおよそ半分の正解率となった。実験パターン１～５と同様に、全体的に LSA の正解率が高い結果となった。

Table 6.2: Accuracy of Livedoor Newscorpus.

Genres	Documents	IIC	LSA	Hierarchical	K-means
		Accuracy[%]			
9	18	61.1	72.2	55.5	55.5
9	27	37.0	59.25	44.4	48.1
4	12	50.0	91.6	58.3	66.6
4	20	50.0	90.0	55.5	65.0

Table 6.3: Accuracy (9jenres * 2documents).

Genres	Documents	Clusters	IIC	LSA	Hierarchical	K-means
			Accuracy[%]			
9	18	9	61.1	72.2	55.5	55.5
		10	61.1	72.2	61.1	61.1
		18	55.5	55.5	61.1	61.1
		27	55.5	55.5	72.2	61.1

Table 6.4: Accuracy (9jenres * 3documents).

Genres	Documents	Clusters	IIC	LSA	Hierarchical	K-means
			Accuracy[%]			
9	27	9	37.0	59.25	44.4	48.1
		10	40.7	59.25	48.1	59.25
		18	44.4	59.25	55.5	70.37
		27	55.5	55.5	48.14	48.14

実験パターン６～９の正解率を高めるため、これまでジャンル数と同数としていたクラスタリングの際のクラスタ数を、ジャンル数の+1、ジャンル数× 2, ジャンル数× 3 として再度実験を行った。この場合での正解率を比較する。そのうちパターン６と７の結果を Table 6.3 と Table 6.4 に示す。

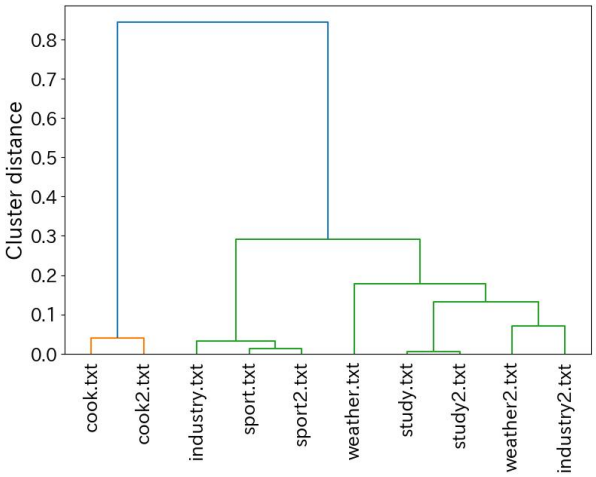
パターン６をみると、クラスタ数をジャンル数+1 とした場合は LSA の正解率が最も高いことが分かる。しかしジャンル数× ２、ジャンル数× ３と、クラスタ数を増やした場合は階層的クラスタリング、K-means 法の正解率が高くなることが分かる。反対に IIC と LSA の正解率は低くなることが分かる。パターン６では IIC と LSA、階層的クラスタリングと K-means 法の正解率がほぼ等しい結果となった。パターン７をみると、LSA の正解率が最も高いことにが分かる。しかし LSA はクラスタ数を変化させると、正解率が低くなるのに対して、その他の手法では、正解率が高くなった。IIC はジャンル数× ３、階層的クラスタリングと K-means 法ではジャンル数× ２とした場合の正解率が高い結果となった。また実験パターン７にて、ジャンル数× ３とした場合、LSA と同正解率となった。

6.1.4 IIC、潜在的意味解析、クラスタ分析による次元削減のデンドログラム

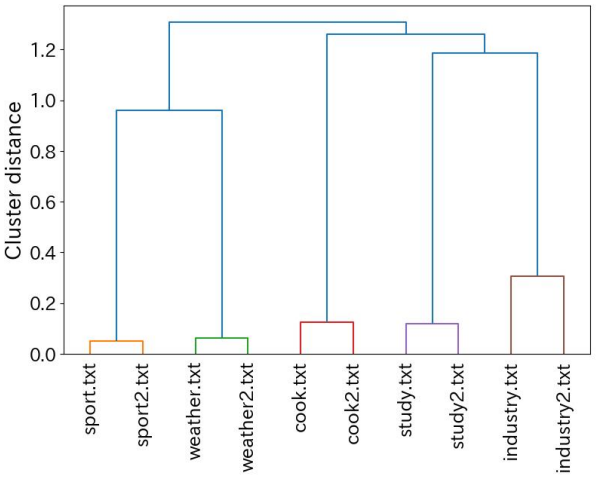
実験パターン１～５のうち、正解率に差があった実験パターン１と３におけるデンドログラムを各手法ごと出力する。実験パターン１の場合のデンドログラムを Figure 6.11(a)～6.11(d) に、実験パターン３の場合のデンドログラムを Figure 6.12(a)～6.12(d) に示す。

Figure 6.11(a)～6.11(d) の実験パターン１における結果を確認すると、Figure 6.11(a) の IIC と、Figure 6.11(d) の K-means 法では産業以外はジャンルごとのクラスタが出来ていることが分かる。Figure 6.11(b) の LSA では全ジャンルのクラスタにまとまりがあり、クラスタが生成されていることが分かる。Figure 6.11(c) の階層的クラスタリングでは、スポーツと食べ物、教育にはまとまりがあるが、それ以外は他クラスタと混合していることが分かる。

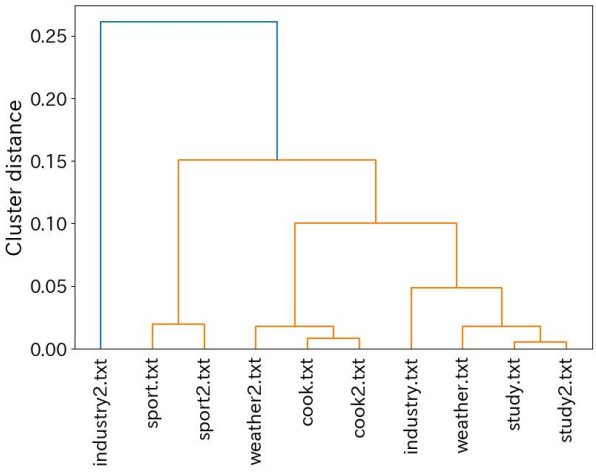
Figure 6.12(a)～6.12(d) の実験パターン３における結果を確認すると、Figure 6.12(a) の IIC では、気象の１文書が産業のクラスタに入ってしまっているが、それ以外はジャンルごとにクラスタにまとまりがあることが分かる。Figure 6.11(b) の LSA では教育の１文書が産業のクラスタに入ってしまっているが、それ以外はジャンルごとにクラスタ



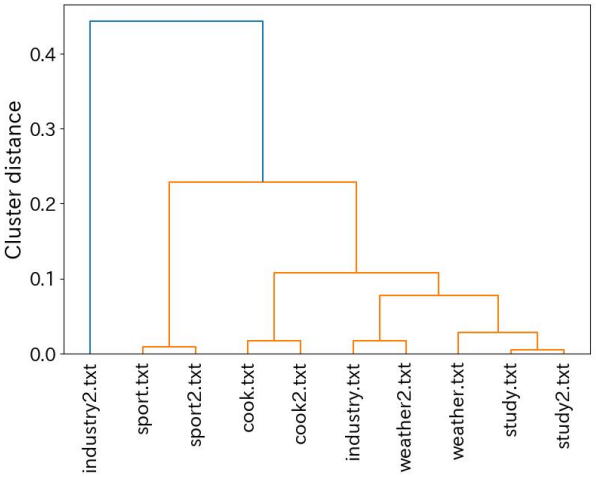
(a) IIC.



(b) LSA.

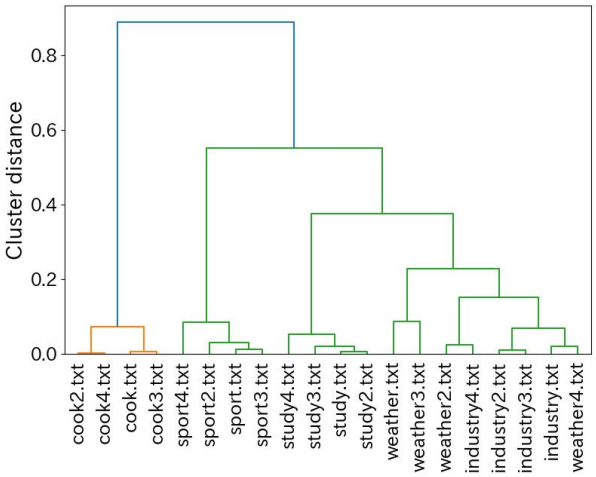


(c) Hierarchical.

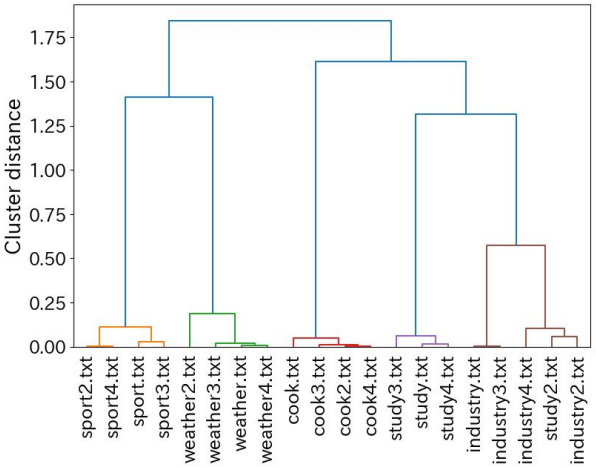


(d) K-means.

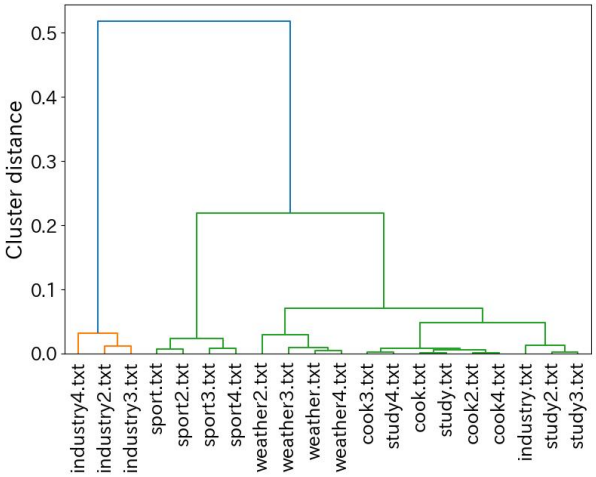
Figure 6.11: Dendrogram of 10 documents.



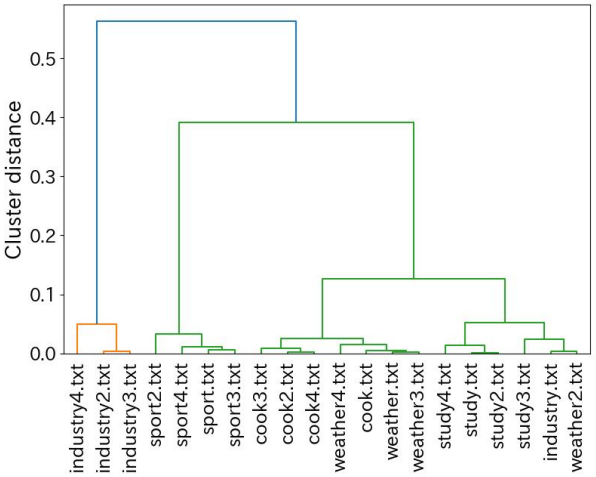
(a) IIC.



(b) LSA.

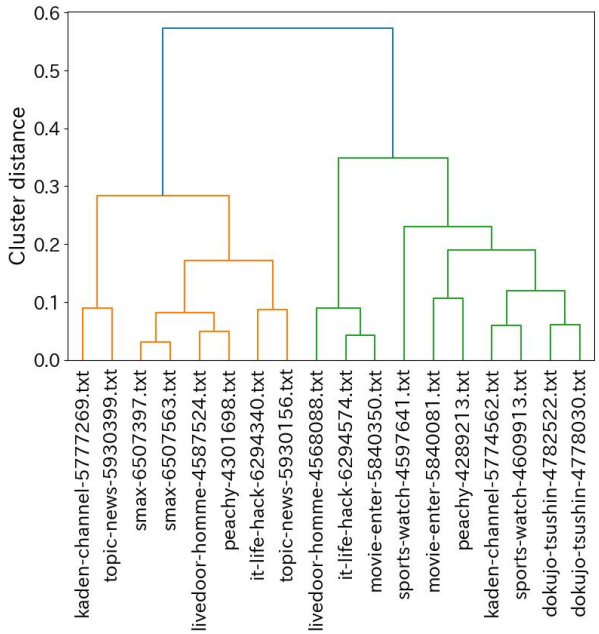


(c) Hierarchical.

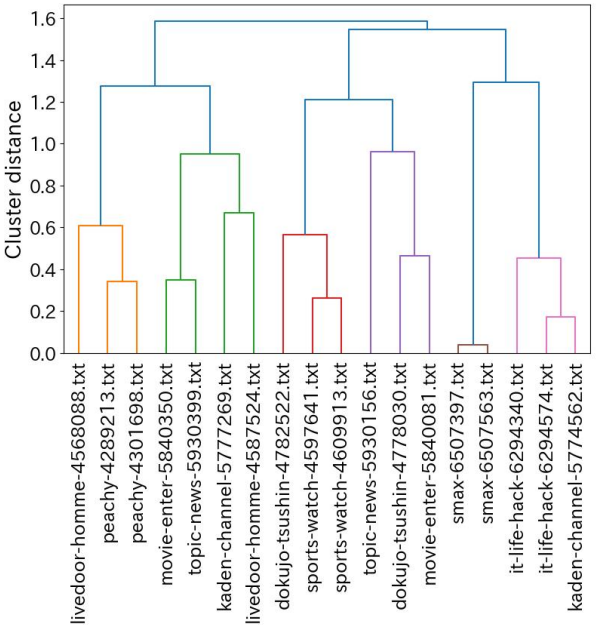


(d) K-means.

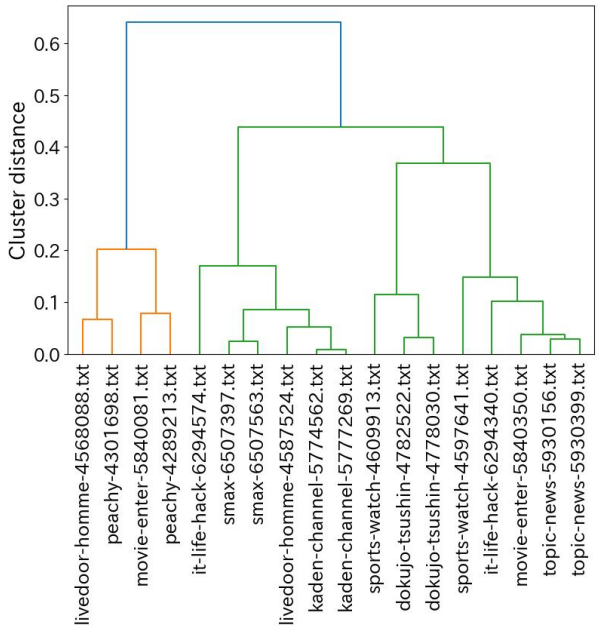
Figure 6.12: Dendrogram of 20 documents.



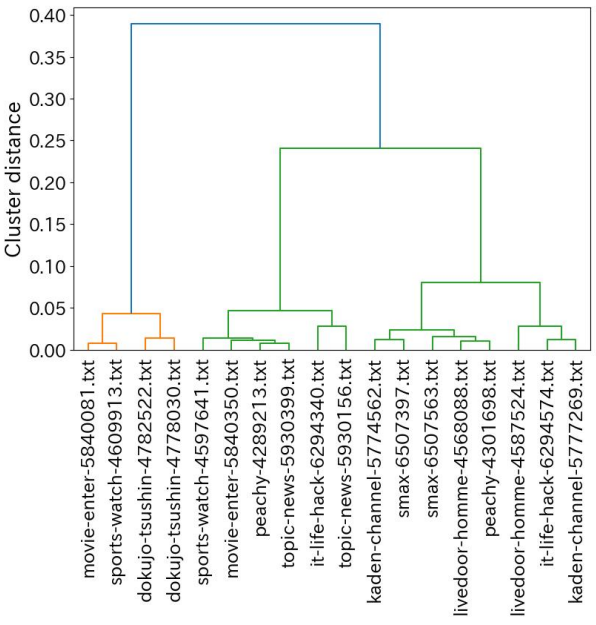
(a) IIC.



(b) LSA.

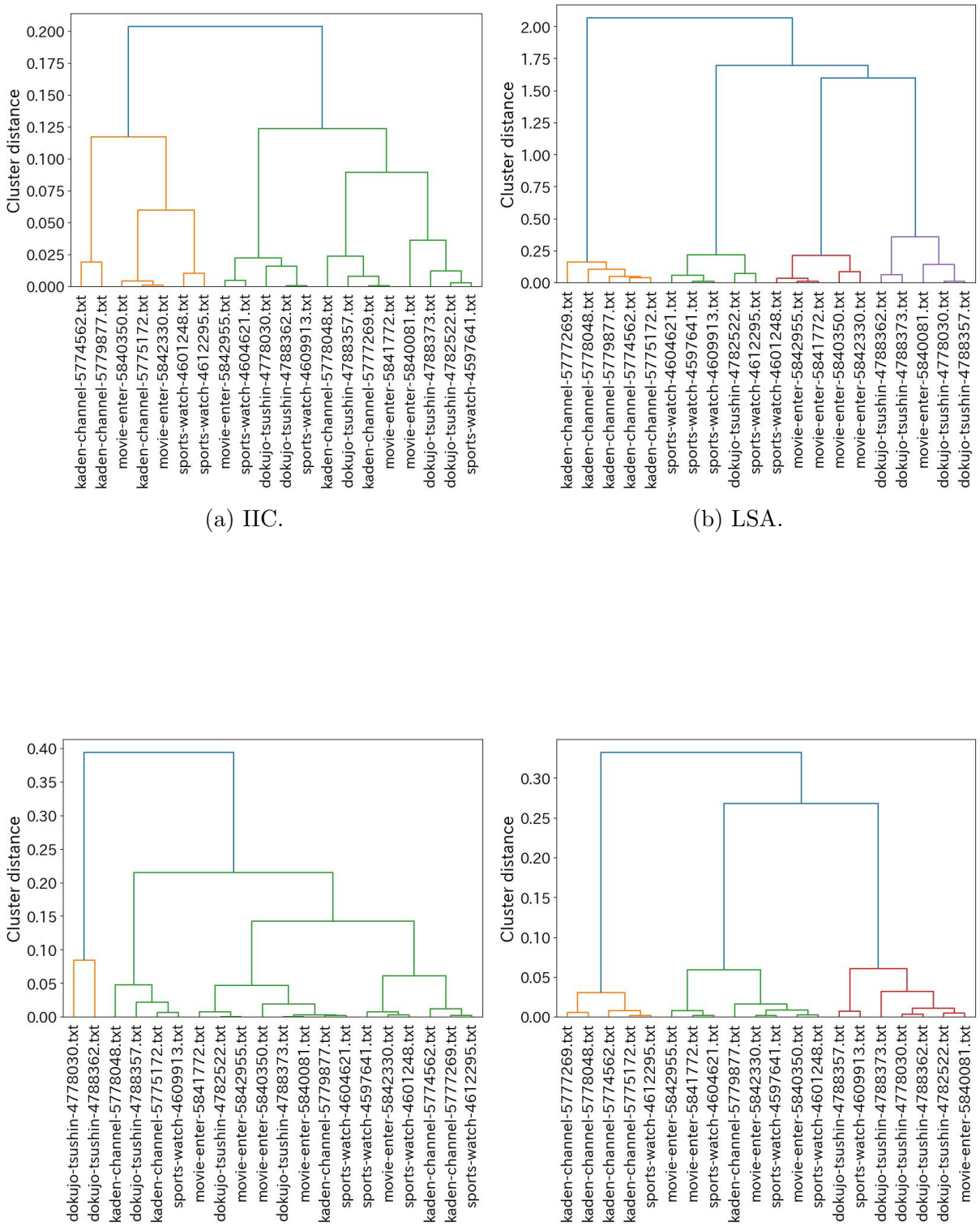


(c) Hierarchical.



(d) K-means.

Figure 6.13: Dendrogram of 18 documents.



(a) IIC.

(b) LSA.

(c) Hierarchical.

(d) K-means.

Figure 6.14: Dendrogram of 20 documents.

にまとまりがあることが分かる。Figure 6.11(c) の階層的クラスタリングでは、産業と教育にばらつきがあるが、それ以外はまとまりがある。Figure 6.11(d) の K-means 法ではスポーツと、産業の一部の文書以外は、ジャンルが混合していることが分かる。

実験パターン６と７のうち、パターン７は文書数が多くファイル名が見にくいため、実験パターン６におけるデンドログラムを Figure 6.13(a)〜6.13(d) に示す。実験パターン８と９のうち、正解率に差があった実験パターン９におけるデンドログラムを Figure 6.14(a)〜6.14(d) に示す。

実験パターン６の結果である Figure 6.13(a)〜6.13(d) と、実験パターン９の結果である Figure 6.14(a)〜6.14(d) より、IIC と階層的クラスタリング、K-means 法には大きな違いはなく、あまりジャンルに分類できていないことが分かる。LSA は一部ジャンルに分類できており、おおまかにクラスタが形成されていることが分かる。

6.2 考察

6.2.1 単語数分布の評価

本研究では、文書のクラスタリングにおける IIC の特性を明らかにするため、潜在的意味解析（LSA)、階層的クラスタリング、K-means 法との比較実験を行った。

各クラスタに含まれる単語数を比較した結果、IIC ではクラスタ間の単語数が比較的均等に分布する傾向が確認された。反対に階層的クラスタリング、K-means 法では、一部のクラスタに単語が集中し、極端に単語数の少ないクラスタが発生するケースが多く見られた。これは従来のクラスタ分析手法での、クラスタの特徴量が他クラスタと似通ることや、学習データのノイズに左右されることで、クラスタが一つにまとまってしまう（本来あるべきクラスタが消失する）という課題が顕著に表れた結果であると考えられる。結果 IIC が、クラスタリングの際の、特定クラスタへの過度な集中を軽減する性質を持つことを示している。特定クラスタに対する偏りが小さいことは、安定した分類結果につながる要因の一つであると考えられる。このことから、IIC はクラスタの単語数の均衡性という観点において優れた特性を持つ手法であるといえる。

6.2.2 WordCloud による評価

WordCloud を用いた可視化結果である、Figure 6.3(a)〜6.10(d) から、各クラスタに含まれる単語の意味的特徴を確認した。LSA では、同ジャンルの類似した分布結果が現わ

れること、二つのジャンルの単語が混在するトピックが形成されていることが確認された。類似性の高い単語が強調されてはいるが、各トピックにジャンルに対応する語彙が均等に分類されているわけではなかった。IIC では、同一クラスタ内に類似性の高い単語が集まる傾向が確認され、文書や単語の意味的な関連性をある程度反映したクラスタが形成された。一方で、階層的クラスタリングおよび K-means 法では、クラスタ内に複数ジャンルの単語が混在する傾向が強く、意味的な関連性の低さが確認された。

LSA は文書行列に対して特異値分解を行い、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、次元削減する手法であるが、各トピックがどのような意味を持つのか分かりにくい場合が見られた。これは LSA では一つの単語が複数のトピックに、同時に関係する場合があることが原因であると考えられる。結果としてトピックの解釈が難しくなった可能性がある。比較すると、IIC が文書全体の単語の意味を考慮してクラスタリングできていることが分かり、次元削減をした際にも意味的な特徴を反映して次元削減を行うことが出来たと考えられる。また WordCloud による定性的評価は正解率だけでは把握できないクラスタの生成結果の品質を可視化する手法として有効であるといえる。

6.2.3 正解率と各手法の特性

Table 6.1 と Table 6.2 より、文書数によっては IIC が LSA と並ぶが、全体を通して LSA による次元削減を用いた場合の正解率が高い結果となった。またクラスタ分析手法である、階層的クラスタリングと K-means 法と比較すると、IIC が優位である結果となった。加えて、文書数が増加するにつれて IIC の正解率が向上し、結果が安定する傾向が確認された。反対にクラスタ分析手法である階層的クラスタリング、K-means 法は文書数の増加により、正解率が落ちる結果となった。

Wordcloud では、IIC が最も同一クラスタ内に類似性の高い単語が集まる傾向が確認され、LSA は類似性の高い単語が強調されるが、各トピックにジャンルに対応する語彙が均等に分類されず、階層的クラスタリングと K-means 法では、意味的な関連性の低さが見られた。クラスタ内に意味的に近い単語が多く含まれる場合、ジャンルごとの特徴が明確になり、文書の分類精度が向上しやすいと考えられる。一方で、単語の意味的なまとまりが弱いクラスタではジャンルの特徴が曖昧となり、異なるジャンル間の判別が困難になる。その結果、階層的クラスタリングおよび K-means 法と比較して、IIC の正解率が高くなったと考えられる。LSA ではトピックの意味の解釈は難しいが、文書中に

含まれる重要な特徴が強調されることで、ノイズとなる単語の影響が抑えられる。その結果、同じジャンルに属する文書同士の類似性が高まり、正解率が高くなったと考えられる。

IIC は教師なし学習に基づく深層学習手法であり、本研究では文書集合から得た単語集合に対してクラスタリングを行っている。そのため、学習に用いる文書数が少ない場合には、単語間の関係を十分に捉えることができず、クラスタごとの特徴が不明瞭になると考えられる。一方、文書数が増加することで、単語間の関係性が安定し、意味的に近い単語が同一クラスタに集まりやすくなり、正解率の向上につながったと考えられる。この結果から、IIC は十分な文書数が確保される環境において有効性を発揮する手法であると考えられる。一方で、階層的クラスタリングおよび K-means 法では、単語ベクトル間の距離のみに基づいてクラスタを形成するため、文書数の増加に伴い語彙の種類が多様になるほど、単語間の距離関係が複雑化し、クラスタ間の違いが不明瞭になったと考えられる。その結果、次元削減後の文書分類において各ジャンル固有の特徴が弱まり、判別が困難になった可能性があると考えられる。

パターン６から９のニュースコーパスの記事の場合にも LSA が比較的高い正解率となったが、パターン１から５の現在掲載されている記事の結果には及ばなかった。ニュース記事のように、ジャンルごとに用いられる単語に一定の傾向がある文書集合に対して、有効性があると考えられる。

6.2.4　文書数およびジャンルについて

実験パターン１～５と実験パターン６～９を比較すると、後者において正解率が全体的に低下する結果となった。これは使用したジャンル構成の違いが大きく影響していると考えられる。実験パターン１～５では、スポーツ・食べ物・産業・教育・気象・海外といった比較的語彙の特徴が明確なジャンルを対象としている。これらのジャンルでは、特定の分野に特有の単語が頻出する傾向があることで、文書間の類似性が捉えやすい。よって使用した手法に問わず、ジャンル構造が捉えやすかったと考えられる。

一方、実験パターン６～９で使用した dokujo-tsushin・livedoor-homme・movie-enter などのジャンルは、日常的話題や雑記的内容、日々の生活や暮らしなどを多く含む特徴を持つ。このようなジャンルでは話題の多様性が高く、使用される単語のばらつきが大きいため、文書間の意味的な類似性を単純な単語ベクトル表現から推定することが困難

となる。よって同一ジャンル内の文書同士の距離が離れ、クラスタ形成が不安定になったと考えられる。このことから、対象とするデータセットに大きく影響を受けることが示唆される。

6.2.5　クラスタ数の変化

パターン６と７では、ジャンル数に対してクラスタリングの際のクラスタ数を変化させた条件下で IIC を適用し、クラスタ数の違いがクラスタリング結果に与える影響について検討を行った。実験結果より、パターン６では、IIC と LSA はジャンル数よりも多いクラスタ数を指定すると、正解率が低下したが、階層的クラスタリング、K-means 法では正解率が向上した。パターン７ではジャンル数よりも多いクラスタ数の場合は、全体的に正解率が向上する傾向が見られた。これは、各ジャンルに明確に分類できない文章が多数存在していたことが原因と考えられる。前項で示した通り、実験パターン６～９で使用した文書データセットには、複数ジャンルにまたがる内容や、感想や気持ちといった感情や、明確なジャンル特性を持たない文章が含まれており、それらをいずれか一つのジャンルクラスタに強制的に割り当てることが、誤分類の増加につながった可能性がある。そこで明確なジャンルに当てはまらない文章や特徴が曖昧な文章が、特定ジャンルとは異なるクラスタを用意したことで、このクラスタに集約され、主要ジャンルにおけるクラスタの純度が向上したと考えられる。データ数がより多い場合、明確に分類できない文章が増加したため、パターン６よりパターン７で正解率が向上したと考えられる。しかし、データ数が増えることで、本来分類したいジャンルの文書のパターンや傾向がよりジャンルごと明確になるため、データ数を増やし、クラスタ数も変化させることが、正解率を必ずしも向上させるわけではないとも考えられる。ジャンル数が最適なクラスタ数とは限らない場合があることから、クラスタ数を事前に固定せず、データ構造に基づいて自動的に推定することで、より正解率の向上が期待できると考えられる。

6.2.6　デンドログラムによるクラスタ構造

デンドログラムによる可視化結果から、各手法のクラスタ構造の違いを確認した。IIC では条件によってジャンルごとのまとまりが形成される傾向が確認された。特に文書数が比較的多い条件では、同一ジャンルの文書が近い位置に配置され、一定のクラスタ構造が形成される結果となった。一方で、文書数が少ない条件ではクラスタ間の距離が不

安定となり、明確な分離が困難となる場合も見られた。これは、IIC における学習が十分に進まなかったことが要因であると考えられる。この結果から、IIC は学習量の影響を強く受ける特徴を持つことが示唆された。

LSA では、多くの実験パターンにおいてジャンルごとの文書が比較的近い距離に配置されており、意味的に類似した文書同士がまとまりを形成していることが確認された。一方、K-means 法および階層的クラスタリングでは、IIC と比較すると異なるジャンルの文書が同一クラスタ内に混在する傾向が確認された。このことから、LSA には及ばないが、クラスタ分析手法に対する IIC の優位性を確認することが出来たと考えられる。

第7章　結論

本研究では、livedoor ニュースの記事について、文書行列の次元削減において、提案手法である情報量最大化クラスタリング（IIC）と、潜在的意味解析（LSA）、ward 法での階層的クラスタリング、K-means 法での非階層的クラスタリングによる次元削減との比較によって、その有用性について検討した。手順としては、初めに livedoor ニュースの記事の取得、形態素解析、Word2vec による単語のベクトル化、TF-IDF の導出を行った。その後、IIC ではデータセット生成、ニューラルネットワークの構築、学習を行い、単語を分類した。ward 法での階層的クラスタリング、K-means 法での非階層的クラスタリングでも単語の分類を行った。クラスタの生成結果の評価のため、各クラスタの単語数の調査、WordCloud による可視化を行った。その後、これらによって得た行列から次元削減を行い、COS 類似度を用いて各文書間の類似度計算を行った。その分析結果をデンドログラムにて表した。類似度計算の評価のために文書分類を行い、正解率を算出した。LSA による次元削減も行い、WordCloud による可視化、デンドログラムと正解率による評価を行った。

クラスタリングの際の生成結果の評価という観点では、他手法と比較して、IIC の有効性を得ることができた。この結果が得られた理由として、本研究の結果で明らかになった、IIC の複数の特性が挙げられる。文書数の増加に伴う性能の安定化、単語数分布（クラスタのサイズ）の均等性、類似性の高い単語が同一クラスタ内に配置された点、またこれらにより文書分類のデンドログラムは、クラスタの構造的なまとまりが確認された点は、従来の LSA や距離尺度に基づくクラスタ分析手法には見られない IIC の優れた特徴であるといえる。文書分類の正解率という定量的評価の観点では、LSA が最も高い性能を示した一方で、IIC は階層的クラスタリングおよび K-means 法と比較して高い正解率を示し、次元削減手法として一定の有効性を有することが確認された。本研究において IIC は、単語の意味的な特徴を考慮したクラスタリングを通じて、文書行列の次元削減に有効に機能する手法であることが示された。また IIC は大規模な文書データにおいても、クラスタが一つにまとまる現象、本来あるべきクラスタが消失する現象を防ぎ、より単語の意味を考慮したクラスタリングを用いた次元削減が可能であると考えられる。

また、正解率という定量評価に加え、文書分類のデンドログラムや各単語クラスタの WordCloud といった定性評価を組み合わせることで、クラスタリング結果を多角的に評

価できることを示した点は、本研究の特徴の一つであるといえる。

今後の課題としては、IIC におけるパラメータの最適化や学習安定性の向上が挙げられる。また、クラスタ数を事前に固定せず、データ構造に基づいて自動的に推定する構造の構築、BERT などの分散表現を用いた手法を導入することで、文書の意味情報をより高精度に表現できる可能性がある。また、対象とするデータセットに大きく影響を受けることが示唆されたため、小説の文章など扱った文章を変えた場合の結果について検討が必要である。

第8章　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) 元田浩　津本周作　山田高平　沼尾正行　共著, データマイニングの基礎, オーム社, 2008,
2) 新納　浩幸　古宮　嘉那子　著, 文書分類からはじめる自然言語処理入門ー基本からBERTまでー, 科学情報出版株式会社, 2022,
3) ヤン・ジャクリン, 機械学習で「超重要な」特徴量とは何か？ 設計方法などについてわかりやすく解説する, ビジネス+IT, 2021,
   https://www.sbbit.jp/article/cont1/76066, (参照 2026-01-12)
4) SONY, ディープラーニングにおける中間層の役割とは？基本的な仕組みや考え方を解説,
   https://dl.sony.com/ja/deeplearning/about/middle_layer.html, (参照 2026-01-14)
5) atmarkIT,［活性化関数］ソフトマックス関数（Softmax function）とは？, 一色政彦, 2023-10-02,
   https://atmarkit.itmedia.co.jp/ait/articles/2004/08/news016.html, (参照 2026-01-14)
6) 機械学習ナビ, ソフトマックス関数 (softmax 関数) とは?機械学習の視点で分かりやすく解説!!, 2021-11-19,
   https://nisshingeppo.com/ai/softmax-function/, (参照 2026-01-13)
7) AI 教師あり学習の精度を超えた!? 相互情報量の最大化による教師なし学習手法 IIC の登場！, 2020-02-01,
   https://ai-scholar.tech/articles/treatise/iic-ai-367, (参照 2026-01-13)
8) 高校数学の美しい物語, 相互情報量の意味とエントロピーとの関係, 2022,
   https://manabitimes.jp/math/1403, (参照 2026-01-14)
9) 金子　冴, 【技術解説】形態素解析とは？MeCab インストール手順から Python での実行例まで, ミエルカ AI media, 2018-05-14,
   https://mieruca-ai.com/ai/morphological_analysis_mecab, (参照 2026-01-12)
10) Smiley, PyTorch とは?特徴やメリットからインストールの方法まで解説, 2026-01-25,

https://aismiley.co.jp/ai_news/pytorch/, (参照 2026-01-14)

11) 初心者 DIY プログラミング入門,【実践】Python で WordCloud（ワードクラウド）しようぜ！, 2023-11-10, https://resanaplaza.com/2022/05/21/%e3%80%90%e5%ae%9f%e8%b7%b5%e3%80%91python%e3%81%a7wordcloud%ef%bc%88%e3%83%af%e3%83%bc%e3%83%89%e3%82%af%e3%83%a9%e3%82%a6%e3%83%89%ef%bc%89%e3%81%97%e3%82%88%e3%81%86%e3%81%9c%ef%bc%81/, (参照 2026-01-14)

12) 株式会社ライブドア, Livedoor ニュース, https://news.livedoor.com/, (参照 2026-01-10)

13) ロンウィット, Livedoor ニュースコーパス, https://www.rondhuit.com/download.html, (参照 2026-01-10)

14) 鈴木正敏, 日本語 Wikipedia エンティティベクトル https://www.cl.ecei.tohoku.ac.jp/~m-suzuki/jawiki_vector/, (参照 2026-01-10)

15) Qiita, Pytorch のニューラルネットワーク (CNN) のチュートリアル 1.3.1 の解説, 2020-01-29, @poorko, https://qiita.com/poorko/items/c151ff4a827f114fe954, (参照 2026-01-14)