卒 業 研 究 報 告 題 目

Word2vecを利用した クラスター分析による文書の分類

Classification of documents
by cluster analysis using Word2vec

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

2016E30 長谷川　翔海

令 和 ０ ３ 年（ ２ ０ ２ １ 年 ）　　２ 月 １ ５ 日 提 出

Abstract

The purpose of this study is to confirm the effectiveness of dimensionality reduction by cluster analysis proposed in the laboratory. This method can reduce the dimension by considering the meaning of the word. I classify texts in experiments. This experiment compares latent semantic analysis with dimensionality reduction by cluster analysis The text used is a synopsis of the novel. I did the analysis using Python. Dimensionality reduction by cluster analysis is performed using Word2vec. Text classification is done by cluster analysis using cosine similarity as the distance. The method of cluster analysis is Hierarchical clustering by Ward's method. Figure 1 is an example of a dendrogram. The dendrogram and accuracy rate made from the analysis results are used to verify the effectiveness of the dimensionality reduction method. As a result of the experiment, the effectiveness of dimensionality reduction by cluster analysis was confirmed. I thought it was because the dimensionality reduction by cluster analysis could distinguish the meaning of the word.

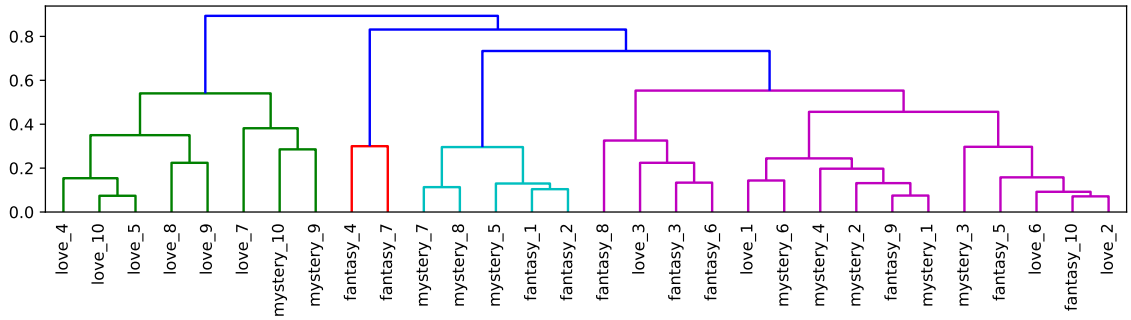

Figure 1　Example of dendrogram.

目 次

第1章　序論

現在、コンピュータやスマートフォンといった情報端末は、企業から個人、高齢者から若年者まで幅広い層へと普及が進んでいる。多くの人々によるソーシャルネットワーキングサービスの利用、それに伴うインターネットの著しい発展により、膨大な量のデータが蓄積されてきた。今後も情報端末の数は増大していくため、蓄積されるデータ量は増えていくことが予想される。次々に生み出される情報は有益なものから虚偽のものまで様々であり、その中から個人にとって必要なものを見つけ出すことは労力を要するようになりつつある。そうした問題を解決するため、コンピュータを活用し、有益なデータのみを抽出するための技術が生み出された。これを、データマイニングという。その中でも、大量のテキストデータに対し、言語処理技術を用いてデータ解析することをテキストマイニングという。テキストマイニングは、様々な事に応用されているが、現状では課題も多く、完成された技術ではないとされている。

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、研究室で提案された手法を実装し、その有効性の確認を目的として実験を行う。提案された次元削減手法は、単語分散表現が取得できる word2vec とクラスター分析の手法を併用したものであり、これにより単語が持つ意味を考慮した次元削減が可能であるとされた。類似度を計算するテキストデータの対象には、小説のあらすじを採用した。取得した小説のあらすじはいくつかのジャンルに分けられており、あらすじ間の類似度が高いものほど似た作品、また同じジャンルに属する小説であると考えられたためである。

実験では、従来の統計的な次元削減手法である潜在的意味解析 (LSA) を適用したテキスト間の類似度と、本研究室で提案された次元削減手法を適用したテキスト間の類似度の 2 つを導出する。それを基に、データの関係を表した樹形図を出力し、比較する。これまでの研究では、手法の有効性の評価を出力された樹形図を比較することで行ってきた。本研究ではそれに加え、分類タスクをこなす機械学習モデルに用いられる評価指標を、本実験の出力結果に適用することで、有効性を判断するための材料とした。

これらを踏まえ、実験の条件に変化を付けた結果をいくつか出力することで、単語間の意味を考慮した次元削減の有効性の検討を行う。

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、データベースに蓄積された大量のデータから、有益な情報や意思決定に必要とされる知識を特定するために用いられる手法のことをいう。少量のデータであれば人の手でも情報を発見することは可能である。しかし、扱うデータ量が膨大になると、人間では有益な情報の発見はおろか、確認作業でさえ多大な作業量を要することとなる。そのためコンピュータの高速な処理を用いて、データの法則等を発見することが求められる。このような場面で、データマイニングは用いられる。

2.1.2　テキストトマイニング

テキストマイニングとは、テキストデータを対象としたデータマイニングのことである。主に言語処理や数学・統計といった技術・手法が使用される。テキストデータには、SNS の投稿やサービスに対するアンケート結果などの文書が例として挙げられ、それらをもとに市場におけるトレンドの分析や文書の検索・分類といった処理がされている。言葉は意味を持っているため、数値を対象としたマイニングと比較して、分析の難度が高いとされている。

2.1.3　形態素解析

形態素とは、単語において意味を持つ最小単位のことである。形態素解析は、テキストデータを文法や品詞の情報をもとに、形態素に分割する処理のことをいう。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。

形態素解析でテキストを単語に分割する理由は、多くのテキストマイニングの技術において単語を入力値として与えて処理をすることが多いためである。

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 というソフトウェアを使用した。

2.1.4 MeCab

形態素解析プログラム MeCab 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である。C 言語で作られたプログラムであり、多くの言語環境で使用できるよう作成された。

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。以下に、形態素解析を行った例を示す。

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

2.2 自然言語

2.2.1 自然言語

自然言語とは、人間が日常的に読み書きに使用し、進化してきた言語のことをいう。日本語や英語、中国語が例に挙げられる。自然言語と対比する概念として、人間によって人工的に作りだされてきた言語である人工言語が存在する。人工言語の例には、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。

自然言語と人工言語には、文法や解釈のしやすさといった違いがある。人工言語の中でも、プログラミング言語はコンピュータが簡単に意味を解釈、実行できるようにデザインされている。これに対し、自然言語はある単語や文が複数の意味に解釈可能であるというような曖昧性を持つことがある。

2.2.2 自然言語の曖昧さ

自然言語の曖昧さには、多義性と類義性の二つの性質がある。

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をいう。多義性の例として「甘い」という単語を挙げる。「甘いお菓子」という文では、味覚的な意

味で使われていることがわかるが、「見通しが甘い」では厳しさに欠けるという意味で使われていることになる。このように「甘い」という単語には2つの解釈が存在することになる。

類義性とは、異なる単語が同じような意味を持つという性質のことを言う。「事典」と「辞書」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。こうした曖昧さが、テキストマイニングの分析の難易度を高めている。

第3章　実験で使用した技術・手法

3.1　計算手法

3.1.1　TfIdf

TfIdf とは、文書をベクトルで表現する計算手法である。ベクトルで表現することで、プログラムにテキストデータを数値として与えることが可能となり、文書の特徴づけや類似度の計算が行える。TfIdf は Tf と Idf の積で求められる。Tf とは Term frequensy の略称であり、文書における特定の単語の出現頻度を表す。出現頻度が高い単語ほど重要であると考えられ、出現頻度が低い単語ほどさほど重要ではないと考えられる。Idf とは Inverse document frequensy の略称である。単語の逆文書頻度と呼ばれ、文書集合における単語の分布の偏りを考慮する値である。ほとんどの文書に出現している単語は、さほど重要ではない単語と考えられ、Idf が低い値となる。逆にとある文書にしか出現しない単語はその文書を特徴づけする重要な単語と捉えられ、Idf が高い値となる。TfIdf はこれら二つの数値の積となる。Tf と Idf には、いくつかの導出方法があるが、本研究では下記の式で計算を行った。

$$Tf_{ij} = \text{文書 } d_i \text{ における } t_{ij} \text{ の出現回数} \tag{3.1}$$

$$Idf_{ij} = \log \frac{\text{全文書数} + 1}{\text{単語 } t_{ij} \text{ が出現する文書数} + 1} + 1 \tag{3.2}$$

$$TfIdf_{ij} = Tf_{ij} \times Idf_{ij} \tag{3.3}$$

3.1.2　cos 類似度

cos 類似度とは、2 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 値が 1 に近いほど、ベクトル同士の挟む角は小さくなり、同じ方向を向いていることとなる。逆に cos 値が -1 に近いほどベクトル同士の挟む角は大きくなり、逆の方向を向いていることとなる。

2 つのベクトルをベクトル $\vec{a}$ とベクトル $\vec{b}$ とすると cos 類似度は以下の式で導出される。

$$\cos(\vec{a}, \vec{b}) = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}||\vec{b}|} \tag{3.4}$$

3.1.3 主成分分析 [1)]

実験や調査において、測定値の項目が少ない場合はグラフや図を用いて人の目でも判断が可能である。しかし、項目数が多い場合、グラフや図では表現することが難しく、人の目による確認も容易にできないパターンがある。こういった事態を解決する技術として、主成分分析 (principal component analysis; PCA) が存在する。

主成分分析とは、ベクトルといった多次元のデータについて、本来持っている情報をできる限り損なうことなく次元を削減する手法である。変数が何百もある多次元データを 10 や 20 といった少ない次元数まで削減するようなことを言う。これにより、データ間の比較や可視化が容易となる。

具体的な方法については、以下に式を交えて説明する。P 個のデータ $x_p(p = 1, 2, \ldots, P)$ について、$N(N \leq P)$ 個の主成分 $z_n(n = 1, 2, \ldots, N)$ とこれらの関係は、次の式のように互いに独立な線形結合として表される。

$$z_n = \sum_{p=1}^{P} a_{pn} x_p \tag{3.5}$$

ここで、z_n は第 n 主成分と呼ばれ、その結合係数 a_{pn} は次の式を満たす必要がある。

$$\sum_{p=1}^{P} a_{pn}^2 = 1 \qquad (\forall n) \tag{3.6}$$

主成分ができる限り多くの情報を持つようにするためには、データの分散に着目し、結合係数を上手く決める必要がある。例として、Figure 3.1 に示す二次元のデータについて考える。この図において、データの分散が最も大きくなる方向に着目すると、z_1 という軸ができる。これが第 1 主成分となり、このような軸ができるように式 (3.5) の結合係数を決定する。しかし、この軸だけでは、データが本来持つ情報を十分に表しているとは言い難い。そこで、z_1 に次いでデータの分散が大きくなる方向に着目し、z_2 という軸をとる。これが第 2 主成分となり、第 1 主成分にて表せないデータを補うことができる。このように結合係数を決めていくことで、情報量の損失を最小限に抑えながら、Figure 3.1 に示される X, Y の特性を把握することができる。今回の例では 2 次元データであったため、目視で判断しやすいものであった。そのため主成分分析の利点は少ないように思える。しかし、高次元のデータでは、その利点が大きく表れるようになる。

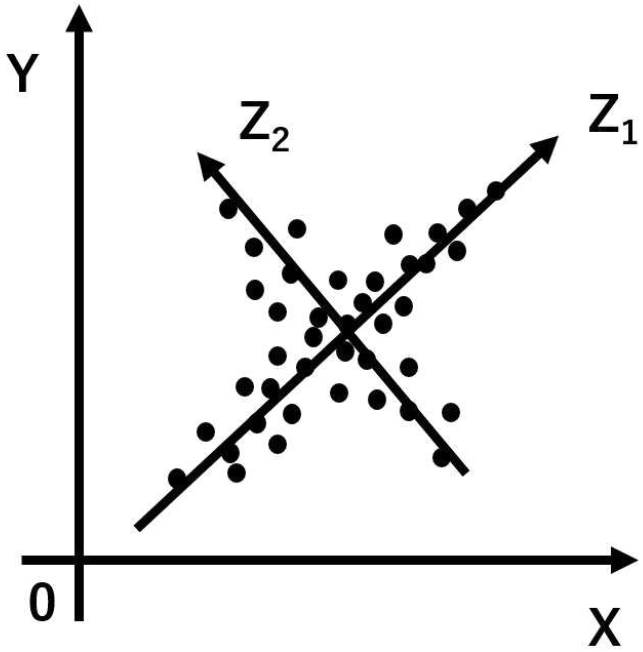

Figure 3.1　Example of two-dimensional data.

3.1.4　主成分数の選択

主成分分析において、主成分数の設定は極めて重要な問題となる。主成分数が少なすぎると、重要な情報までも損なわれてしまい、解析データに綻びが生じてしまう。逆に主成分数が多すぎると、データの削減が十分ではなくなり、主成分分析を行う意味がなくなってしまう。そのため、主成分数の決定には以下に示す 3 通りの方法が存在する。

- 固有値が 1 を越える主成分を採用する。
- ある固有値とその次の固有値の差が小さくなるまでの主成分を採用する。
- 累積寄与率がある値に達するまでの主成分を採用する。

一つ目の方法は、平均と分散を共に 1 としたことで、分散 (固有値) がこの標準化された値である 1 よりも大きければ、説明力のある主成分として用いることができるという考えに基づいている。二つ目の方法は、ある固有値とその次の固有値の差が小さければ、主成分の採用、非採用の区別に大きな意味はないという考えに基づいている。三つ目の方法は、主成分分析後のデータが、主成分分析前のデータが持つ情報の何割かを含んでいればよいという考えに基づいている。主成分ひとつひとつの情報量を計算していき、

その情報量の合計が 60〜80 パーセントである主成分数で次元削減される場合が多い。

累積寄与率とは主成分の寄与率を合計した値を言う。また寄与率とは、ある主成分のあらわす情報は、全体の情報に対してどの程度の情報を含んでいるかを表すものである。上記、3 つ目の方法の説明において記述した、情報量の合計が累積寄与率にあたり、主成分ひとつひとつの情報量が寄与率にあたる。寄与率は次式で表される。

$$P_n = \frac{\lambda_n}{\sum_{p=1}^{P} \lambda_p} \tag{3.7}$$

ここで、式中の λ_n は、n 番目の主成分の固有値を示している。累積寄与率はこの寄与率の総和であるため、主成分数が N の場合は次式で表される。

$$C_n = \sum_{i=1}^{N} P_i \tag{3.8}$$

3.1.5 潜在的意味解析

テキストマイニングにおいて、形態素解析後に生成される単語-文書行列が生成される。単語-文書行列は式 (3.9) で表される。

$$TD = \left(\begin{array}{c|cccc} Term & doc_1 & doc_2 & \cdots & doc_N \\ \hline w_1 & I_{w_1,doc_1} & I_{w_1,doc_2} & \cdots & I_{w_1,doc_N} \\ w_2 & I_{w_2,doc_1} & I_{w_2,doc_2} & \cdots & I_{w_2,doc_N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ w_M & I_{w_M,doc_1} & I_{w_M,doc_2} & \cdots & I_{w_M,doc_N} \end{array}\right) \tag{3.9}$$

単語-文書行列は高次元である事が多い。それにより計算処理に時間をかけてしまうことや、分析には必要ない単語が含まれており後々の妨げとなることがある。これらの問題を解決するときに、潜在的意味解析 (Latent Semantic Analysis;LSA) が用いられる。潜在的意味解析は、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手法である。潜在的意味解析では、特異値分解 (singular value decomposition; SVD) という行列分解手法を用いて次元を削減する。以下に、文書行列 TD を特異値分解する式を示す。

$$TD = U\Sigma V^T \tag{3.10}$$

この式における U, Σ, V^T は行列を表しており、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^T は右特異 (文書) ベクトルと呼ばれる。 特異値分解で得られた左特異ベクトルは含まれる情報の重要度が高い成分から順に並んでいる。そのため、行列の左から k 列を抜き出した行列 U_k と文書行列 TD を式 (3.11) のように計算することで、必要な情報のみを抜き出した行列を生成することが可能となる。

$$TD_k = U_k^T TD \tag{3.11}$$

3.1.6 クラスター分析

クラスター分析とは、様々な性質が混在したデータから、似た性質を持つ者同士に分類するアルゴリズムである。主にデータの傾向をつかみたい場合に使用され、分類により作られたデータの集合はクラスターと呼ばれる。また、教師なし学習の手法であるため、教師あり学習のようなデータへのラベル付けをすることなく分類が行える。

クラスター分析には、大きな枠組みとして階層的クラスター分析と非階層的クラスター分析が存在する。本研究では、階層的クラスター分析を採用したため、こちらの説明をする。階層的クラスター分析とは、データ間の距離を基に、距離が近いものから順にクラスターを作成していき、最終的に階層のようなクラスター構造を形成する分類手法である。形成されたクラスターの構造は、Figure 3.2 に示す樹形図によって視覚的に判断が可能となる。樹形図において、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。階層的クラスター分析において、クラスター間の距離測定の方法にはいくつか種類があるが、本研究ではウォード法を採用した。ウォード法は 2 つのクラスターを融合した際に、同クラスター内の分散とクラスター間の分散の比を最大化するようにクラスターを形成していく方法である。

3.1.7 正解率 [2)]

正解率 (Accuracy) とは、機械学習の分類問題に用いられる評価指標のひとつである。本研究では、分析手法の評価項目の 1 つとして導入した。分類問題の評価指標には正解率のほかに適合率、再現率、F1 値といった値がある。これらは、ラベルごとのデータ数に偏りがある場合や重要視する評価の側面に応じて使い分けされる。本研究では、全体

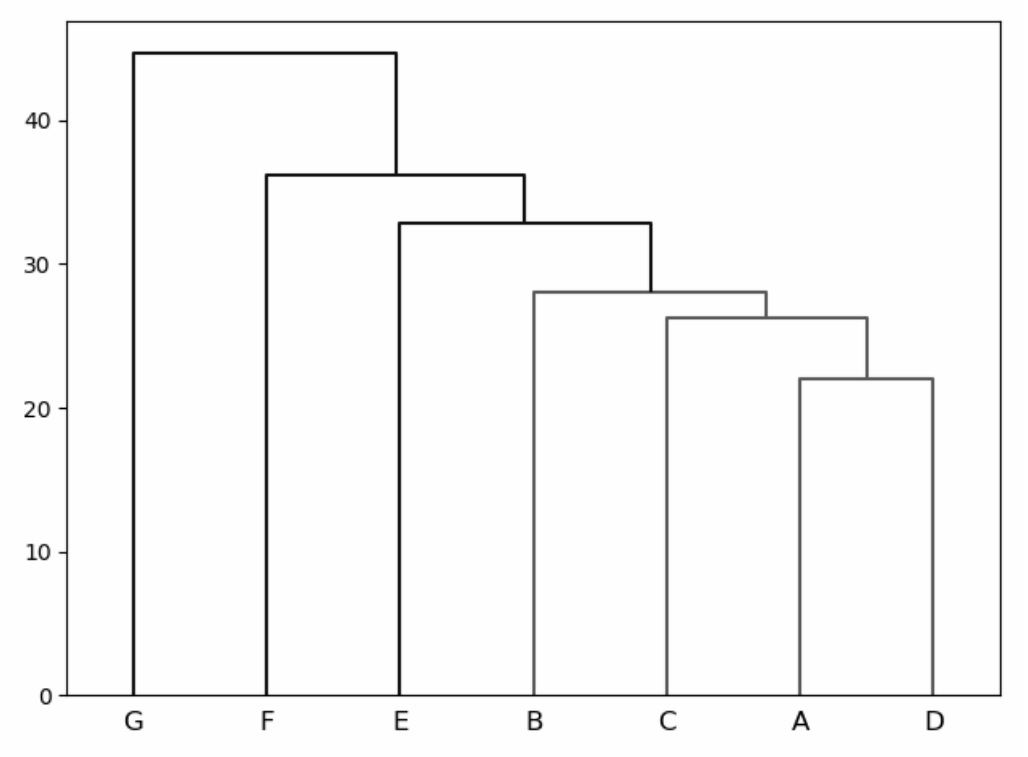

Figure 3.2　Dendrogram.

のパフォーマンスをわかりやすい数値で知りたかったことや分析データの特徴を考慮した結果、正解率を採用することにした。

正解率は、分類の結果をまとめた混同行列を用いて計算される。以下に混同行列の具体例を交えて正解率の導出方法について説明する Figure 3.3 は 3 クラス分類における混同行列の具体的な例である。混同行列では各列が予測されたラベルを、各行が真のラベルを表す。そのため行列の対角成分にある値が、各ラベルにおいて正しく予測がなされたものとなる。この正しく予測がなされた値と全体のデータ数により、正解率は式 3.12 のように計算される。

$$\text{正解率} = \frac{\text{正解した数}}{\text{予測した全データ数}} \tag{3.12}$$

実際に例の混同行列で計算すると、対角成分にある正しく予測された値の合計値 24 をデータの全体数である 30 で割った 0.80 が正解率となる。式 3.12 からもわかるように、正解率は 1 に近いほど正しい予測がなされたという見方になる。

ここまで正解率について説明を行ってきたが、本来、教師なし学習に当たるクラスター分析の評価はほぼ不可能とされている。なんらかの正解データをもとに分類を行っているわけではないことや、人間の判断基準ではわからない評価を含んでいる可能性もあり、

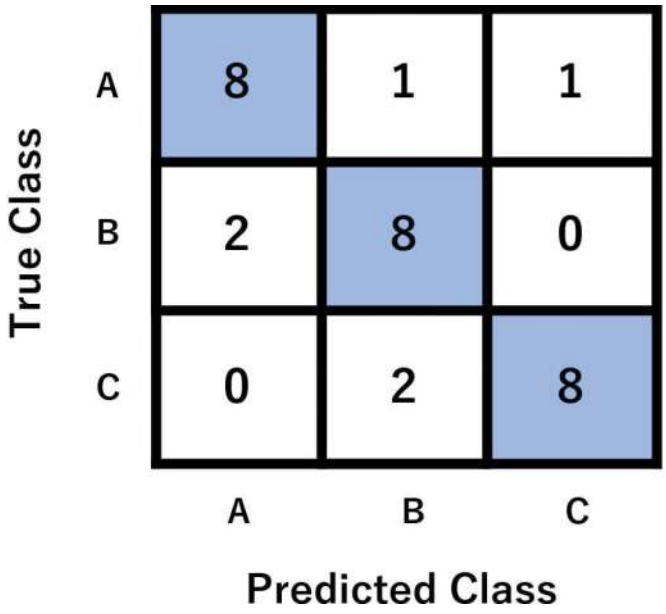

Figure 3.3　Confusion Matrix.

分析結果を見る側面によって良し悪しが変化するためである。そのため正解率も、通常はクラスター分析の評価に使用できるものではない。

しかし本研究では、あらかじめ正解ラベルの付いたデータを分析対象にすることで、疑似的に評価を導入できる環境を整えた。ここで、本研究における正解ラベルとは分析対象である小説のあらすじに割り振られた各ジャンルとなる。同じジャンルのあらすじなら、同一のクラスターとして分類され、異なるジャンルであれば別のクラスターに分類されるだろうという考えを適用したものである。これにより、分析手法の評価・比較が可能となる。

3.2　WebAPI

3.2.1　API

API とは Application Programming Interface の略称で、あるソフトウェアから他のソフトウェアを制御するためのインターフェース（規約）を意味する。何かしらの決められた API がシステムに存在する場合、それを介することでシステムの内部構造を深く理解することなく、その機能を呼び出すことが可能となる。API の目的には、ソフトウェア開発における開発工程の大幅な削減や開発における標準化、利便性の向上などが挙げられる。

3.2.2 WebAPI[3)]

本研究で言うWebAPIとは、HTTPプロトコルを利用してネットワーク越しに呼び出すAPIのことである。ユーザー側があるURIにアクセスすることで、サーバ側の情報の書き換えやサーバ側に保存されている情報を取得できることが可能なウェブシステムのことを指す。プログラムからアクセスすることで、そのデータを機械的に利用することなどに用いられることが多い。有名な例として、Googleが提供する各種APIやAmazonのProduct Advertising API、Twitter社が提供するTwitter APIなどが挙げられる。近年ではWebAPIを公開することの重要度が高くなっており、企業によってはサービスの価値や収益を左右するケースまでもが確認されている。そのため活用事例も多く、特にSNSやECサイト等での利用が多くみられる。

3.2.3 WebAPIを利用したテキストデータの取得[4)]

本研究では解析するテキストデータの取得にWebAPIを利用した。利用したAPIは、株式会社ヒナプロジェクトが運営する投稿型小説サイト「小説家になろう」に用意されたなろう小説APIというものである。このWebAPIは、ホームページやブログの管理者そして、システムエンジニア、プログラマに向けた各種技術情報の公開を目的として提供されている。いくつかのオプションを指定し、特定のURLにリクエストを行うことでWebサイトに投稿されている小説の情報が取得できるよう開発された。取得できる情報には、小説タイトル、小説のあらすじ、作者、小説の評価などが挙げられる。本研究では、複数のジャンルから小説タイトルとあらすじを対象として作品情報の取得を行った。

3.3 Python

3.3.1 Python[5)]

Pythonとは、グイド・ヴァン・ロッサム氏により開発された汎用プログラミング言語である。1991年に初のリリースがされ、現在では何百万人ものユーザーが利用しているとされている。Pythonの特徴として以下のような点が挙げられる。

- インタプリタ形式の、対話的な言語
- オブジェクト指向プログラミング言語
- 移植が容易で、多くのUnix系OS、Mac、Windowsで動作が可能
- オープンソースで運営されている

- コードの記述がシンプルであり、可読性が高いとされている
- 汎用的なライブラリから、専門的なライブラリまで豊富に用意されている。

こうした特徴から、人工知能をはじめとした様々な分野で活用されており、多くのユーザーから支持を得ている。

3.3.2 MeCab[6)]

本研究では、Python から MeCab を呼び出すことで形態素解析を行った。MeCab の辞書には、標準のシステム辞書ではなく mecab-ipadic-NEologd という辞書を採用した。

mecab-ipadic-NEologd は、新語・固有表現に強く、語彙数が多いオープンソースソフトウェアとして公開されている。本研究の解析対象である小説のあらすじは、現在も投稿・更新が盛んにおこなわれている小説投稿サイトのものである。そのため、デフォルトの辞書には登録のない新しい語や表現も使用されている可能性があると考えられた。

また、mecab-ipadic-NEologd は厳密には形態素解析用の辞書ではなく単語分かち書き用の辞書とされている。辞書に登録されている単語によっては、形態素まで分解されていないものもあるためである。しかし、語彙数や多くの固有表現を網羅していることから、今回の実験においてはこちらの辞書の方がより正確に文書を解析できると判断した。

そうした理由から、mecab-ipadic-NEologd を本研究に使用する辞書として導入した。

3.3.3 WebAPI

Python では Requests というライブラリを使用することで HTTP 通信を行うことができる。HTTP 通信では利用目的に応じてリクエストメソッドを指定し、実行を行う。本研究では API を介してデータの取得を行うため、プログラムでいくつかのオプションを指定した後、GET メソッドによる通信を行った。

3.3.4 TfIdf

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 TfIdf 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 (3.1)、(3.2) の通りに TfIdf が導出される。また TfIdf の出力以外にも、計算に使用した単語の一覧を出力することも可能である

3.3.5 cos 類似度

Python では scikit-learn ライブラリに用意されている cosinesimilarity 関数で cos 類似度の計算を行える。また、scipy というライブラリに用意されている pdist 関数では距離の公理に当てはめた cos 類似度の計算が行える。本来、cos 類似度は距離ではないため、距離として扱うためには別の計算が必要となる。

2 つのベクトルをベクトル $\vec{a}$ とベクトル $\vec{b}$ とすると、cos 類似度をもとにした距離は以下の式で導出される。

$$\cos\left(\vec{a},\vec{b}\right)_{distance} = 1 - \frac{\vec{a}\cdot\vec{b}}{|\vec{a}||\vec{b}|} \tag{3.13}$$

本研究では、文書間の類似度を値として確認する際に cosine-similarity 関数を使用し、pdist 関数はクラスター分析に与えるデータを計算する際に使用した。

3.3.6 主成分分析

Python では、scikit-learn ライブラリに用意されている PCA 関数で主成分分析を行うことができる。PCA 関数では、引数の n_components に削減後の次元数を指定することで主成分分析が行える。主成分分析の実行以外にも、各主成分の寄与率や累積寄与率といった値の出力も可能となっている。

3.3.7 潜在的意味解析

Python では、numpy というライブラリに用意されている svd 関数で特異値分解を行うことができる。この関数に TfIdf を与えることで、左特異（ターム）ベクトル、特異値、右特異（文書）ベクトルを取得できる。そうして出力された左特異（ターム）ベクトルと式を用いて潜在的意味解析を行った。

3.3.8 クラスター分析

Python では scipy ライブラリに用意されている linkage 関数で階層的クラスター分析を行うことができる。linkage 関数では、測定方法を指定し、pdist 関数で計算された距離行列を与えることでクラスター分析が実行できる。また分析を行ったデータに対して、fcluster という関数を使用すれば任意の数のクラスターに分類することが可能となる。クラスター分析した結果を樹形図として確認したい場合には、dendrogram 関数を使用すれば結果が出力される。

3.3.9 正解率

Python では、scikit-learn ライブラリに用意されている classification_report 関数に、分析データの正解リストと予測リストを与えることで正解率を計算できる。また混同行列は、同じく scikit-learn ライブラリの confusion_matrix 関数に classification_report 関数と同じデータを与えることで生成できる。

3.4 Word2vec

3.4.1 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・公開がされた。Word2vec を利用して単語をベクトル化することによって、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

具体的な例について以下の式を用いて説明する。Word2vec によって生成されたベクトル空間上に「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ことから、単語同士で次のような計算を行うことができる。

$$「\text{king}」 - 「\text{man}」 + 「\text{woman}」 = 「\text{queen}」 \tag{3.14}$$

式 3.14 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。

3.4.2 Word2vec による単語間の類似度計算

Word2vec では、学習済みモデルに対して単語を指定することで様々な操作が行える。その操作の中に、指定した単語のベクトル表現の取得がある。これにより、単語がベクトル空間上のどこに位置するかを数値で判断することができる。また、取得できる値はベクトルであることから、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基にしたクラスター分析を行うことが可能となる。本研究ではこれを利用して、解析対象の

文書に含まれる単語間の関係を導出した。

3.4.3 Word2vec を用いたクラスター分析による次元削減

本来、クラスター分析は次元削減を行う技術ではない。しかし、Word2vec による単語間の距離をもとにクラスター分析を行うことで、単語を複数のクラスターに分類することができる。ここで、Word2vec において計算される単語間の距離は、単語同士の意味の近さを表すものになる。そのため、その距離を利用して形成されたクラスターは、意味が似通った単語が集められたクラスターとなる。これにより単語の数だけあった次元を、似た意味を持つ単語が集められたクラスターの数まで削減することが可能となる。この手法をクラスター分析による次元削減とする。

TfIdf を重要度とした式 3.9 のような文書行列に、クラスター分析による次元削減を行うことで、クラスターと文書からなるクラスター-文書行列が作成される。このとき、クラスター-文書行列の重要度は、式 3.15 で表されるクラスターと名詞の行列と式 3.9 にある元の文書行列都の積で求められる。

$$CW = \left(\begin{array}{c|cccc} Term & w_1 & w_2 & \cdots & w_M \\ \hline C_1 & a_{1,1} & a_{1,2} & \cdots & a_{1,M} \\ C_2 & a_{2,1} & a_{2,2} & \cdots & a_{2,M} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ C_N & a_{N,1} & a_{N,2} & \cdots & a_{N,M} \end{array}\right) \tag{3.15}$$

各クラスターを $C_i(i = 1, 2, \ldots, N)$ 単語を $w_j(j = 1, 2, \ldots, M)$ 単語 w_k と単語 w_j の cos 類似度を $S_{k,j}$ とする。このとき、式 (3.15) にある要素 $a_{i,j}$ は次の式で与えられる。

$$a_{i,j} = \begin{cases} \dfrac{1}{|C_i|} \displaystyle\sum_{k \in C_i} S_{k,j} & (j \in C_i) \\ 0 & (j \notin C_i) \end{cases} \tag{3.16}$$

第4章　実験

4.1　実験の概要

本実験では小説のあらすじを対象として、テキスト間の類似度計算を行う。出力される類似度は、潜在的意味解析を使用したものとクラスター分析による次元削減を使用したものの2種類となる。類似度の計算は累積寄与率をもとに次元数を幾度か変更し、複数の結果を出力する。その後、出力された類似度を基にクラスター分析を行い、テキスト間の関係を比較する。比較の材料には、クラスター分析の結果をもとに生成されたデンドログラムと、結果から導出する正解率を用いる。なお、正解率は評価材料として妥当かどうかわかっていないため、その判定を行うことも踏まえての採用である。この比較により、クラスター分析による次元削減が従来手法である潜在的意味解析と比べ、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。

実験では、分析対象である小説のあらすじのジャンル数と作品数を4パターンに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。以下が各実験パターンの詳細である。

実験パターン1

・ジャンル数：3　作品数：30（各ジャンル10作品）

実験パターン2

・ジャンル数：3　作品数：150（各ジャンル50作品）

実験パターン3

・ジャンル数：6　作品数：60（各ジャンル10作品）

実験パターン4

・ジャンル数：6　作品数：300（各ジャンル50作品）

ジャンルの内容や各パターンについての詳細は4.2.4項で後述する。実験としては、上記4つのパターンのデンドログラムや正解率を比較することで考察を行うこととなる。

4.2　実験準備

4.2.1　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
- テキストデータの取得

4.2.2 Python, MeCab の導入

本実験では、Python の実行環境として Anaconda を導入した。Anaconda はデータサイエンス向けの環境を提供しており、科学技術計算などを中心とした、多くのモジュールやツールが準備されている。これにより簡単に Python の利用環境が構築できるとされている。

MeCab は、公式から配布されている 32bit 版ではなく、有志によって作成された 64bit 版の MeCab を使用した。これは導入した Python の 64bit とバージョンを合わせるためである。それに伴い辞書の mecab-ipadic-NEologd も、開発者である佐藤敏紀氏の Github ページより取得を行った。

4.2.3 Word2vec の学習済みモデルの取得 [7)]

単語のベクトルを取得するには Word2vec のモデルを作成する必要がある。しかし、モデルを作成するための学習には多大な時間を要する。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、2019 年に学習されたモデルを採用した。

4.2.4 テキストデータの取得

分析対象である小説のあらすじは、3.2.3 項で述べた WebAPI を利用することで取得した。取得内容は、小説の作品名とあらすじである。作品ごとに、作品名とあらすじを同じテキストファイルにまとめることで解析対象の 1 つとした。

作品の取得を行った「小説家になろう」サイトでは、いくつかのジャンルが存在しており、各作品に 1 つのジャンルが割り当てられている。本実験では、そうしたジャンルの中から以下の 6 ジャンルを取得する作品ジャンルとして採用した。

- ハイファンタジー 〔ファンタジー〕
- 現実世界 〔恋愛〕
- 推理 〔文芸〕
- ホラー 〔文芸〕
- 空想科学 〔SF〕
- 童話 〔その他〕

ジャンルを指定して作品を取得した理由は、クラスター分析による分類が理由として挙げられる。クラスター分析において、同じジャンルの作品は同じクラスターに分類される可能性が高いと考えられ、そうした分類結果が本研究の目的である手法評価につなげることができるためである。指定した 6 ジャンルの作品は、ジャンルごとに 50 作品、計 300 作品取得した。

取得した作品は、4.1 節で述べた 4 つの実験パターンに分割を行った。各パターンの詳細について順に説明していく。

まずはジャンル数についてである。実験パターン 1 と実験パターン 2 では、ジャンル数が 3 ジャンルとなっている。これは上記のジャンルにあるハイファンタジー〔ファンタジー〕、現実世界〔恋愛〕、推理〔文芸〕の 3 つとなる。この 3 つを採用した理由は、それぞれのジャンルの方向性がとりわけ異なっており、その違いが分析結果に影響を及ぼすのではないかと考えられたためである。実験パターン 3 と実験パターン 4 のジャンル数が 6 であるのは、上記の 6 つのジャンル全てを含んで解析を行うということになる。

次に作品数である。実験パターン 1 と実験パターン 3 では作品数が各ジャンル 10 作品となっている。これは、選択した各ジャンル 50 作品の中から、それぞれ 10 作品ずつを抽出して解析対象にしたものである。実験パターン 2 と実験パターン 4 では選択した各ジャンルにおいて、取得した 50 作品すべてを含めたものということになる。

以上の通りに、4 つの実験パターンを用意して実験を行った。

4.3 データの前処理

4.3.1 テキストデータの形態素解析

取得した小説のあらすじに対して、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。本実験ではテキスト 1 つの文量がさほど多くないことから、抽出する品詞を名詞（数以外)、動詞、形容詞の 3 つ

に設定し、それ以外の単語は削除した。これらの作業を行った結果、各実験パターン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 (単語種類数) … 1364 語
- 実験パターン 2 (単語種類数) … 4409 語
- 実験パターン 3 (単語種類数) … 2205 語
- 実験パターン 4 (単語種類数) … 7022 語

4.3.2 TfIdf の計算

Python を用いて TfIdf を計算する。計算は、3.3.4 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。ここで、計算が終了した後に 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。クラスター分析による次元削減を行う際に、TfIdf で使用された単語一覧が必要となるためである。

4.4 次元削減

4.4.1 実験パターンにおける主成分数の決定

本実験では前述した 2 つの次元削減手法を比較するために、主成分数の変更を何度か行って結果を出力する。主成分数の変更は累積寄与率に基づいて行う。3.3.6 項に示した PCA 関数で主成分分析を行い、累積寄与率が、50%から 55%、60%、65%と 5%ずつ間隔をあけて 90%に至るまでの主成分数を記録していく。そのため各実験パターンの手法ごとに 7 回次元削減を行うことになる。通常は 3.1.4 項にも示したように 60%〜80%の累積寄与率で削減数を決定することが多いが、本研究では次元削減への影響の確認や結果の値を多く取るため、累積寄与率を 50 %〜90 %という範囲に設定した。Table 4.1 に各実験パターンで主成分分析を行い、決定した主成分数をまとめたものを示す。

4.4.2 潜在的意味解析による次元削減

3.3.7 項に記述があるように、numpy ライブラリの svd 関数に TfIdf 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 Table 4.1 に示してあるように、各実験パターンの各累積寄与率に適する主成分数をもとに指定した。

Table 4.1　Dimensional quantity of each pattern.

cumulative contribution ratio	pattern 1	pattern 2	pattern 3	pattern 4
50%	9	40	15	70
55%	10	47	18	80
60%	12	53	21	95
65%	13	60	24	110
70%	15	68	27	125
75%	17	77	30	140
80%	19	87	35	160
85%	21	100	39	185
90%	23	110	43	210

4.4.3　クラスター分析による次元削減

クラスター分析による次元削減は以下の手順で実行した。

1. 単語の意味を、Word2vec を用いてベクトルとして抽出する
2. 抽出した単語のベクトルを基に、式 (3.4) で単語間の類似度を計算する
3. 単語間のクラスター分析を行い、Table 4.1 にある主成分の数に合わせて、クラスターの数を分割する
4. クラスタ内の総単語数を計算する
5. 各クラスターに含まれる単語を確認し、有無を数値として行列にまとめる
6. 2.，4.，5. を用いて、式（3.16）の値を求め、行列としてまとめる

1 では Word2vec を用いて単語の意味をベクトルとして取得している。しかし取得したい単語の中には Word2vec に登録されていない語も存在する。以下に各実験パターンごとの Word2vec 非登録語の数を示す。この非登録単語に関しては、単語間の類似度が計算できないため行列からこの単語のラベル部分を抜くことで対応した。

- 実験パターン 1 (Word2vec 非登録語) … 77 語
- 実験パターン 2 (Word2vec 非登録語) … 273 語
- 実験パターン 3 (Word2vec 非登録語) … 101 語

- 実験パターン 4 (Word2vec 非登録語) … 422 語

3 では単語のクラスター分析を行った後に、Table 4.1 の主成分数に合わせてクラスターの数を分割している。これは潜在的意味解析の次元数と次元を合わせ、なるべく同じ条件で比較を行うためである。

4.5 文書のクラスター分析

4.5.1 文書間の cos 類似度

次元削減された行列に対して、3.3.5 項で記述したように cosinesimilarity 関数と pdist 関数を用いることで 2 種類の cos 類似度を計算した。それぞれ cosinesimilarity 関数による cos 類似度は値の確認を行うために計算し、pdist 関数による値は、次に行うクラスター分析用に計算を行った。

4.5.2 クラスター分析

前項で計算した文書間の類似度を基にクラスター分析を行った。3.3.8 項にあるように階層的クラスター分析を実行し、距離の測定方法には 3.1.6 項で説明を行ったウォード法を指定した。

4.5.3 デンドログラムの出力

3.3.8 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。デンドログラムは、実験パターン 1 と実験パターン 3 に関するものを複数個出力し、確認に使用した。実験パターン 2 と実験パターン 4 に関しては文書の数が多く、目視でのデンドログラムによる確認が不可能であったため出力は行わなかった。

4.6 正解率の計算

4.6.1 クラスター分析結果の取得

4.5.2 項で出力した分析結果に対して fcluster 関数を使用し、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ーの分割を行う。実験パターン 1 と実験パターン 2 では 3 つのクラスター、実験パターン 3 と実験パターン 4 では 6 つのクラスターに分割することとなる。これは分割を行ったクラスターに各ジャンルを順に割り当てていき、ジャンルの偏りが発生しているかを調べるためである。次項で述べる正解率導出の手順

の一つでもある。

4.6.2　正解率の計算

各実験パターンの各手法でジャンル数分に分割されたクラスターに対して、3.3.9 項で述べた各ライブラリを用いて以下の手順で正解率を導出する。

1. 分割されたクラスターに、適当なジャンルを割り当てる。
2. confusion_matrix 関数で、割り当てられたジャンルに基づく混同行列を生成する
3. classification_report 関数で正解率を求める
4. 1. とは別のジャンルを各クラスターに割り当て、2.3. を行う。以後クラスターに対して割り当てるジャンルを繰り返し変更していき、すべてのパターンの正解率を求める
5. 求め終わった正解率の中で、最も正解率が高いものを選ぶ

これにより、どの程度各クラスター内でジャンルの偏りがあったかを正解率で確認することができる。これが手法評価の材料となる。

4.6.3　グラフの作成

計算された正解率を実験パターンごとに折れ線グラフにまとめる。内容は縦軸が正解率、横軸が累積寄与率となっており、次元削減数による正解率の推移を表している。潜在的意味解析による次元削減の正解率、クラスター分析の結果による正解率、2 つの手法の正解率の平均値が 1 つのグラフにまとめて記載されている。

4.7　実験結果

4.7.1　実験結果の評価方法

本実験では、導出した正解率を比較することでクラスター分析による次元削減手法の有効性を評価する。しかし、正解率が手法の評価として使用できるか判断がついていないため、実験パターン 1 と実験パターン 3 の出力結果を用いて、手法評価における正解率の妥当性を決定する。妥当性の判断には正解率のグラフとデンドログラムを用いる。正解率のグラフより同じ次元数で両手法の正解率が大きく離れている場所を見つけ、その時のデンドログラムと正解率の関係を手法ごとに確認する。これは正解率の値によって、デンドログラムのまとまりに差が出ているのではないかという考えのもと行うもの

である。正解率が低ければ、デンドログラムにジャンルごとのまとまりがなく、正解率が高ければ、ジャンルごとのまとまりができているだろうという予想のもと行う。実験パターン 1 と実験パターン 3 にした理由は、デンドログラムを可視化できるのがこの 2 つであったためである。

4.7.2 正解率の推移

実験パターン 1 の正解率を Figure 4.1、実験パターン 2 の正解率を Figure 4.2、実験パターン 3 の正解率を Figure 4.3、実験パターン 4 の正解率を Figure 4.4 に示す。各グラフより、パターン 1 とパターン 3 では、2 つの手法の正解率にさほど差が出ていないことがわかる。それに対し、パターン 2 では潜在的意味解析の正解率に比べ、クラスター分析による次元削減の正解率の方が高い水準となっていることがわかる。パターン 4 では、どの次元数においてもクラスター分析による次元削減の正解率の方が高い値となっていることが分かった。

同じジャンル数のパターン同士を比較すると、3 ジャンル、6 ジャンルどちらとも作品数の多いパターンの方がクラスター分析による次元削減の正解率が高いことがわかる。また、作品数が増えるほど全体の正解率も低下する傾向がグラフより読み取れる。

4.7.3 潜在的意味解析, クラスター分析による次元削減のデンドログラム

正解率の妥当性を確認するため、実験パターン 1,3 におけるデンドログラムをそれぞれ出力する。デンドログラムは正解率に最も差があるところを各手法ごとに出力するため、計 4 つのデンドログラムを生成することとなる。実験パターン 1 では、累積寄与率 70%において、潜在的意味解析による分析結果の正解率がクラスター分析による次元削減のものより高く、グラフの中で最も差が開いていることがわかる。実験パターン 3 では、累積寄与率 75 %において、クラスター分析による次元削減の正解率が潜在的意味解析のものより高い値であり、グラフの中で最も差が開いていることがわかる。このことから、実験パターン 1 の累積寄与率 70%における各手法のデンドログラム、実験パターン 3 の累積寄与率 75%における各手法のデンドログラムを出力する。その結果、Figure 4.5、Figure 4.6、Figure 4.7、Figure 4.8 のようなデンドログラムとなった。　各実験パターンのデンドログラム同士について比較を行う。まず実験パターン 1 からである。

Figure 4.5 の実験パターン 1 における潜在的意味解析による結果を確認すると、上部の

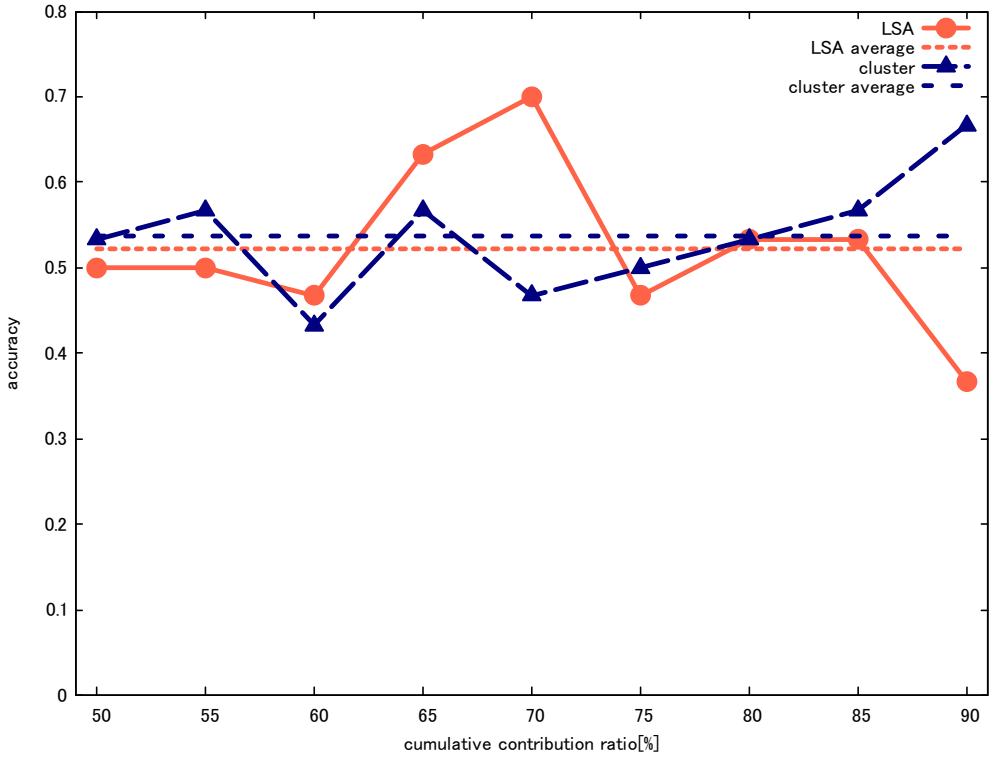

Figure 4.1　Accuracy of 3 genres and 30 works.

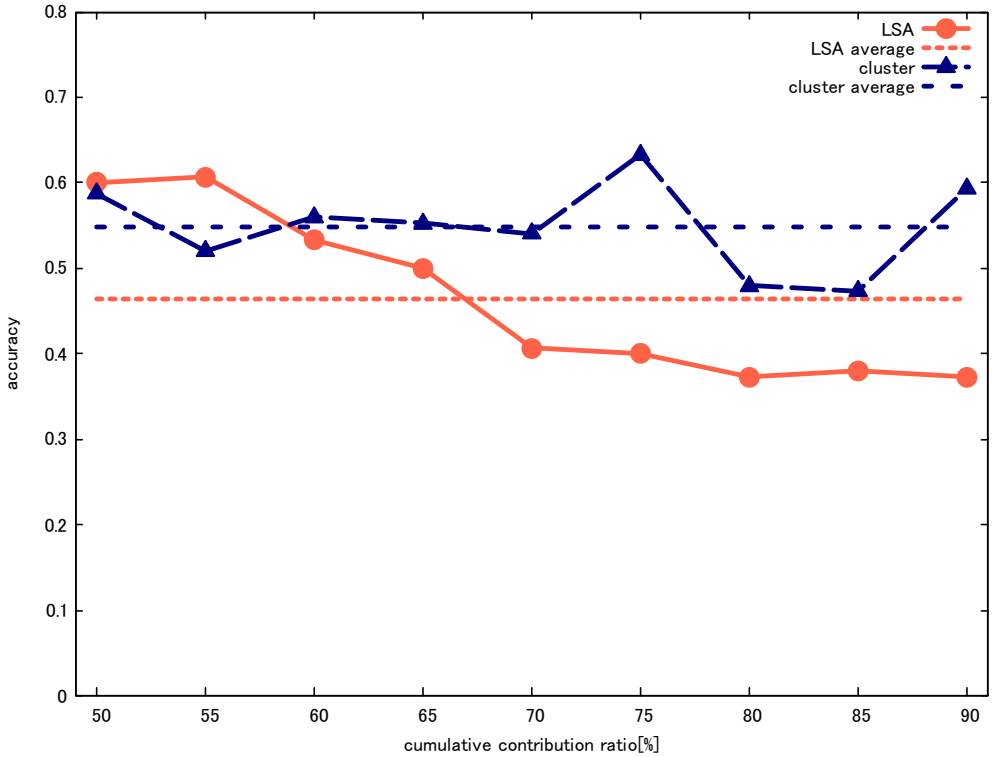

Figure 4.2　Accuracy of 3 genres and 150 works.

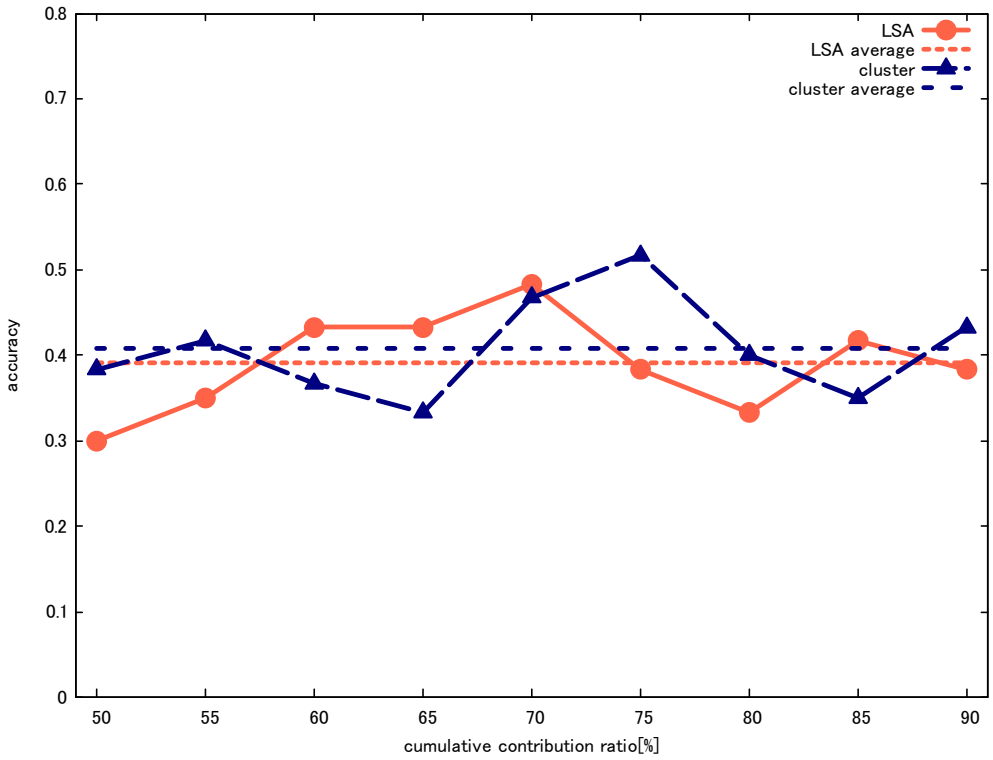

Figure 4.3　Accuracy of 6 genres and 60 works.

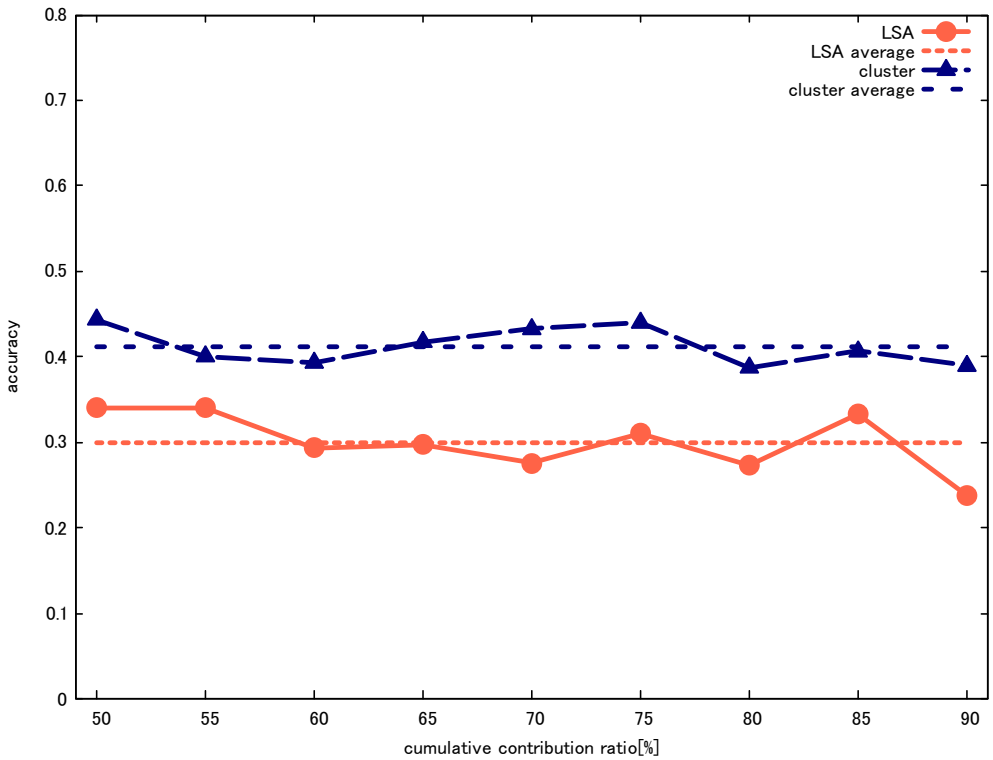

Figure 4.4　Accuracy of 6 genres and 300 works.

クラスターにおいてジャンルのまとまりが少しできており、下部のクラスターではファンタジーのクラスターと推理のクラスターができていることが分かる。次に Figure 4.6 のクラスター分析による次元削減のデンドログラムを確認する。下部のクラスターを見ると、全体的に 2〜4 の数でジャンル同士のクラスターが形成されていることがわかるが、それ以上の大きな数で同ジャンルのクラスターが形成されていないように見える。上部のクラスターに関しては、あまり同ジャンルでの作品のクラスターが形成されていないように見える。このことより、実験パターン 1 の累積寄与率 70%では潜在的意味解析の方がこちらの意図した通りに分類を行えている。

次に実験パターン 3 について比較を行う。Figure 4.7 の実験パターン 3 における潜在的意味解析による結果を確認すると、上部のクラスターでファンタジー、下部のクラスターで SF がまとまっていることが分かる。それに対して、Figure 4.8 クラスター分析による次元削減のデンドログラムでは上部のクラスターで SF と恋愛がまとまっている。一見すると、どちらも同程度のジャンル同士のまとまりを形成しているように見えるが、潜在的意味解析の下部において、別ジャンルの作品同士のクラスターが形成されていることが分かる。デンドログラムの結合点を見ると、これらのクラスターは別のクラスターからも離れた位置にあることが分かる。そういった影響を考えると、比較的にクラスター分析による次元削減の分析結果がこちらの意図した通りになったものと言える。

4.8 考察

4.8.1 手法評価における正解率適用の妥当性

4.7.3 項の結果より、本研究の手法評価における正解率の適用は概ね妥当であると考えられる。その理由としてデンドログラムと正解率の関係性が挙げられる。4.7.3 項でデンドログラムと正解率を比較した際に、正解率が高いときはジャンルごとにクラスターのまとまりが見られ、正解率が低いときは高いときに比べ、ジャンルごとのまとまりが見られなかったことがわかった。このことから、正解率が高いほど単語の意味を考慮し、関連性が近いものからクラスターを形成できていると考えられるためである。

しかし、正解率を適用する上での懸念点としてクラスターに含まれる要素数が均等ではないことが挙げられる。本研究では、ジャンルごとに均等な個数の小説のあらすじを分析対象として読み込ませたが、分析後に分割したクラスター内が必ずしも均等な数ではないためである。分析結果によってはクラスターの要素数に極端な偏りが生じてしま

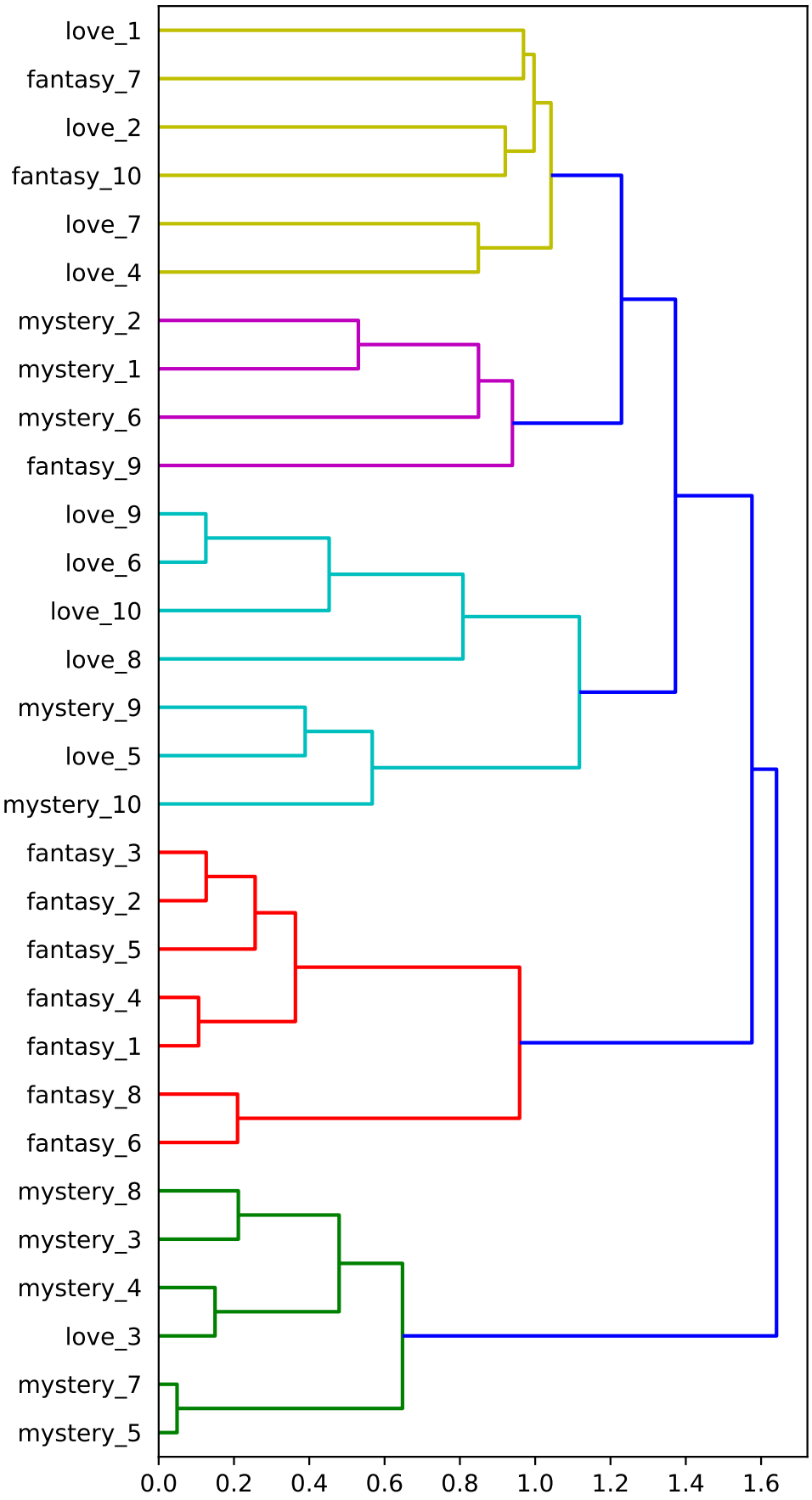

Figure 4.5 Result of dimension reduction by LSA(3_30_70%).

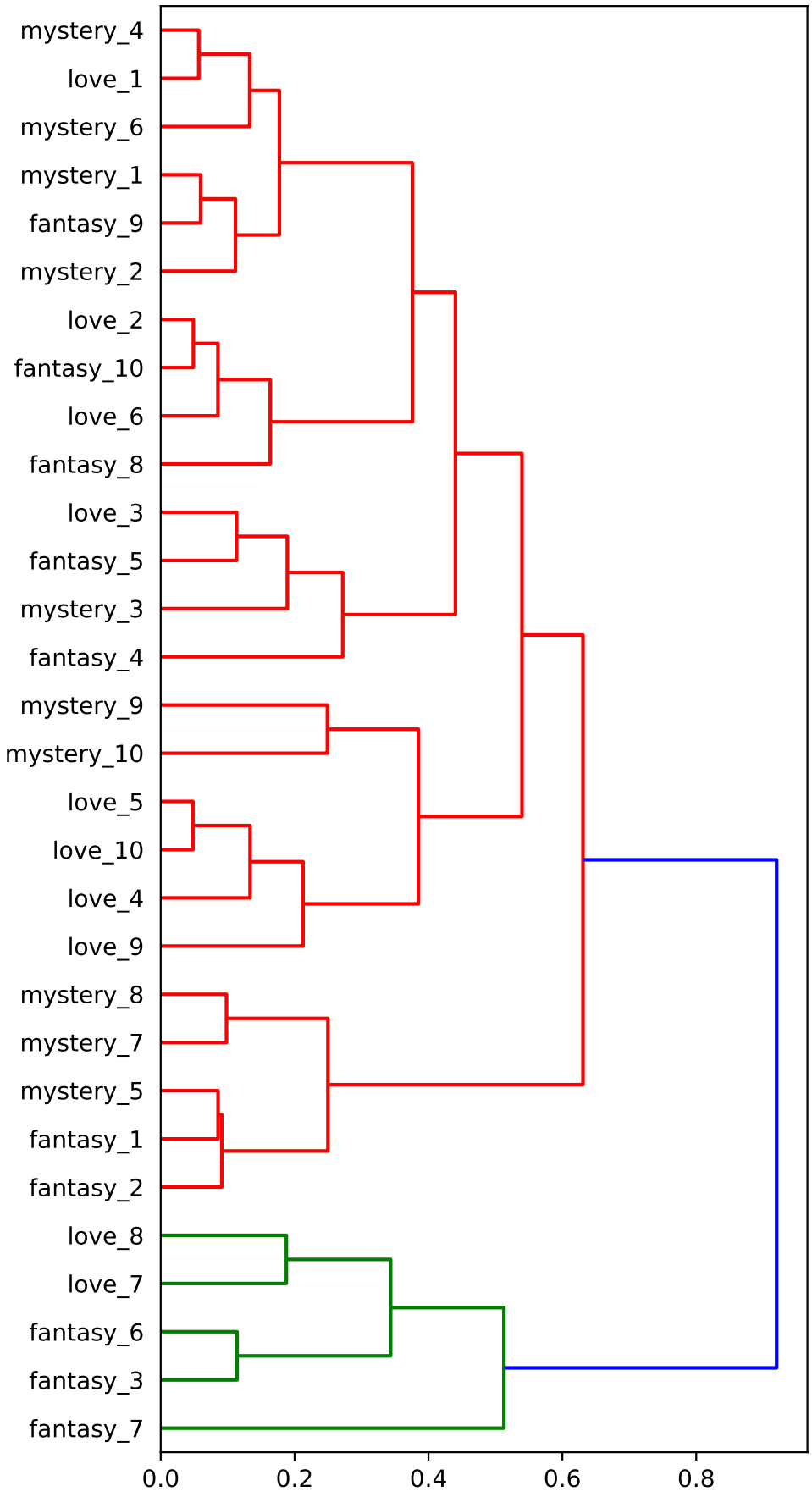

Figure 4.6 Result of dimension reduction by by cluster analysis(3_30_70%).

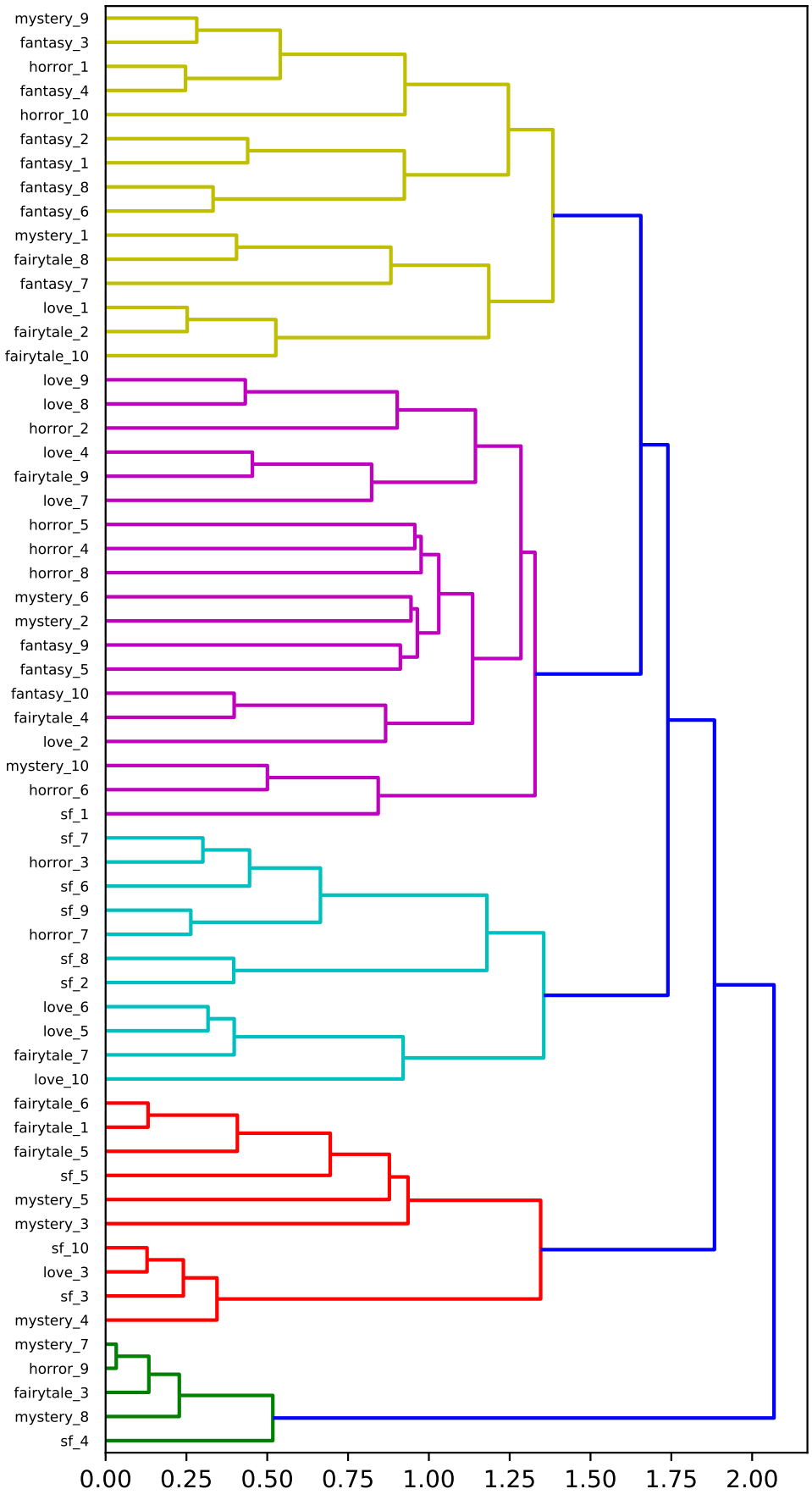

Figure 4.7 Result of dimension reduction by by LSA(6_60_75%).

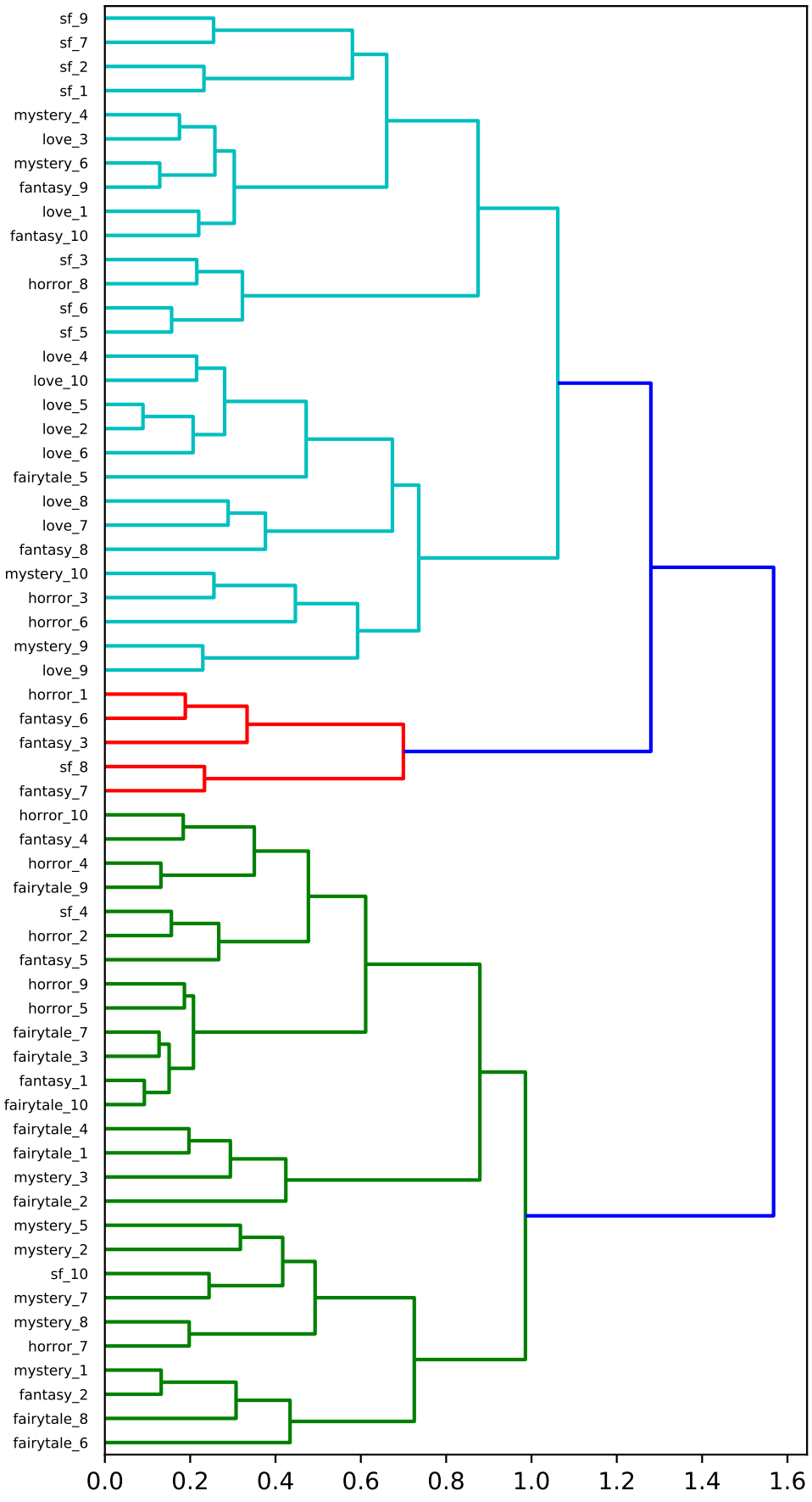

Figure 4.8 Result of dimension reduction by by cluster analysis(6_60_75%).

う可能性も存在する。そのため、正解率のみで判断するのではなく混同行列を用いてクラスター内の要素数や正解ラベルの偏りについても見る必要があると感じた。また、そういった偏りを考慮できる評価方法を見つけ、それと合わせて使用するのも良いかもしれない。

4.8.2 潜在的意味解析, クラスター分析による次元削減の比較

Figure 4.1〜Figure 4.4 と 4.8.1 項の考察より、クラスター分析による次元削減は作品数が増えるほど、ジャンルごとにまとまった分類をしていることが確認できた。特に実験パターン 2 や実験パターン 4 では潜在的意味解析を上回っていることがわかる。このような結果が得られた理由として分析対象である小説のあらすじの特徴が挙げられる。小説のあらすじは 1 つ 1 つのテキストの量は短いが、それぞれに作品を象徴するための単語が含まれていた。ジャンルごとに出現する単語を調べたところ、ファンタジーでは「世界」や「魔法」、推理では「事件」や「探偵」、恋愛では「幼馴染」や「彼女」といった象徴的な単語が何回か出現していた。また、単語の出現回数について調べたところ、上記のよく使われる特徴を表した語以外にあたるほとんどの単語は一桁の回数でしか出現していなかった。これは、小説のあらすじごとに様々な表現がなされていることや使用した形態素解析の辞書による影響が理由として挙げられる。そうした豊富な単語が出現してくるなか、それに対応する形で Word2vec からほとんどの単語のベクトルを取得できたことは、Word2vec を使用した大きな強みとして表れていた。以上のことからクラスター分析による次元削減において、単語間のクラスターを生成する際に意味が似通った多くの単語同士でクラスターが形成され、結果にも影響を及ぼしたと考えられる。

また上記の考察を含めると、分析対象と手法の相性も影響の可能性として挙げられる。潜在的意味解析は、単語の出現回数などから統計的な処理で次元削減をしており、単語自体の区別はついていない。そのため、テキストデータに捉えたい傾向とは別の単語が多く出てくると対処できないという側面がある。それに対し、クラスター分析による次元削減は、Word2vec を利用することで単語の区別を付けることが可能となり、単語間の関係をデータに付加できるという強みがある。こうした理由から、潜在的意味解析とクラスター分析による次元削減は実験パターン 2 や実験パターン 4 の結果を出したのだと考えられる。

第5章　結論

本研究では、小説のあらすじについて、従来手法である潜在的意味解析と本研究室で提案されたクラスター分析による次元削減を行い、類似度を計算した。これらの実験は、クラスター分析による次元削減の有効性を確認するために行った。

手順としては、初めに小説のあらすじの取得、形態素解析、TfIdf の導出、Word2vec を用いた単語間の類似度計算を行った。その後、潜在的意味解析とクラスター分析による次元削減を行い、分析結果をデンドログラムと正解率で表した。なお、正解率は評価材料として妥当かどうかわかっていないため、その判定を行うことも踏まえて導入した。以上の作業を、条件を変更した 4 つのパターンで行い、結果を比較した。

出力された結果を比較したところ、クラスター分析による次元削減の有効性を得ることができた。これは正解率の妥当性を確認した上でのことである。この結果が得られた理由として、テキストデータと次元削減手法の相性が挙げられた。クラスター分析による次元削減は潜在的意味解析と比べ、単語間の関係を付加価値として与えられるメリットがある。そのため、本研究の分析対象である小説のあらすじは、その付加価値が良く効くテキストデータであったと考えられ、それにより、良い結果が得られたのだと判断した。こうして、目的であったクラスター分析による次元削減の有効性の確認を行えた。

しかし、本実験を通して 3 つの懸念点が浮かんできた。1 つは正解率の適用についてである。デンドログラムと導出された正解率の比較により、ある程度の妥当性は確認できたがクラスター分析の結果によっては正常に機能するかわからない部分もまだ存在する。そのため、正解率とは別の有用な評価指標を見つけ、それと合わせて使うことでよりよい評価ができるかもしれない。2 つ目はクラスター分析による次元削減における単語間の距離の測り方である。今回は文書のクラスター分析で用いられる cos 類似度を単語のクラスター分析にも適用したが、他の測定方法については計算していない。そのため、測定方法によってはより良い結果が得られる可能性がある。最後の 3 つ目は、文書間のクラスター分析の手法である。本研究では階層的クラスター分析で分類を行ってきたが、他の分析手法は試していない。そのため他の分析手法を試して結果を比較することで、違う有用性を発見できるのではないかと考えられた。

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) 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
2) 有賀康顕 中山心太 西林孝 著, 仕事で始める機械学習, オライリージャパン, 2018.
3) 水野貴明 著, Web API: TheGood Parts, オライリージャパン, 2014.
4) 株式会社ヒナプロジェクト, なろうデベロッパー.
   https://dev.syosetu.com/（2020 年 12 月 31 日アクセス）.
5) 山内長承 著, Python によるテキストマイニング入門, オーム社, 2017.
6) Toshinori Sato, Neologism dictionary based on the language resources on the Web for Mecab, 2015.
   https://github.com/neologd/mecab-ipadic-neologd（2020 年 12 月 12 日アクセス）.
7) 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/（2020 年 10 月 28 日アクセス）.
8) 服部修平, テキストマイニングによる文書の類似度計算に関する研究, 岐阜工業高等専門学校電気情報工学科卒業研究報告, 2017.
   http://www.gifu-nct.ac.jp/elec/deguchi/sotsuron/hattori/
9) 長尾彪真, 文書の類似度計算における次元削減手法に関する研究, 岐阜工業高等専門学校電気情報工学科卒業研究報告, 2019
   http://www.gifu-nct.ac.jp/elec/deguchi/sotsuron/nagao/