卒 業 研 究 報 告 題 目

X-meansによる次元削減を用いた文書の類似度計算

Document Similarity Calculation using Dimension Reduction by X-means

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

2019E07 小川 楓葉

令 和 6 年（ 2 0 2 4 年 ）　　2 月 1 5 日 提 出

Abstract

This study proposes a dimension reduction method using Word2vec and X-means clustering. The document matrix is dimensionally reduced in two steps. Firstly, word clustering is performed using word vectors, to classify words. By multiplying the classification results with the document matrix, dimension reduction is performed. To evaluate the effectiveness of this method, documents were classified using the reduced document matrix. Three dimension reduction techniques were used for comparison with other methods.

- Latent Semantic Analysis
- Dimensionality reduction method using k-means clustering
- Dimensionality reduction method using x-means clustering

K-means clustering was chosen as the comparison method because it is the basic method for x-means clustering. Each method was compared in terms of accuracy, computation time and dendrogram. In this study, novel synopses and the first chapter of the text were used as the subjects of analysis. As the cumulative contribution ratio of latent semantic analysis is varied, Figure 1 shows the change in accuracy for each method.

The experimental results showed no significant difference in accuracy between k-means and x-means using k-means++. However, when pure x-means clustering was performed, the accuracy was comparable to k-means, while the computation time was significantly lower.

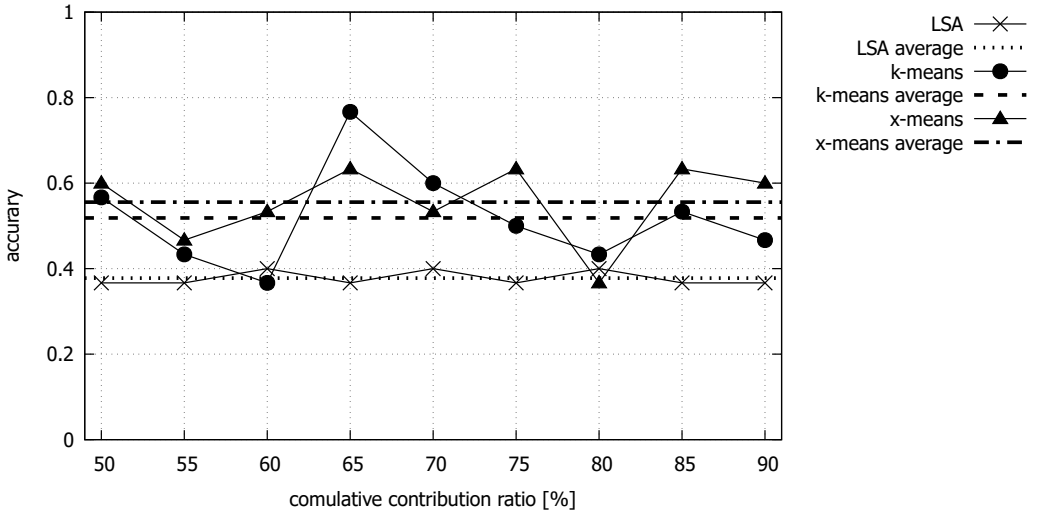

Figure 1　A result of the comparison experiments.

目 次

第1章　序論

インターネットが発展した現代社会では、情報が爆発的に増加し、あらゆるデータを簡単に発信、入手できる。しかしその一方で、信頼性の低い情報が混在してしまうという一面がある。この課題に対処するために、データマイニングと呼ばれる技術が生まれた。データマイニングは、情報端末から蓄積される膨大なデータの中から有益な情報を抽出する技術であり、特にテキストマイニングは言語処理技術を用いて大量のテキストデータを解析する手法である。統計学や人工知能を活用して有益なパターンやルールを発見し、個人が必要な情報を見つけ出す手助けをしている。しかし、テキストマイニングはまだ発展途上であり、課題も存在しているとされている。

本研究では、文書間の類似度計算をする際に用いら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、その有用性について検討することを目的とする。提案手法とは、x-means 法を利用したクラスタリングによる次元削減法である。先行研究にて、Word2vec によって得た単語ベクトルをクラスタリングすることにより、単語の数だけあった次元をクラスタ数にまで削減する方法が提案されていた。このクラスタリングに用いる手法として、x-means 法を実装する。x-means 法は k-means 法を改良した手法で、アルゴリズムによってクラスタ数を自ら決定することができる。この手法の評価を行うため、文書間の類似度計算を用いた文書分類を行う。事前にジャンル分けされたテキストを分類することで正解率を得て、これを評価指標とする。分析対象のテキストは、小説のタイトル、あらすじ、そして本文の第一章とした。また従来の手法との比較を行うため、以下の 3 つの手法で実験を行う。

- 潜在的意味解析
- Word2vec + k-means 法
- Word2vec + x-means 法

文書分類の正解率、次元削減の計算にかかった時間、文書間の距離を示したデンドログラムなどを結果として出力し、各手法を比較する。それによって提案手法の有用性の確認・検討を行う。

第2章　テキストデータの解析

2.1　膨大なデータの解析

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に対して、統計学や人工知能などを駆使した分析を行うことで、何らかの有益な知見を得るための手法である。これを用いて、ある事象の発生予測や新たな仮説を立てるなど、意思決定において重要な役割を果たす。人間では到底発見できない規則やルールを、コンピュータを用いることで高速かつ低コストで導き出すことができる。

2.1.2　テキストマイニング

テキストデータを対象にしたデータマイニングを、テキストマイニングという。非構造化データである自然言語を単語列に分解し、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。テキストマイニングの対象として、SNS の文章や顧客からのアンケート回答、コールセンターに寄せられる意見や質問などが用いられる。これらは他社商品との競合調査や、顧客満足度の分析などに役立てられる。

2.2　自然言語の解析

2.2.1　自然言語

自然言語とは、人間が日常的な意思疎通や情報の伝達、記述、思考を行うときに用いる言語のことである。「英語」「中国語」「日本語」といった文化的背景を持っておのずから発展してきた言語であり、構文や単語の用法に厳格な規則が存在しないと考えられるものを指す。この対義語として、プログラミング言語や論理式などを指す、「人工言語」「形式言語」が挙げられる。

自然言語と人工言語の明確な違いは、言葉の曖昧性にある。自然言語には、一つの文章に対して複数の解釈が生まれてしまうという曖昧性がある。これにより表現の柔軟性が非常に高いことが特徴となっている。

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の性質がある。

多義性とは、ある単語に対して複数の意味が存在しており、多様な解釈が生じることである。例として、「明るい」という単語を挙げる。「部屋が明るい」というように用いられた場合、光が十分にさしているというような意味になるが、「その道に明るい」という文章であると、専門的な話に精通しており、その方面に詳しいといった意味を表している。このように「明るい」という単語には二つの解釈が存在している。これは日本語だけではなく、「left －左、去る」のように、英語などの他の自然言語にも共通している性質である。

類義性とは、異なる単語が同じような意味を持っていることをいう。例えば「対等」と「互角」というように、全く違う文字と発音であっても、似たような意味をもつ言葉が多く存在している。このような曖昧性が、自然言語の分析を難易度を高くしている。

2.2.3　自然言語処理

自然言語処理とは、人間が自然言語を用いて作成した文章に対し、コンピュータに処理をさせる一連の技術のことである。この技術は機械翻訳や予測変換、テキストマイニングなどに応用されており、さらに開発が進んでいくことが期待されている。

2.2.4　形態素解析

形態素解析とは、自然言語処理の主要な技術の一つである。文章を形態素と呼ばれる意味を持つ最小の単位まで分割し、そのそれぞれに対して品詞や読みなどの情報を振り分ける。これにより、文章や単語よりも細かな意味を抽出することができ、コンピュータに解析させるための情報を手に入れることができる。

形態素解析を行うためのツールを形態素解析器と言い、これらのいくつか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。

2.2.5　MeCab[1)]

MeCab は京都大学と日本電信電話株式会社の共同研究を通じて開発された、オープンソースの形態素解析エンジンである。辞書、コーパスに依存しない汎用的な設計を基本方針とし、その他のオープンな形態素解析器である ChaSen, Juman, KAKASI より平均

的に高速に動作する。

MeCab での形態素解析は、オプションによって分かち書きや ChaSen 互換などの出力フォーマットに変更できる。ここでは例として、以下の文章をデフォルトのフォーマットで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

第3章　実験で用いた技術・手法

3.1　テキストデータの取得

3.1.1　API

API(Application Programming Interface) は、ソフトウェア間のインターフェースを規定するものであり、あるソフトウェアが他のソフトウェアを制御するための手段を提供する。システム内部の構造を深く理解することなく、API を介して機能を呼び出すことが可能で、これによりソフトウェア開発の効率向上や標準化、利便性の向上が図られる。

3.1.2　WebAPI[2)]

WebAPI とは、HTTP プロトコルを使用してネットワーク経由で呼び出される API のことを指す。ユーザーが特定の URL にアクセスすることで、ウェブシステムの情報の書き換えや取得が可能である。主にプログラムからアクセスされ、そのデータは機械的に利用される。Google や Amazon、Twitter などが提供する API がその例であり、近年では企業にとってサービスの価値や収益に影響を与える重要な要素となっている。SNS や EC サイトなどで広く利用されている。

3.1.3　WebAPI を利用したテキストデータの取得[3)]

本研究では、投稿型小説サイト「小説家になろう」に掲載されている小説のテキストを分析対象として得るため、なろう小説 API と呼ばれる WebAPI を用いた。この WebAPI は技術情報の公開を目的としており、特定の URL にリクエストを行うことで小説のタイトル、あらすじ、ジャンル、作者、評価などの情報を取得できる。本研究では、6 つのジャンルから小説タイトルとあらすじを対象としてデータの収集を行った。

3.2　自然言語のベクトル化

3.2.1　Tf-Idf

Tf-Idf とは、文書に含まれる各単語において、その単語が文書内でどれくらい重要かを示す値である。具体的には、文書内で単語がどれくらい多い頻度で出現するかを表す Tf(term frequency：単語頻度) 値と、全文書中でその単語を含む文書がどれくらい少ない頻度で存在するかを表す Idf(inverse document frequency：逆文書頻度) 値を掛け合わ

せた値である。数式的に表現すると、以下のようになる。

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

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

$$Tf\text{-}Idf_{ij} = Tf_{ij} \times Idf_{ij} \tag{3.3}$$

この計算を複数の文書間の全単語に対して行うことで、文書ごとに含まれている単語を用いたベクトルを得ることができる。

3.2.2 cos 類似度

cos 類似度とは、2 つのベクトルがどれくらい似ているか、類似性を表す指標の一つである。ベクトル間の cos 値を求めることによって、ベクトルがどれだけ同じ方向を示しているのかを確認できる。cos 類似度は -1 から 1 の値をとる。1 に近いほど 2 つのベクトルの方向は似ており、反対に -1 に近いほど真逆の方向を示している。よって 1 に近いほど類似性が高いといえる。また 0 の場合、2 つのベクトルは完全に直行しているため無関係である。ベクトル $\vec{a}$ とベクトル $\vec{b}$ の cos 類似度は以下の式で導出される。

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

3.2.3 cos 距離

cos 類似度は距離ではないため、クラスタリングの距離関数として用いるには、一手間加える必要がある。

ベクトル $\vec{a}$ とベクトル $\vec{b}$ の cos 距離は以下の式で導出される。

$$\cos(\vec{a}, \vec{b})_{distance} = 1 - \frac{\vec{a} \cdot \vec{b}}{|\vec{a}||\vec{b}|} \tag{3.5}$$

cos 類似度が 1、つまりベクトルの方向が同じであった場合、cos 距離は 0 となる。このように cos 距離はベクトル間の角度によるものであり、ベクトルの大きさとは無関係であるため、数学的な定義では距離とは異なるものとなっている。本研究では単語や文書のクラスタリングを行う際に距離関数として用いた。

3.3 次元削減のための技術

3.3.1 次元削減

次元削減とは、膨大な次元数をもつデータに対し、何らかの処理を有効に実行するため、データの意味をできるだけ損なわずに次元を削減する技術のことである。

3.2.1 項の Tf-Idf を例に挙げる。Tf-Idf は単語の数だけ次元数が存在し、また文書に含まれない単語の要素は 0 である性質上、疎な行列であるといえる。これに次元削減を行わずに類似度計算を行うと、膨大な計算量となることが予想される上に、有効な値を得られない可能性がある。次元削減によって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になる。

3.3.2 特異値分解 [4)]

任意の行列に対し、2 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ことを特異値分解という。以下に式を示す。

$$A = U\Sigma V^T \tag{3.6}$$

右辺の行列である U、V^T を左特異ベクトル、右特異ベクトルと呼び、それぞれ入力の行、列ベクトルの張る空間の正規直交基底を表す。本研究ではこの特異値分解を文書-単語行列である Tf-Idf に対して行ったため、それぞれ単語-トピック行列、トピック-文書行列となっている。この要素はトピックの重要度を示す。

3.3.3 潜在意味解析 [4)]

潜在意味解析 (Latent Semantic Analysis: LSA) とは、高次元の文書の行列に対して決定したトピック数まで次元削減する手法である。文書の分類や情報検索の分野などに使われるトピックモデルの代表例として知られている。文書行列に対して特異値分解を行い、各トピックの重要度が低いものから削減していき、設定したトピック数になるまでそれを繰り返すことで次元削減を実現する。これによって作成されたベクトル空間内では、近い概念は近くに、遠い概念は遠くにプロットされる。低次元に縮約することによって、疎らなデータやノイズが多いデータに対応できる。また、メモリに乗らない巨大なデータにも対応できる。

3.3.4　累積寄与率

次元削減において、削減する次元数の決定は重要なものである。次元数を過剰に削減すればデータの情報が大きく損なわれ、その後の処理の信頼性が失われる。反対に削減する次元数が少なすぎた場合、データが依然膨大であることによって計算量が大きくなったり、無駄な情報によって結果に影響が及ぼされたりする可能性がある。本研究では、この削減する次元数の決定に累積寄与率を用いた。寄与率というのはデータの重要度を示す値で、潜在意味解析ではトピックごとの重要度を示す。累積寄与率はこの寄与率の合計で、全てのデータでの累積寄与率は100%である。寄与率 P_n と累積寄与率 C の導出式を以下に示す。

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

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

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 (3.7) より、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。

3.3.5　クラスタリング

クラスタリングとは、異なるものが混ざり合ったデータから互いに似ているものをまとめ、それをクラスタと呼ばれる集団として形成し、対象を分類する手法である。この方法は教師なしの分類法であり、形や色などの具体的な基準があらかじめ定められていない点が他の分類手法と異なる。クラスタリングは主にデータの傾向を把握したい場合に使用される。

クラスタリングの手法は階層的クラスタリングと非階層的クラスタリングの2種類が存在する。階層的クラスタリングは、距離が短い者同士から順にクラスタを作成し、最終的には階層のようなクラスタ構造が形成される。形成されたクラスタの構造は、Figure 3.1 に示すようなデンドログラムによって視覚的に確認できる。縦軸はデータ間の距離を示す。クラスタ同士の距離の計算方法は複数存在し、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間の距離とする)
- 重心法 (クラスタ内のデータの中心をクラスタ間の距離とする)
- 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同士が結合する)

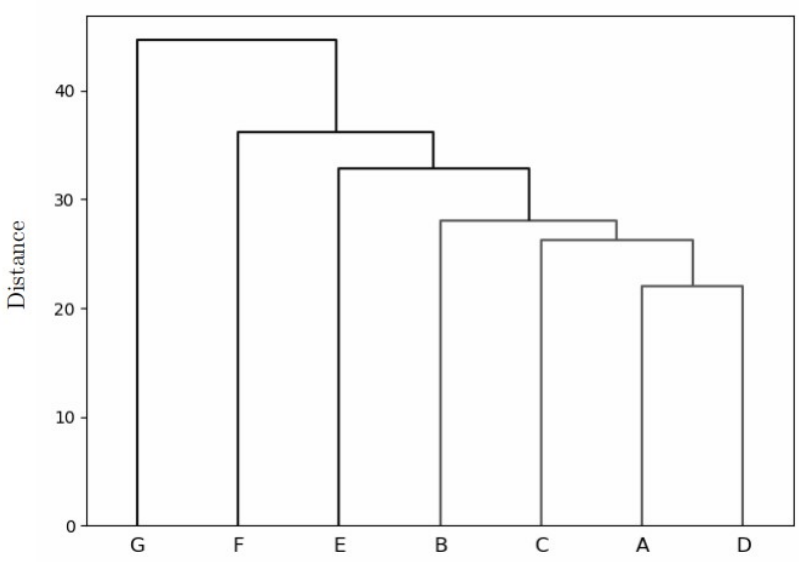

Figure 3.1　Dendrogram.

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に集まるように分割する手法であり、階層的な構造を持たないため、いくつのクラスタに分類するかを決定する必要がある。あらかじめクラスタ数を与える必要がある手法の例が k-means 法であり、自らクラスタ数を決定できる手法の例が x-means 法である。この 2 つの手法については、次項以降で詳しく記載する。本研究では次元削減に用いるクラスタリングとして k-means 法と x-means 法を用いた。また次元削減後の文書行列のクラスタリングには階層的クラスタリングであるウォード法を使用した。

3.3.6　k-means 法

k-means 法とは、非階層的クラスタリングの一つで、k 個のクラスタの中心を利用してクラスタに振り分けるアルゴリズムである。データ総数 n、クラスタ数 k のとき、以下の手順で実装される。

1. データ $x_i(i = 1, 2, ..., n)$ の中から k 個をランダムに選び、クラスタの中心 $V_j(j = 1, 2, ..., k)$ の初期値とする。

2. x_i と V_j の距離をそれぞれ計算し、最も距離が短いクラスタに x_i を所属させる。
3. 各クラスタの中心を計算し、V_j に振りなおす。
4. 手順 2, 3 を繰り返し、中心の移動が閾値以下になれば処理を終了する。

k-means 法のメリットは、計算量の少なさにある。階層的クラスタリングでは、各データ間の距離をすべて計算する必要があるが、k-means 法はデータとクラスタの中心との距離を求めるのみであるため、計算量が少なくなる。しかし、最初に選ばれる中心の初期値によって、結果が大きく異なったり、計算量が膨大になったりすることがある。これを克服するために、k-means++という手法が考案された。本研究ではこれを用いて初期値を決定した。3.3.7 項にて詳しく記載する。

また k-means 法には、あらかじめクラスタ数を決定しておく必要がある、という欠点がある。つまり、有効にクラスタリングを行うためには、何らかの方法を用いて最適なクラスタ数を探す必要がある。クラスタ数を変化させて k-means 法を繰り返し実行し、結果の誤差が最も大きくなったときを最適なクラスタ数とする方法が一般的である。本研究では、この欠点を排除するために考案された、x-means 法を使用した。3.3.8 項にて詳しく記載する。

3.3.7 k-means++法 [5)]

k-means++法とは、k-means 法の欠点である初期値依存問題の克服を目指して考案されたアルゴリズムである。初期のクラスタの中心は互いに離れていた方がよいという考え方に基づき、データ間の距離に応じて確率的にクラスタの中心を決定する。最適な k-means 法の解に比べて、$O(\log k)$ の近似比率で解を得ることが保証されている。データ総数 n、クラスタ数 k のとき、以下の手順で実装される。

1. データ $x_i(i = 1, 2, ..., n)$ の中から 1 つをランダムに選び、クラスタの中心とする。
2. 全データ x_i に対して、最も近いクラスタの中心との距離 $D(x_i)$ を計算する。
3. 式 (3.9) に示す重み付き確率分布を用いて、新しいクラスタ中心をランダムに選択する。

$$\frac{Dx^2}{\sum D(x_i)^2} \tag{3.9}$$

4. 手順 2, 3 を k 個のクラスタの中心が選定されるまで繰り返す。

式 (3.9) を見ればわかる通り、中心との距離 $D(x_i)$ が最も大きいデータが選択されやすく

なっている。このため、クラスタ間の距離を遠ざけることができる。しかし中心をランダムに選択している都合上、初期値依存問題を完全には克服できていない。

3.3.8 x-means 法[6)]

3.3.6 項で記述したように、k-means 法にはクラスタ数を変化させて繰り返し行うことで、最適なクラスタ数を発見する方法がある。しかしこの方法では、k-means 法のもつ計算量の少なさというメリットが失われてしまう。そこで考案された手法が、情報量基準の一つである BIC(Bayesian Information Criterion) を利用することでクラスタ数を決定する、x-means 法である。x-means 法という名称は、k-means におけるクラスタ数 k が未知であることに由来する。BIC は回帰モデルが多くの項を含みすぎることに対してペナルティを課す情報量基準で、値が小さいほど良いモデルであることを表す。データが n 個であるとき、以下の手順で実装される。

1. 十分に小さなクラスタ数の初期値 k_0(特に指定しなければ 2) を定める。
2. $k = k_0$ として k-means 法を適用し、分割後のクラスタを $C_i(i = 1, 2, ..., k_0)$ とする。
3. 各クラスタ C_i に対して $k = 2$ として k-means 法を適用する。分割後のクラスタを C_i^1, C_i^2 とする。
4. C_i に含まれるデータは、クラスタの中心近くにガウス分布していると仮定して BIC を計算し、B とおく。
5. 2 分割モデルにおける BIC を計算し、B' とおく。
6. $B > B'$ ならば、2 分割モデルをより好ましいと判断し、2 分割を継続すべく C_i^1, C_i^2 に対しても手順 3〜7 を行う。
7. $B \leq B'$ ならば、2 分割しないモデルをより好ましいと判断し、2 分割を停止する。

x-means 法は、全てのクラスタに対して k-means 法で 2 分割をしているため、k-means 法の 2 倍の計算量が必要になる。なお、BIC の計算量は全体に比較すれば微小であるため、ここでは無視する。k-means 法で最適なクラスタ数 k を発見的に見つけるためには、$k-1, k, k+1$ の少なくとも 3 点による評価が必要である。このことを考慮すれば、x-means 法は計算コストに見合った成果を提示するといえる。

x-means 法はクラスタ数を発見しなければならないという k-means 法の欠点を排除したが、初期値依存性は引き継いでいる。よって計算を行うたびにクラスタは少しずつ変化するが、クラスタ数は安定するという特徴がある。この初期値依存性を克服する方法

には k-means++法によってクラスタ中心の初期値を決定する方法があるが、この方法をとる場合クラスタ数を設定する必要がある。よって本研究では、k-means++法を用いて初期値依存性を克服した x-means 法と、クラスタ数を与えない純粋な x-means 法の 2 種類の実験を行った。

3.3.9 正解率

正解率とは、機械学習の分類問題に用いられる評価指標の一つである。正解率のほかに適合率、再現率、特異率などがあり、それぞれ分類の結果をまとめた混合行列を用いて計算される。混合行列とは Table 3.1 に示すようなもののことを指す。

Table 3.1 Confusion Matrix.

		Predicted Values
Actual Values	TP (True Positive)	FN (Flase Nagative)
	FP (False Positive)	TN (True Negative)

TP は正だと予測し実際も正であったとき、FN は負だと予測し実際は正であったとき、というように表されている。ここで正解率は、以下の導出式で計算される。

$$\text{正解率} = \frac{TP + TN}{TP + FP + FN + TN} \tag{3.10}$$

式の通り、正解率とは全体のデータの中で予測がどれだけ的中したのかを示す値である。

評価指標について説明したが、教師なし学習にあたるクラスタリングは、本来評価をすること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うえに、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があり、結果を見る角度によって良し悪しが変化するためである。正解率の計算を行う際にも、正解データが必要であるため、通常クラスタリングの評価に使用できるものではない。

そこで本研究では、あらかじめ正解ラベルのついたデータを分析対象とすることで、正解データを疑似的に得られる環境を整えた。この正解ラベルというのが、分析対象である小説に割り振られたジャンルである。同じジャンルの小説であれば、クラスタリングにおいて同一のクラスタ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、各手法の評価・比較を可能とした。

3.4 単語ベクトル

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として表現する手法である。Word2vec から単語ベクトルを得ることによって、単語同士の加減算や単語間の類似度計算が可能になる。単語同士の加減算の例を以下に挙げる。

$$\text{King} - \text{Masculinity} + \text{Femininity} = \text{Queen} \tag{3.11}$$

Word2vec による単語ベクトルはその意味によって方向や大きさが決められており、似た意味の単語はベクトルも似た値となっている。よって単語間の類似度計算において、似た意味の単語同士は 1 に近い値が得られる。

3.4.2 Word2vec を用いたクラスタリングによる次元削減

本来、クラスタリングは次元削減を行うための手法ではない。しかし、単語ベクトルによる単語間の距離を用いてクラスタリングを行うことで、単語をクラスタに分類することができる。このとき、単語間の距離として cos 距離を用いることにより、距離が近いほど単語の意味が似ていると考えることができる。よってこれにより形成されるクラスタは、意味が似通った単語が集められる。これを用いて、単語数だけあった Tf-Idf の次元数をクラスタ数にまで削減することができる。これをクラスタリングによる次元削減とする。

Tf-Idf を重要度とした文書行列に、クラスタリングによる次元削減を行うことで、クラスタ-文書行列が作成される。このとき、クラスタ-文書行列の重要度は、式 (3.12) で表されるクラスタと名詞の行列と文書行列である Tf-Idf との積で求められる。

$$CW = \left(\begin{array}{c|cccc} Term & w_1 & w_2 & \cdots & w_M \\ \hline C_1 & w_{1,1} & w_{1,2} & \cdots & w_{1,M} \\ C_2 & w_{2,1} & w_{2,2} & \cdots & w_{2,M} \\ \vdots & \cdots & \cdots & \ddots & \cdots \\ C_N & w_{N,1} & w_{N,2} & \cdots & w_{N,M} \end{array}\right) \tag{3.12}$$

各クラスタを $C_i(i = 1, 2, ..., N)$ 単語を $w_k(k = 1, 2, ..., M)$ と単語 w_j の cos 類似度を

$S_{k,j}$ とする。このとき、式 (3.12) にある要素 $a_{i,j}$ は次の式で与えられる。

$$a_{i,j} = \begin{cases} \dfrac{1}{|C_i|} \displaystyle\sum_{k \in C_i} S_{k,j} & (j \in C_i) \\ 0 & (j \notin C_i) \end{cases} \tag{3.13}$$

3.5 Pythonによる各技術の動作方法

3.5.1 Python[7)]

Pythonとは、グイド・ヴァン・ロッサム氏にとって開発された、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、現在では何百人ものユーザ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
- オブジェクト指向の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
- 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さまざまな分野で活用されており、多くのユーザから支持を得ている。

3.5.2 MeCab[1)]

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、開発者から推奨されているunidicを利用した。

3.5.3 WebAPI

Pythonでは、requests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。ライブラリ内にあるget関数に、引数として検索条件や出力内容などのオプションを与え、通信を行った。

3.5.4 Tf-Idf

scikit-learn ライブラリの TfidfVectorizer 関数によってモデルを作成し、さらに fit_transform 関数に文書ごとの単語配列を与えることによって、Tf-Idf を得ることができる。また Tf-Idf の出力以外に、計算に使用した単語の一覧を出力することができる。

3.5.5 cos 距離

cos 距離を求めるためには、式 (3.5) にあるようなベクトルの計算をする必要がある。Python では、numpy ライブラリを用いて、ベクトルの演算を実行できる。

3.5.6 潜在意味解析

scikit-learn ライブラリの TruncatedSVD 関数を用いることで、特異値分解、および潜在意味解析を実装することができる。

3.5.7 階層的クラスタリング

scipy ライブラリの linkage 関数を利用することで、階層的クラスタリングを実装できる。linkage 関数に、距離関数を指定した pdist 関数とメソッド名を与えることで実行される。また同ライブラリの dendorogram 関数に linekage 関数の戻り値を与えることで、デンドログラムを取得できる。クラスタは、fcluster 関数に対して、同じく linkage 関数の戻り値とクラスタ数を与えることで得られる。

3.5.8 k-means++法 [5)]

pyclustering ライブラリの kmeans_plusplus_initializer 関数を用いて実装できる。この関数の引数に、クラスタリングしたいデータとクラスタ数を与えることで実行される。アルゴリズムは 3.3.7 項で述べたものと同じであるが、オプションによって 2 つ目のクラスタ中心を、1 つ目のクラスタ中心から最も遠いデータとすることもできる。本研究ではオプションは加えていない。

3.5.9 k-means 法 [5)]

pyclustering ライブラリの kmeans 関数によって実装できる。scikit-learn ライブラリもまた Kmeans 関数が存在するが、オプションで距離関数を変更できる前者を使用した。

3.5.8 項からクラスタ中心の初期値を得て、データとともに kmeans 関数に与えることで実行される。

3.5.10 x-means 法[5)]

pyclustering ライブラリの xmeans 関数によって実装できる。xmeans 関数の引数として、データと 3.5.8 項で得たクラスタ中心の初期値を与えることで、k-means++法と x-means 法を実行できる。またクラスタ中心の初期値を与えなかった場合、ランダムに初期値が選ばれ、純粋な x-means 法が実行される。

3.5.11 正解率

scikit-learn ライブラリの accuracy_score 関数に、正解データと予測データの 2 つの配列を与えることで、正解率を得ることができる。

第4章　実験

4.1　実験の概要

本実験では、以下の3種類の次元削減法による文書分類を行う。

- 潜在意味解析
- Word2vec + k-means 法
- Word2vec + x-means 法

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 x-means 法は k-means 法を改良した手法であるため、結果を比較することで優位性を検証する。潜在意味解析の際の累積寄与率をもとにいくつかの次元数ごとに次元削減を行い、類似度を計算する。k-means 法、x-means 法では初期値依存性の改善のため、k-means++法によって求めた初期値を利用する。距離関数は cos 距離とし、クラスタリングを行うことで次元削減を行う。またこれを用いて累積寄与率と正解率の関係について検証する。x-means 法については、計算終了後のクラスタ数の変化も結果として出力する。また k-means 法と x-means 法の計算時間の比較を行う。

6つのジャンルに分けられている小説を用いて、文書分類を行う。いくつかの実験パターンに分けて行うことで、ジャンルの組み合わせやジャンル数の違いによる情報量の変化が結果にどのような影響を及ぼすのか確認する。実験パターンは以下の6つとした。作品数は正解率にあまり関係がないことが先行研究[8)]にて明らかになっているため、今回は各ジャンルすべて10作品とした。

実験パターン1

- ジャンル数: 3 (童話, ホラー, 恋愛)　　作品数: 30 (各ジャンル10作品)
  タイトル + あらすじ

実験パターン2

- ジャンル数: 3 (童話, ホラー, 恋愛)　　作品数: 30 (各ジャンル10作品)
  本文の第一章

実験パターン3

- ジャンル数: 3 (SF, 恋愛, 推理)　　作品数: 30 (各ジャンル10作品)
  タイトル + あらすじ

実験パターン 4

- ジャンル数: 3 (SF, 恋愛, 推理)　　作品数: 30 (各ジャンル 10 作品)

  本文の第一章

実験パターン 5

- ジャンル数: 6　　作品数: 60 (各ジャンル 10 作品)

  タイトル + あらすじ

実験パターン 6

- ジャンル数: 6　　作品数: 60 (各ジャンル 10 作品)

  本文の第一章

各実験パターンの詳細については、4.2.4 項で記述する。上記の実験パターンで各手法のデンドログラムや正解率、計算時間を比較する。

4.2　実験準備

4.2.1　実験環境の構築

本実験では Google が提供しているサービスである、機械学習、データ分析に適した Google Colaboratory というプラットフォームを利用した。ブラウザ上で直接 Python を動かすことができ、インストールなしで Python の開発環境を構築できる。Python のバージョンは 3.10.12 を用いた。

さらに本実験に必要な環境構築のため、以下を行った。

- 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　MeCab の導入

MeCab を Python で動作させるため、mecab-python3 というラッパを用いて導入した。また辞書として、MeCab の開発者から推奨されている unidic を用いた。

4.2.3　Word2vec の学習済みモデルの取得 [9)]

本実験の Word2vec は、東北大学の乾・岡崎研究室で作られたモデルを使用した。このモデルは日本語 Wikipedia の本文をもとに学習したもので、一般の単語だけでなく、人

名や地名といった固有表現が考慮されるよう学習されている。モデルには 100, 200, 300 次元の 3 種類があるが、先行研究 [10] より Word2Vec モデルの次元数と正解率とに正の相関関係があることが分かったため、300 次元のモデルを採用した。また、これを gensim の KeyedVectors を用いて Python に実装した。

4.2.4 テキストデータの取得 [3]

「小説家になろう」というサイトに掲載されている小説を、WebAPI とスクレイピングを用いて取得した。分析対象はタイトルとあらすじ、本文の第一章とした。ジャンルごとに CSV ファイルを作成し、各作品のタイトルとあらすじをひとつのセルにまとめて書き込み、分析対象とした。本文の第一章はまた別のファイルとして、ジャンルごとにまとめて作成した。あらすじには「大賞受賞」や「備考」などといった、小説の内容と関係のない文章が含まれているものも存在していたため、それらの文章はなるべく省いて取得を行った。

「小説家になろう」では、いくつかのジャンルに作品が分類されており、この中から以下の 6 つのジャンルを、それぞれ 10 作品ずつ、合計 60 作品のタイトル、あらすじ、本文の第一章を取得した。

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

同じジャンルに分類されている文書は、クラスタリングにおける文書分類においても同じクラスタになることが予想される。よって、このような事前にジャンルの分かっている文書を分析対象とすることで、クラスタリングによる文書分類の正解率を求めることが可能になる。上記の 6 つのジャンルは、使われる単語のベクトルが離れていることが予想され、分類が成功しやすいと考え、これらを採用した。

各実験パターンの詳細を Table 4.1 に示す。実験パターン 1 から 4 では、上記 6 つのジャンルの中から 3 つを選定し、各 10 作品、合計 30 作品を用いて文書分類を行った。3 つのジャンルは、潜在意味解析による文書分類の正解率によって決定した。正解率の平

均が最も高くなった童話, ホラー, 恋愛の組み合わせを実験パターン 1, 2 とし、正解率の平均が最も低くなった SF, 恋愛, 推理の組み合わせを実験パターン 3, 4 とした。潜在意味解析の得意とするジャンルの組み合わせと、クラスタリング法の得意とする組み合わせが異なる可能性を考え、この実験パターンを用意した。実験パターン 5, 6 は上記ジャンル全てを分析対象とし、ジャンル数が多くなった時の正解率の変化を確認する。そして実験パターン 2, 4, 6 はタイトルとあらすじではなく、小説本文の第一章の部分を分析対象としている。あらすじよりも使用されている単語数が多く、一つの作品に対する情報量が増加するため、このときの結果の違いを比較することが目的である。以上 6 つの実験パターンで文書分類を行った。

Table 4.1　Details of experimental Patterns.

Pattern1	Pattern2	Pattern3	Pattern4	Pattern5	Pattern6
3 genres Fairytale Horror Love	3 genres Fairytale Horror Love	3 genres SF Love Mistery	3 genres SF Love Mistery	6 genres	6 genres
Title Synopsis	Chapter 1	Title Synopsis	Chapter 1	Title Synopsis	Chapter 1

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

取得したテキストデータを MeCab を用いて形態素解析を行った。形態素に分解されたものからさらに、名詞 (数詞以外), 形容詞, 動詞のみを抽出した。それ以外の副詞や接続詞には、分類するために必要な情報が含まれていないためである。

4.3.2　Word2vec によるベクトル取得

抽出した単語のベクトルを、Word2vec を用いて取得した。いくつかの単語は本実験で使用した Word2vec モデルに存在しなかったため、情報量は実際の文書より少なくなった。各実験パターンの単語数は以下のようになった。また作品 1 つあたりの情報量を確認するために、各作品の単語数の平均を示す。

- 実験パターン 1...905 単語　(各作品の単語数 平均: 約 57.7 単語)
- 実験パターン 2...7722 単語　(各作品の単語数 平均: 約 1304.9 単語)
- 実験パターン 3...1134 単語　(各作品の単語数 平均: 約 71.3 単語)
- 実験パターン 4...6166 単語　(各作品の単語数 平均: 約 861.3 単語)
- 実験パターン 5...1602 単語　(各作品の単語数 平均: 約 56.2 単語)
- 実験パターン 6...11019 単語　(各作品の単語数 平均: 約 1081.6 単語)

4.3.3　Tf-Idf の計算

文書ごとに抽出した単語を用いて、Tf-Idf の計算を行った。scikit-learn ライブラリの fidfVectorizer 関数を用いてモデルを作成し、fit_transform に単語行列を与えることで求めた。

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、3 つの次元削減法の比較を行うために、いくつかの主成分数に分けて結果を出力した。主成分数は、潜在意味解析で次元削減を行った際の累積寄与率に基づいて決定した。累積寄与率が 50%から 90%まで、5%ずつ間隔を空けて主成分数を取得した。一般的に、次元削減の際には 60%から 80%の累積寄与率が選択されるが、次元削減の影響を確認し、より多くの結果を取得するために 50%から 90%の範囲に設定した。scikit-learn ライブラリの TruncatedSVD 関数でモデルを作成し、fit 関数に引数として Tf-Idf の値を与えることで、特異値分解や潜在意味解析を行うことができる。また explained_variance_ratio_ を用いることで寄与率を取得できる。Table 4.2 に累積寄与率に対する各実験パターンの主成分数を示す。

Table 4.2　Dimensional quantity of each Pattern.

cumulative contribution ratio	Pattern1	Pattern2	Pattern3	Pattern4	Pattern5	Pattern6
50%	8	6	10	8	18	14
55%	9	7	11	9	20	17
60%	10	8	12	11	23	19
65%	12	10	14	12	26	22
70%	13	11	15	14	28	26
75%	14	13	17	15	32	30
80%	16	15	19	17	35	34
85%	18	18	20	19	39	38
90%	20	21	22	21	44	44

4.4.2　潜在意味解析による次元削減 [11)]

前項で行った際と同様に、scikit-learn ライブラリの TruncatedSVD 関数を用いて潜在意味解析を行い、決定した主成分数になるようにトピック数を設定した。

4.4.3　k-means 法による次元削減 [5)]

k-means 法を用いた次元削減を、以下の手順で行った。

1. Word2vec によって得た単語ベクトルを用いて、cos 距離を距離関数とした k-means 法でクラスタリングする。クラスタ数は 4.4.1 項で決定した主成分数と同じとする。
2. 得られたクラスタと単語間の cos 類似度を用いて単語 – クラスタ行列を作成する。
3. 作成した行列と Tf-Idf の積を計算する。

k-means 法によるクラスタリングを、pyclustering ライブラリの kmeans 関数によって実行した。scikit-learn ライブラリもまた Kmeans 関数が存在するが、オプションで距離関数を変更できる前者を使用した。

初めに、同じく pyclustering ライブラリの kmeans_plusplus_initializer 関数に単語ベクトルとクラスタ数を与え、各クラスタ中心の初期値を得る。このクラスタ中心と、任意の距離関数、そして単語ベクトルを kmeans 関数に与えることで、クラスタリングを実

行できる。また predict 関数を用いることで、各単語が分類されたクラスタを得ることができる。

4.4.4 x-means 法による次元削減 [5)]

x-means 法は k-means 法と同様の手順で行った。kmeans_plusplus_initializer 関数によって得た各クラスタ中心の初期値と距離関数、単語ベクトルを xmeans 関数に与えることによって実行した。クラスタの中心を求める際にクラスタ数を与えているが、x-means 法を行った後のクラスタ数には変動があるため、クラスタ数をそれぞれ記録した。またクラスタ中心を与えなかった場合、ランダムにそれらが選ばれ、クラスタ数を設定することなく自動で導出することができる。これを行った時の結果も記録した。

4.5 次元削減の計算時間

4.5.1 計算時間の取得

三つの手法それぞれにおける次元削減行列が導出されるまでの計算時間を、time ライブラリの process_time 関数を用いて記録した。

4.5.2 グラフの作成

実験パターンごとに、各手法の計算時間の推移を折れ線グラフで作成した。横軸を累積寄与率、縦軸を計算時間とし、手法ごとの 3 つの折れ線グラフと、それぞれの計算時間の平均値を出力した。

4.6 x-means 法のクラスタ数

4.6.1 クラスタ数の取得

実験パターンごとに、x-means 法でクラスタリングを行った後、_xmeans__clusters 関数によってクラスタ数を取得した。

4.6.2 グラフの作成

実験パターンごとに、累積寄与率によって決定したクラスタ数と、x-means 法から得たクラスタ数の折れ線グラフを作成した。横軸を累積寄与率、縦軸をクラスタ数とした。

4.7　類似度による文書分類

4.7.1　クラスタリング

次元削減後の文書-クラスタ行列を用いて文書のクラスタリングを行い、文書分類を行った。文書間の距離は cos 距離とし、ward 法を用いて分類した。scipy ライブラリの pdist 関数に行列と距離関数を与え、それを linkage 関数に与えることで実行した。

4.7.2　デンドログラムの出力

linkage 関数と dendrogram 関数を用いて、実験パターン 1, 2, 3 のデンドログラムを出力した。パターン 1, 2 を比較することで各文書の情報量がデンドログラムにどのような影響を及ぼすのか考察する。ただし、実験パターン 5, 6 は文書数が多く、目視での確認が不可能であったため、出力しない。

4.8　正解率の計算

4.8.1　クラスタリング結果の取得

fcluster 関数にクラスタ数を与えることで、各文書のクラスタを得た。この時、クラスタ数は各実験パターンのジャンル数を設定した。

4.8.2　正解率の計算

実験パターンごとの各手法において、以下の手順で正解率を導出した。

1. 分割されたクラスタに、ランダムなジャンルを割り当てる。
2. accuracy_score 関数で正解率を求める。
3. 手順 1 とは別のジャンルを各クラスタに割り当て、手順 2 を行う。以後クラスタ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。
4. 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。

4.8.3　グラフの作成

実験パターンごとに、各手法の正解率の推移を折れ線グラフで作成した。横軸を累積寄与率、縦軸を計算時間とし、手法ごとの 3 つの折れ線グラフと、それぞれの正解率の平均値を出力した。

4.9　実験結果

4.9.1　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 を Figure 4.2、実験パターン 3 を Figure 4.3、実験パターン 4 を Figure 4.4、実験パターン 5 を Figure 4.5、実験パターン 6 を Figure 4.6 に示す。それぞれのグラフより、以下のことが分かった。

- 全てのパターンにおいて、正解率は累積寄与率とは関係なく上下している。また潜在意味解析に対し、クラスタリングによる次元削減は正解率の変動が大きい。
- パターン 1 では k-means 法の正解率の平均値が最も高く、パターン 2 では x-means 法が最も高い。潜在意味解析はパターン 1 よりパターン 2 の方が正解率の平均値が低くなっているが、x-means 法はパターン 2 の方が高くなっている。クラスタリング法の正解率の変動が、パターン 1 よりパターン 2 の方が大きい。
- パターン 3, 4 では、潜在意味解析よりクラスタリングによる次元削減の方が正解率の平均値が高い。またパターン 3 よりパターン 4 の方が全体的に正解率が高い。この 2 つのパターンは、他のパターンに比べて正解率の変動が特に大きいが、平均値は高い。
- パターン 5, 6 もまた、潜在意味解析よりクラスタリング法の方が正解率の平均値が高い。パターン 5, 6 の結果に大きな違いはないが、パターン 6 の方が少し正解率の変動が大きい。またこの 2 つのパターンは、他のパターンに比べて正解率の平均値が低い。

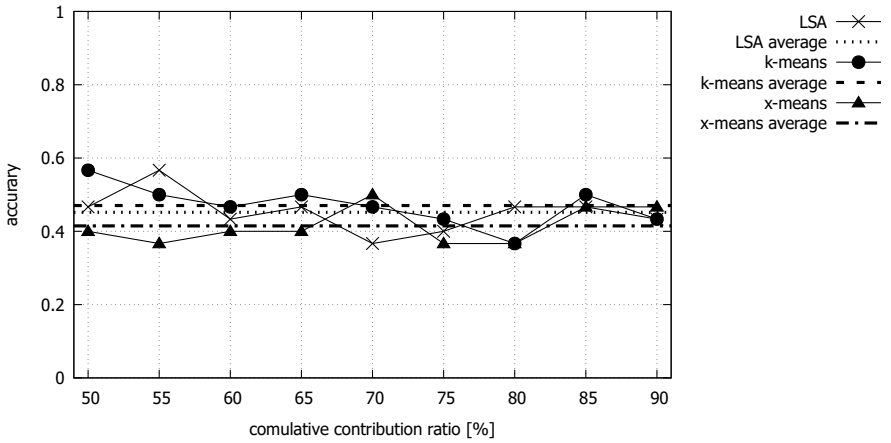

Figure 4.1 Accuracy of synopsis (Fairytale, Horror, Love).

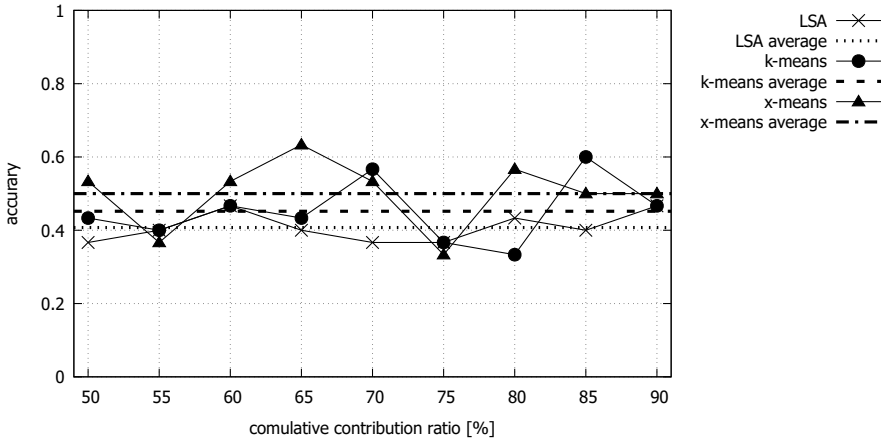

Figure 4.2 Accuracy of chapter one (Fairytale, Horror, Love).

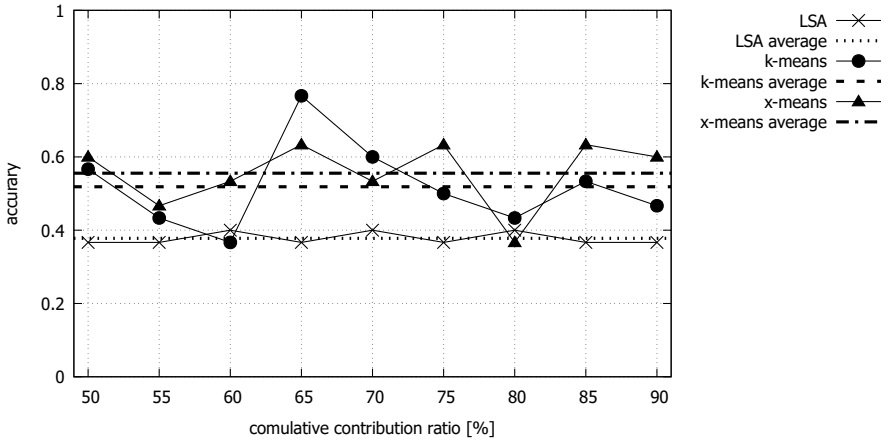

Figure 4.3 Accuracy of synopsis (Science Fantasy, Love, Mistery).

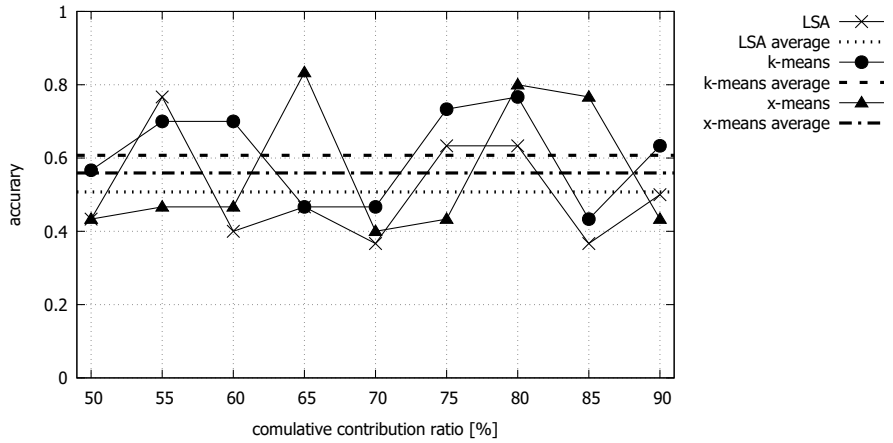

Figure 4.4 Accuracy of chapter one (Science Fantasy, Love, Mistery).

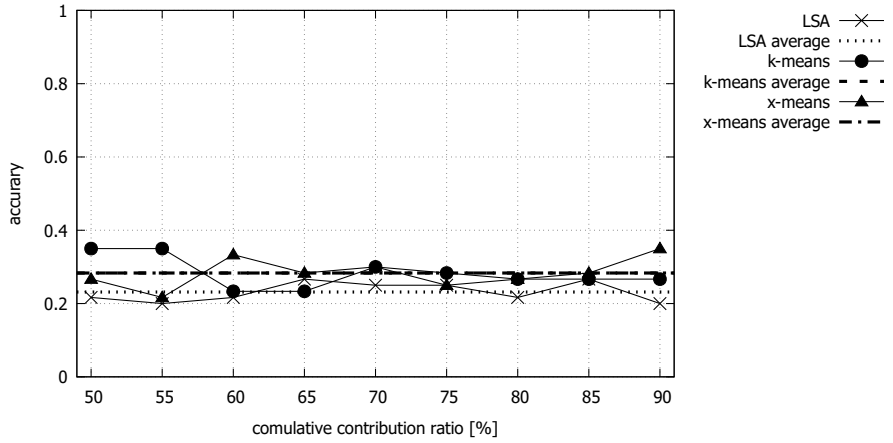

Figure 4.5 Accuracy of synopsis (6 genres).

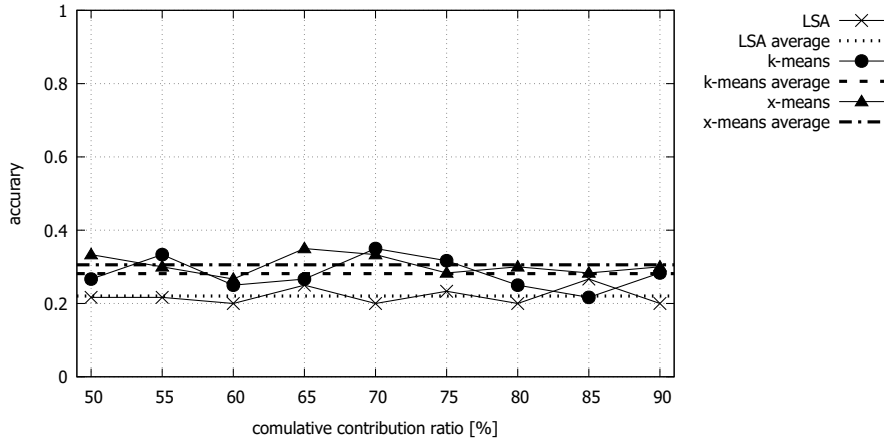

Figure 4.6 Accuracy of chapter one (6 genres).

4.9.2　文書分類のデンドログラム

正解率の妥当性を確認するため、実験パターン 1, 2, 3 の各手法のデンドログラムを出力した。正解率が高いデンドログラムと低いデンドログラムがどう異なるのかを確認するために、パターン 1, 2 では累積寄与率が 70%、パターン 3 では 65%の際のデンドログラムを比較する。パターン 1 のデンドログラムを Figure 4.7 から Figure 4.9 に、パターン 2 を Figure 4.10 から Figure 4.12 に、パターン 3 を Figure 4.13 から Figure 4.15 に示す。

パターン 1 では、潜在意味解析による文書間の距離が近くとも 0.2 ほどあるのに対し、k-means 法では、0.4 以下の距離で隣り合う文書のまとまりが 5 つほどできている。x-means 法は 0.6 付近で 5 つほどのまとまりができている。つまり、正解率の低かった潜在意味解析の文書間の距離は離れているのに対し、正解率の高かった k-means 法、x-means 法の文書間の距離は近いことが分かる。パターン 2, 3 でも同様のことが言える。また正解率が最も高かったパターン 3 の k-means 法のデンドログラムである Figure 4.11 は、文書間の距離が全体的に近く、正解率が低かったパターン 1, 2 の潜在意味解析のデンドログラムである Figure 4.7, Figure 4.10 は、文書同士が特に遠くに分布していることが確認できる。

パターン 1, 2 の潜在意味解析のデンドログラムを比較する。パターン 1 のデンドログラムでは、同じジャンルのまとまりはあまり発見できず、距離の近い文書同士は異なるジャンルであるものが多い。対してパターン 2 のデンドログラムでは、中央に恋愛のまとまりが発見できるが、それ以外には見受けられない。

パターン 1, 2 の k-means 法によるデンドログラムを比較する。パターン 1 のデンドログラムでは、上部に恋愛の文書が多く、中央から下はホラーが多い。しかし、童話はあまりまとまっておらず、複数のクラスタにまたがっていることが分かる。対してパターン 2 のデンドログラムは、上部に童話、中央に恋愛、下部にホラーとおおむね意図したとおりにクラスタが作られていることが確認できる。

パターン 1, 2 の x-means 法によるデンドログラムを比較する。パターン 1 のデンドログラムでは、上部に恋愛、下部にホラーが固まっている。しかし、童話には目立ったまとまりは見られない。対してパターン 2 のデンドログラムは、上部に恋愛、中央に童話、下部にホラーと意図したとおりの分類が行われていることが確認できる。

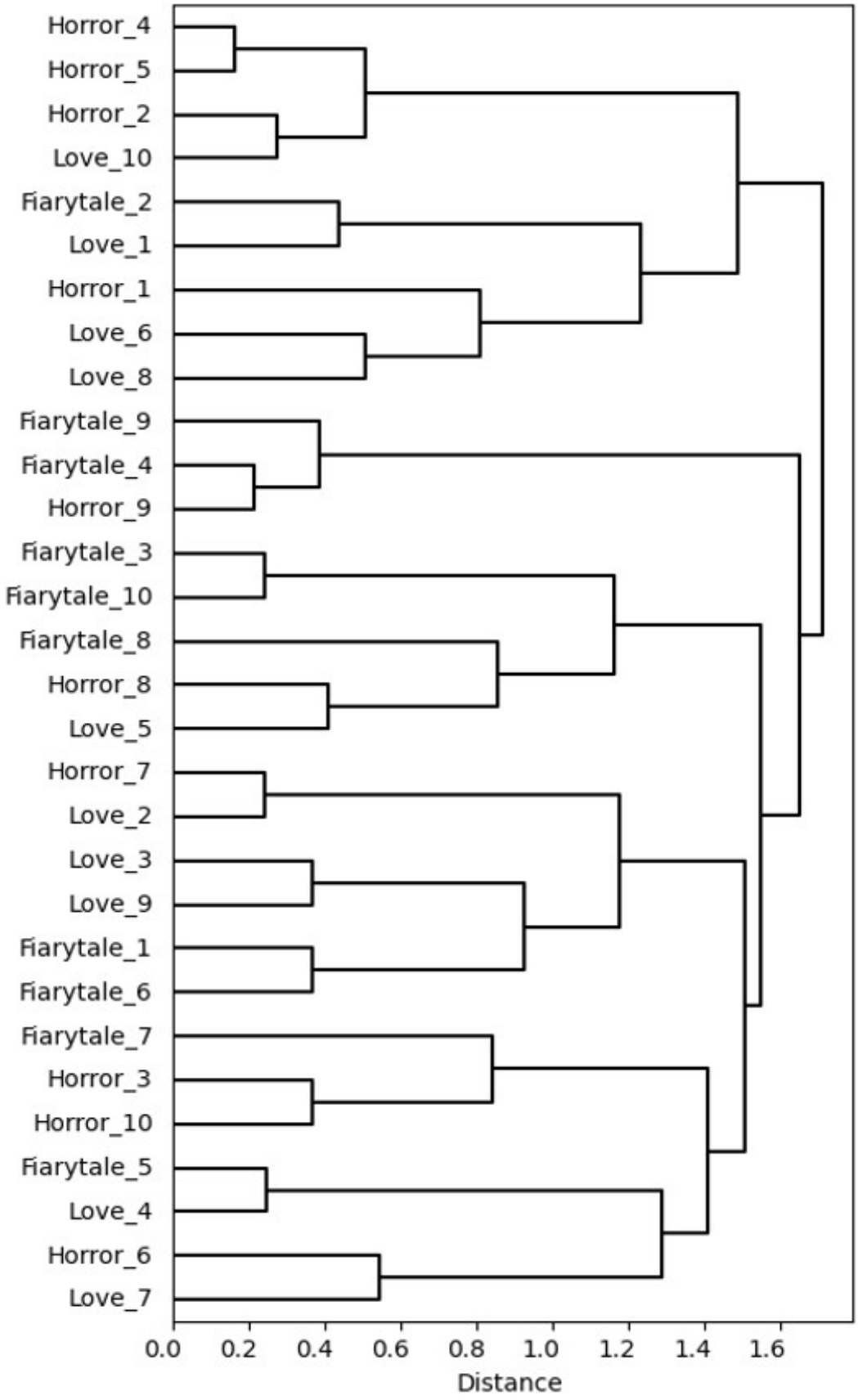

Figure 4.7 Result of dimension reduction by LSA (Pattern1 70%).

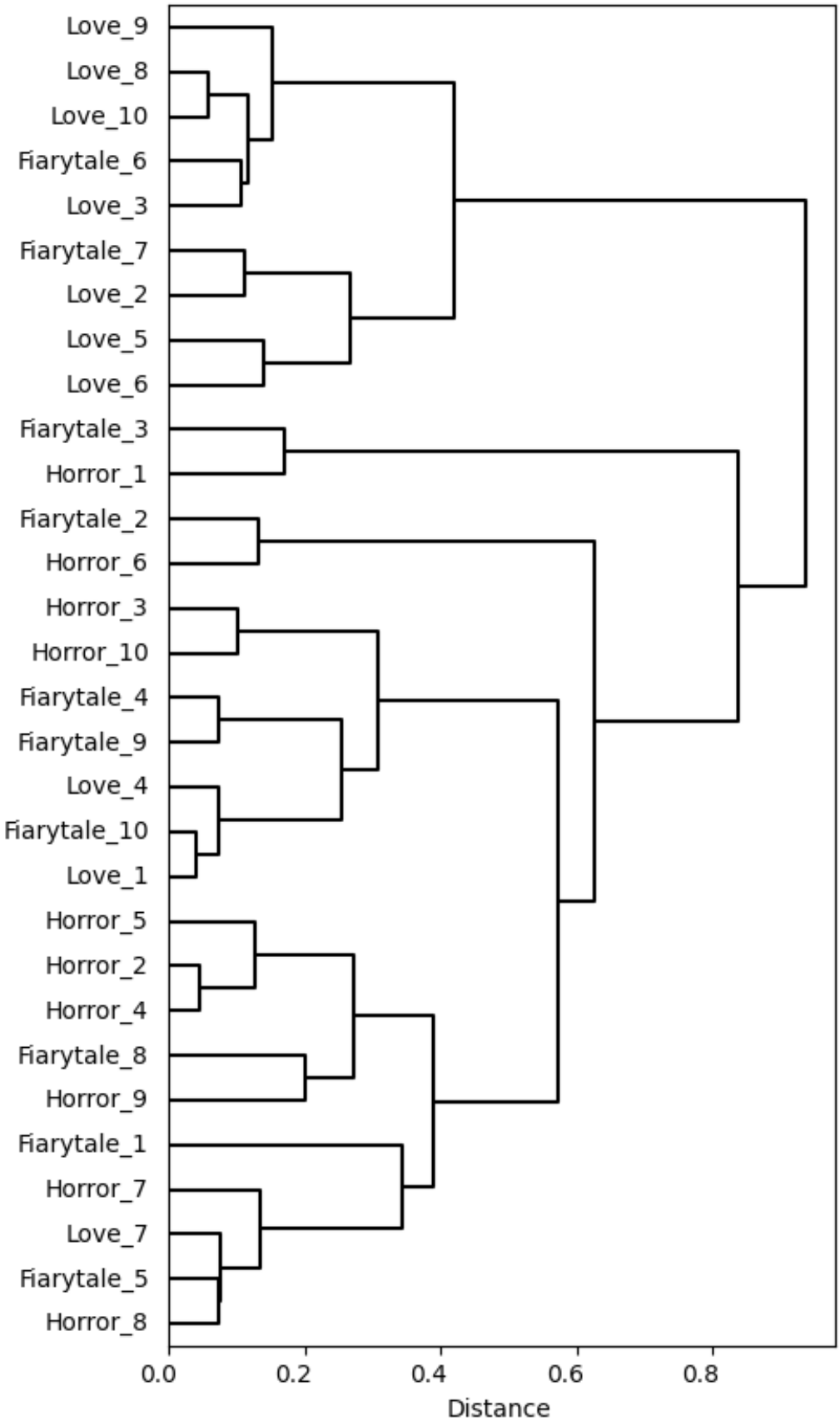

Figure 4.8 Result of dimension reduction by k-means (Pattern1 70%).

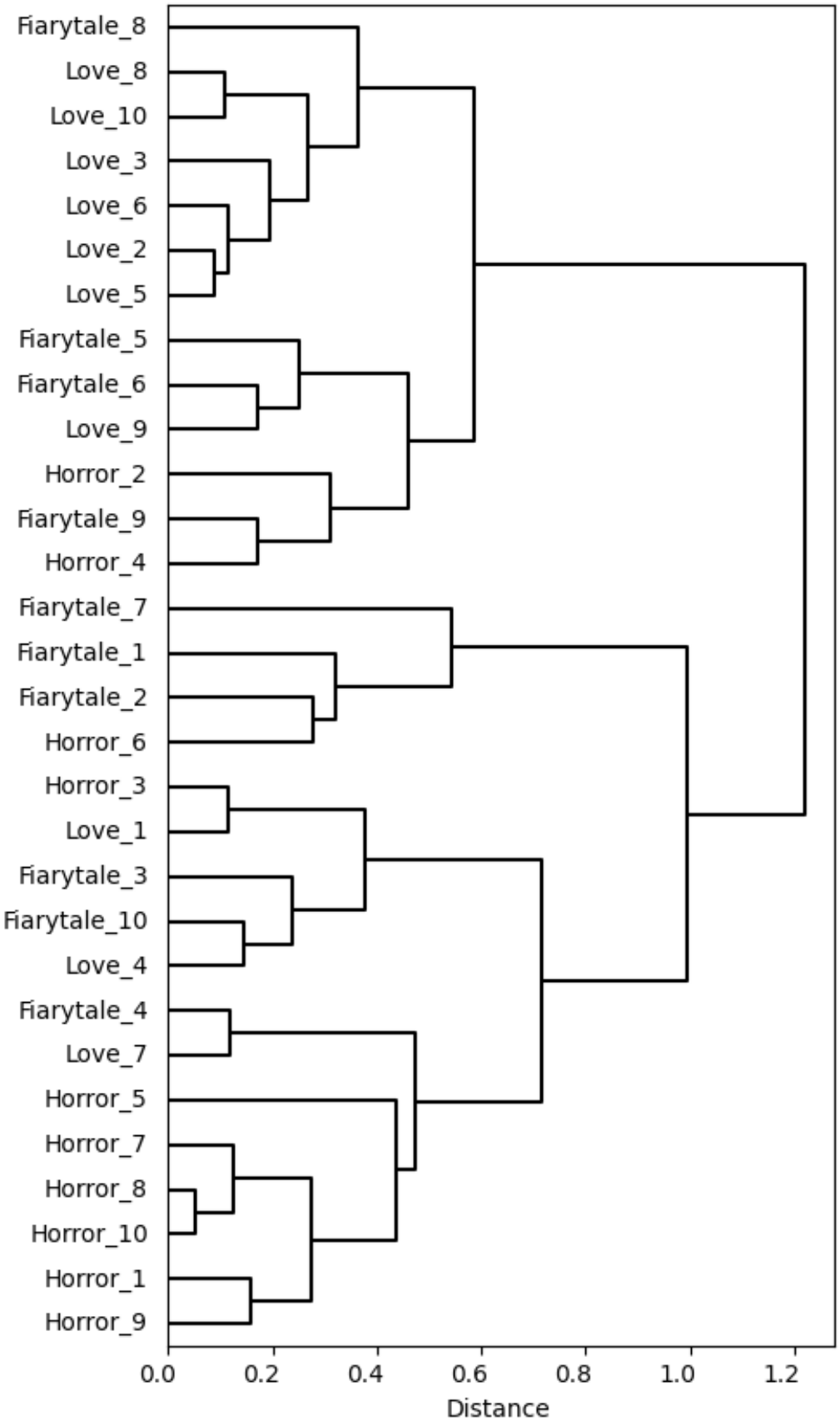

Figure 4.9 Result of dimension reduction by x-means (Pattern1 70%).

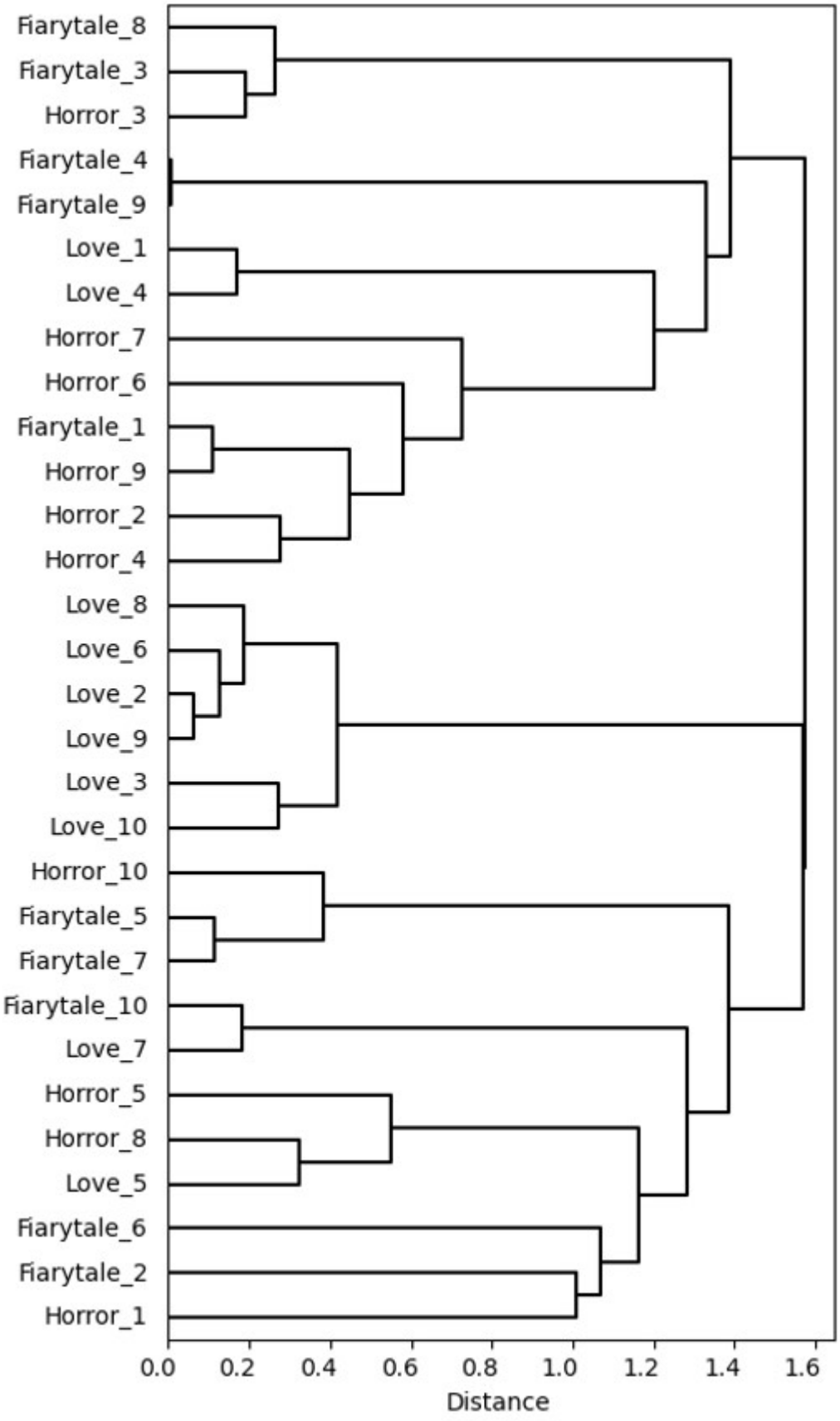

Figure 4.10 Result of dimension reduction by LSA (Pattern2 70%).

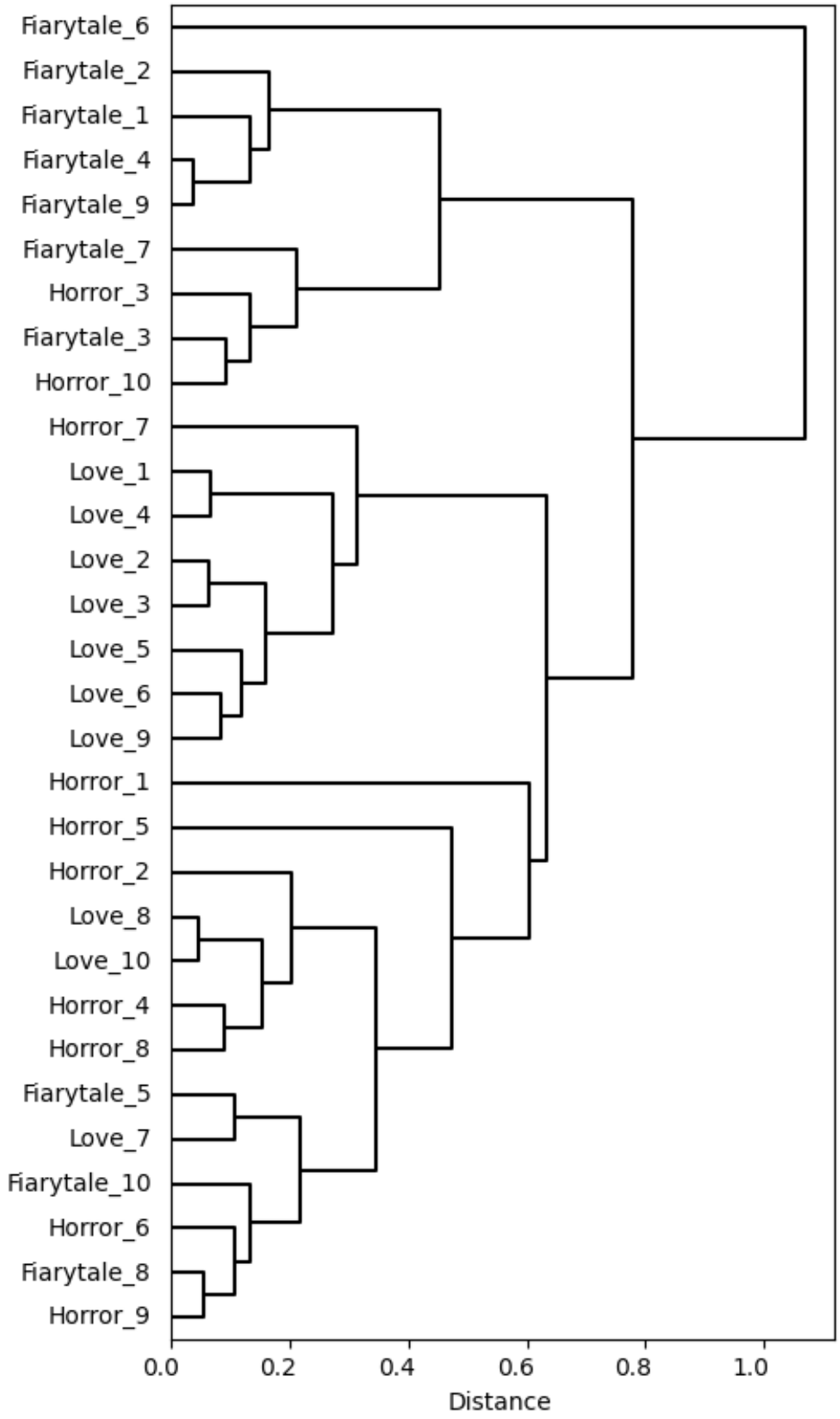

Figure 4.11 Result of dimension reduction by k-means (Pattern2 70%).

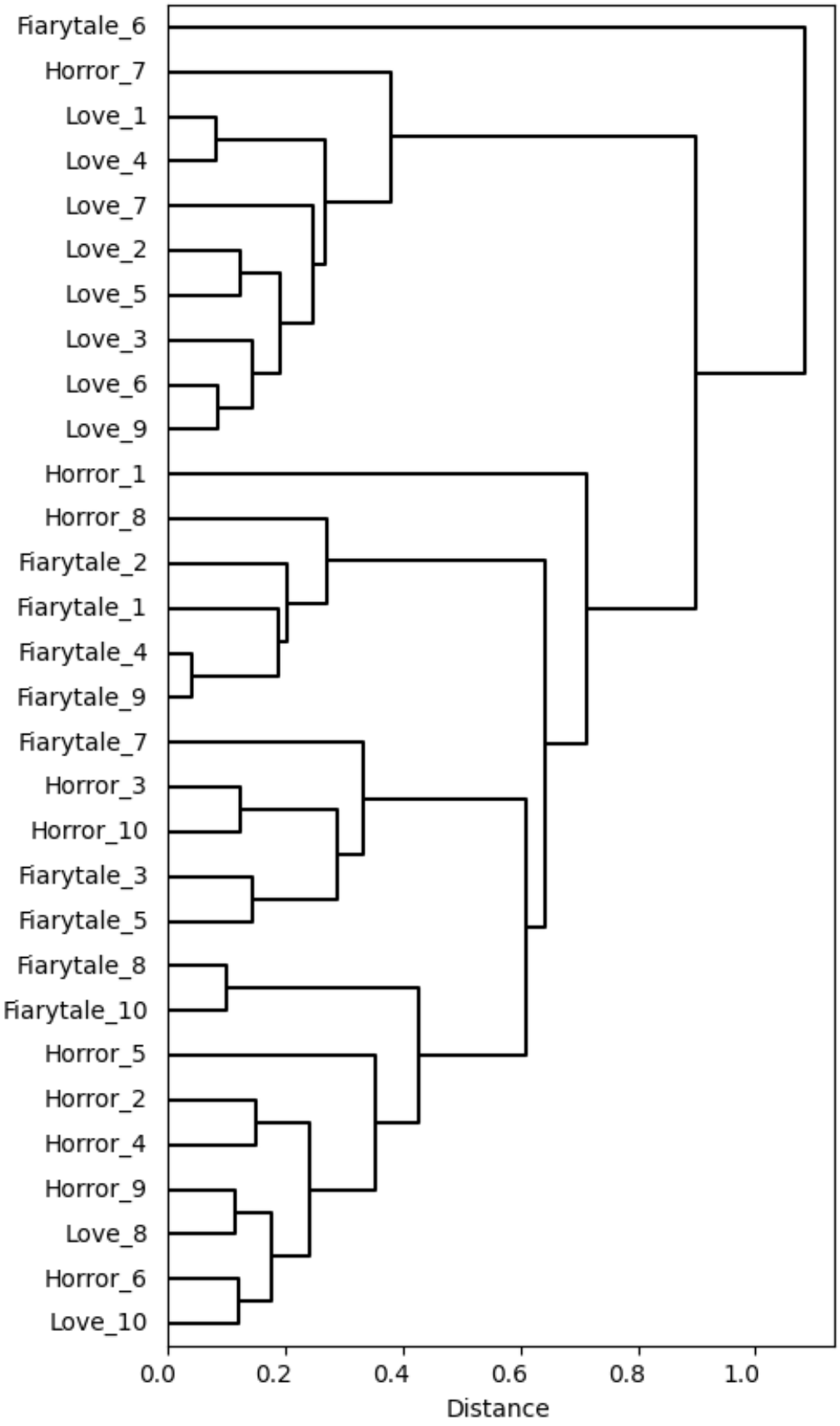

Figure 4.12　Result of dimension reduction by x-means (Pattern2 70%).

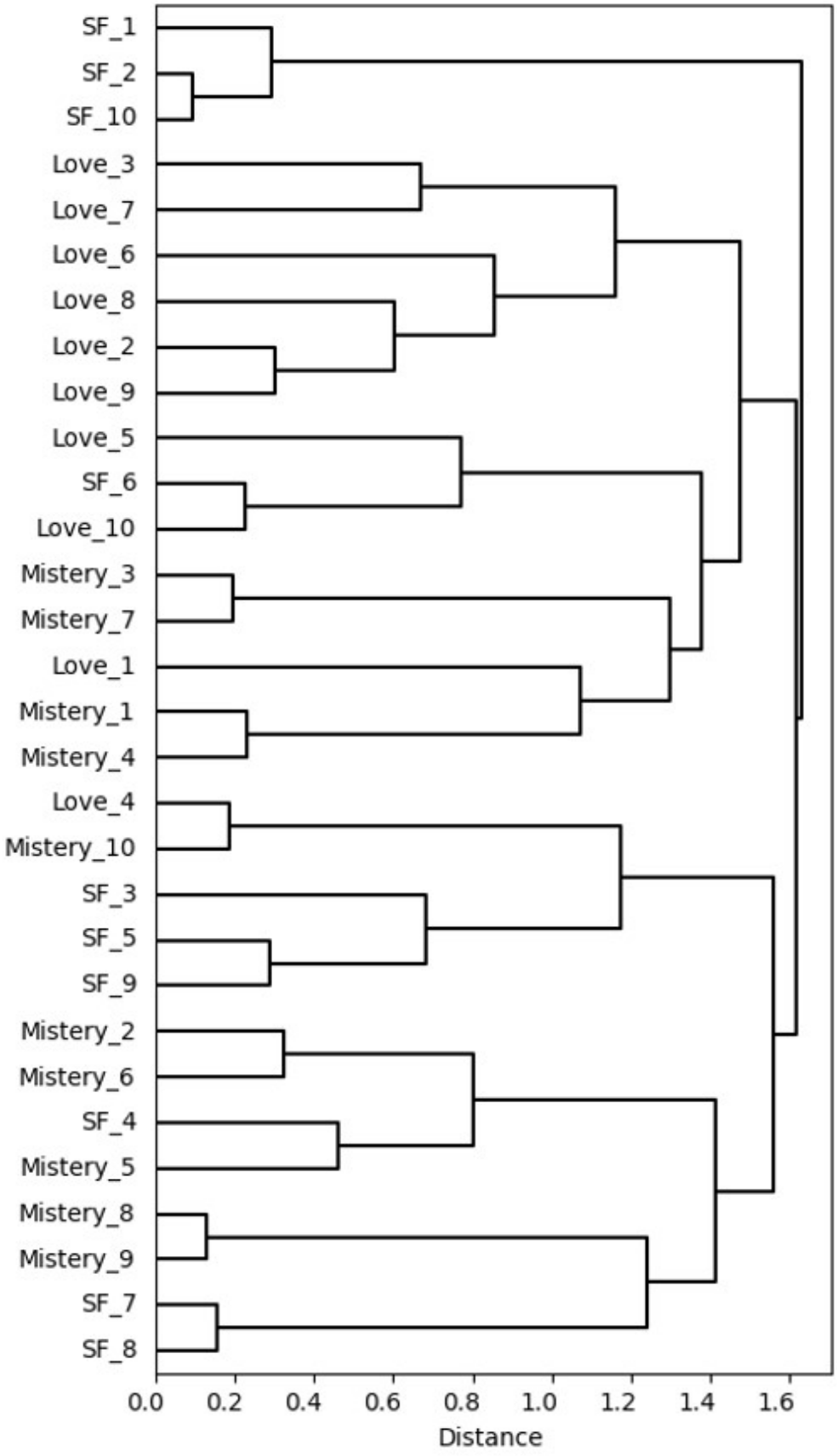

Figure 4.13 Result of dimension reduction by LSA (Pattern3 65%).

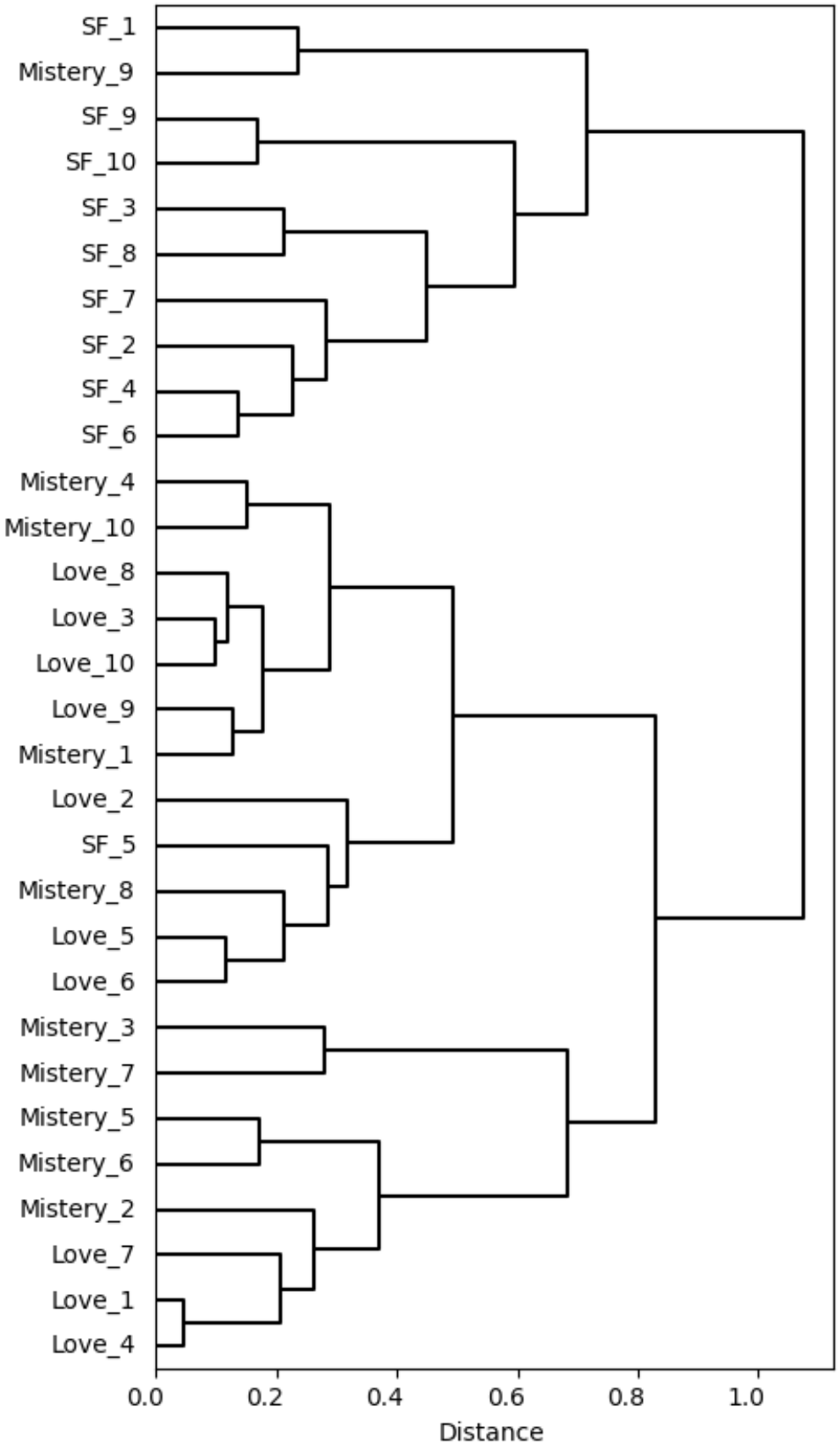

Figure 4.14 Result of dimension reduction by k-means (Pattern3 65%).

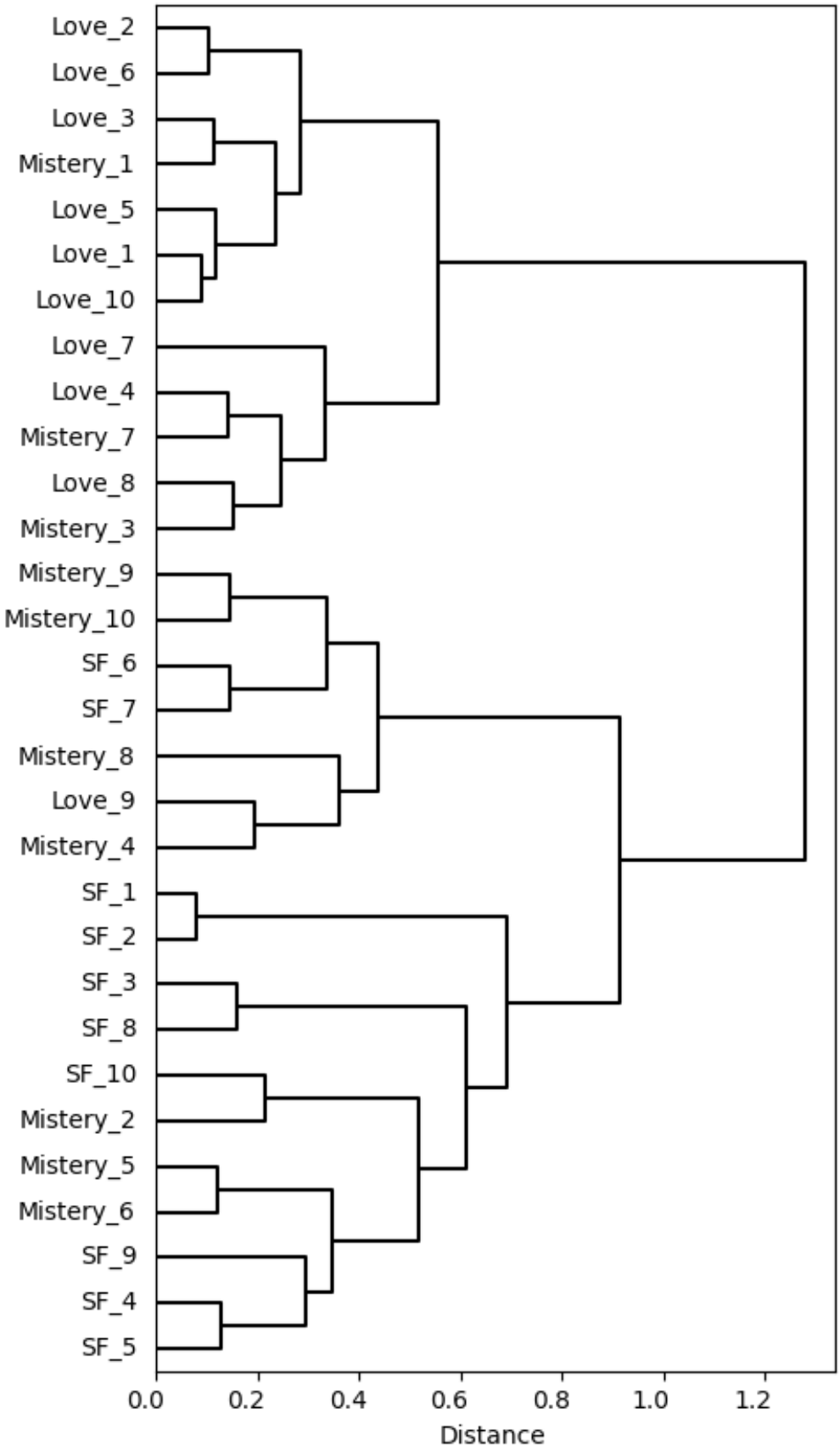

Figure 4.15 Result of dimension reduction by x-means (Pattern3 65%).

4.9.3 計算時間

各手法による、次元削減行列を作成するまでの計算時間を、折れ線グラフとして出力した。実験パターン 1 の計算時間を Figure 4.16、実験パターン 2 を 4.17、実験パターン 3 を Figure 4.18、実験パターン 4 を Figure 4.19、実験パターン 5 を Figure 4.20、実験パターン 6 を Figure 4.21 に示す。

全ての実験パターンにおいて、潜在意味解析は極めて少ない計算時間で実行が完了し、クラスタリングによる次元削減は双方ほぼ同じだけの時間がかかっている。パターン 2 の k-means 法は、累積寄与率が高くなるとともに計算時間が減っていく負の相関関係がみられるが、パターン 3, 5 の k-means 法、x-means 法には正の相関関係が確認できる。また全ての実験パターンにおいて、k-means 法と x-means 法は同じような推移で計算時間が変化している。

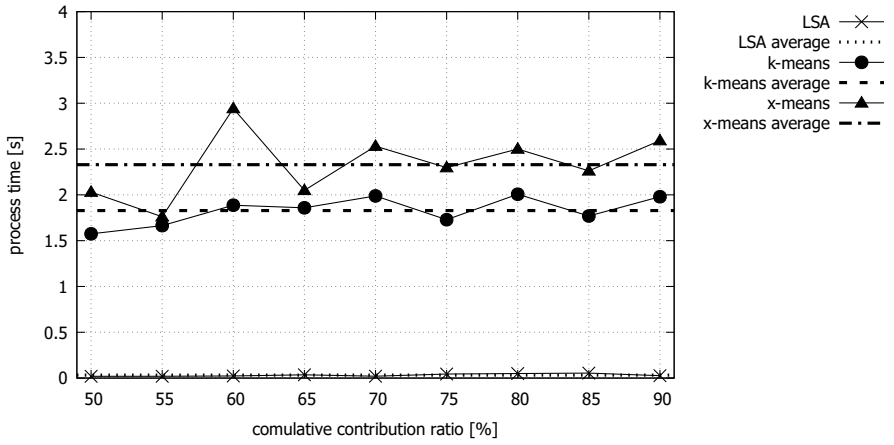

Figure 4.16　Process time of synopsis (Fairytale, Horror, Love).

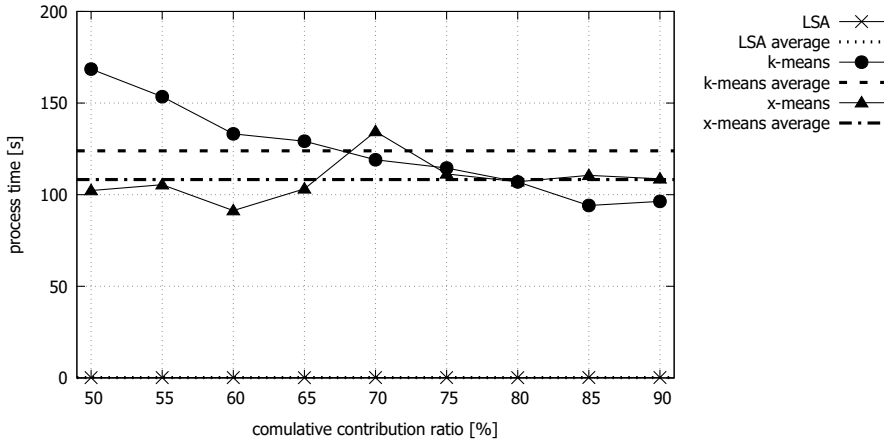

Figure 4.17 Process time of chapter one (Fairytale, Horror, Love).

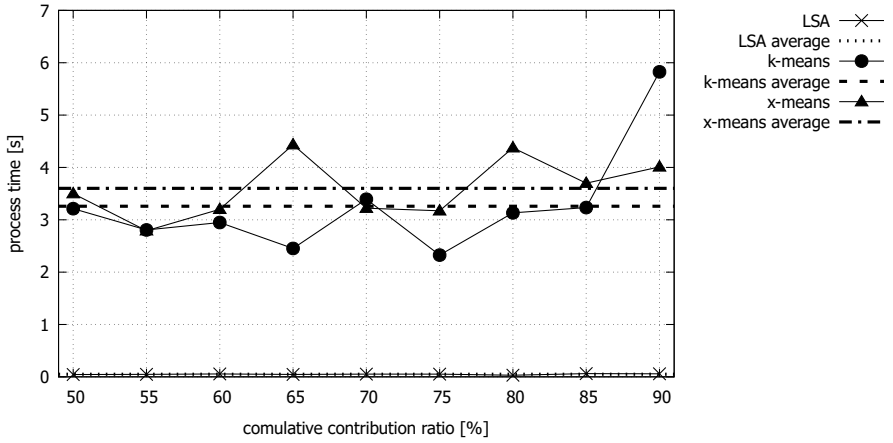

Figure 4.18 Process time of synopsis (Science Fantasy, Love, Mistery).

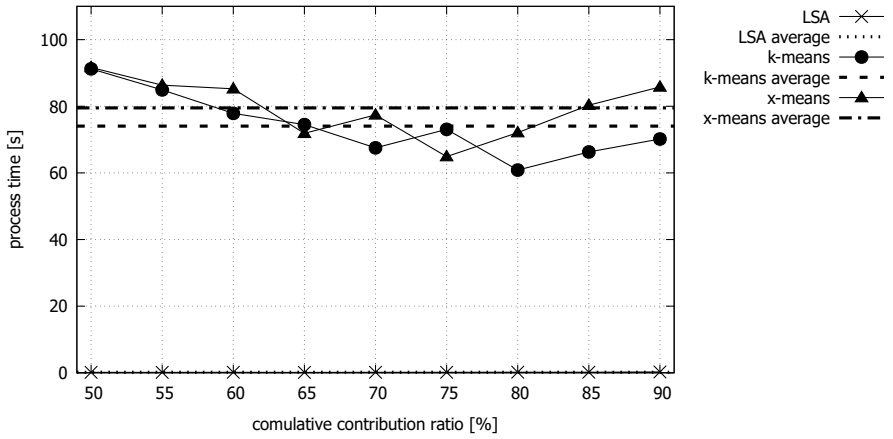

Figure 4.19 Process time of chapter one (Science Fantasy, Love, Mistery).

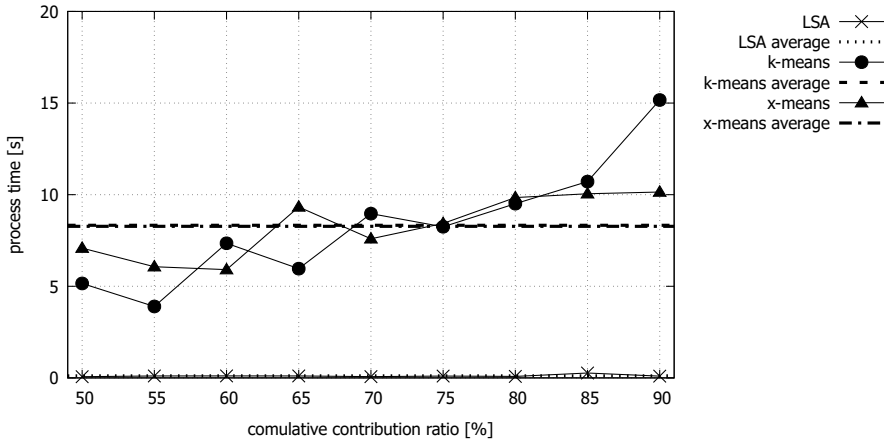

Figure 4.20　Process time of synopsis (6 genres).

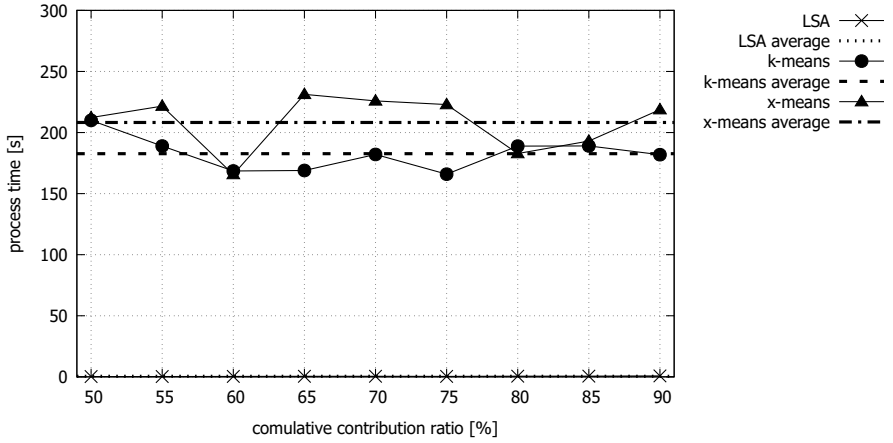

Figure 4.21　Process time of chapter one (6 genres).

4.9.4　x-means 法のクラスタ数

累積寄与率によって決定したクラスタ数と、そのクラスタ数を初期値として実行した x-means 法の最終的なクラスタ数を比較する。実験パターン 1 のクラスタ数を Figure 4.22、実験パターン 2 を 4.23、実験パターン 3 を Figure 4.24、実験パターン 4 を Figure 4.25、実験パターン 5 を Figure 4.26、実験パターン 6 を Figure 4.27 に示す。

パターン 1, 3, 5 では、x-means 法のクラスタ数はほとんど変化していない。それに対し、パターン 2, 4, 6 ではクラスタ数は大きく異なり、x-means 法のクラスタ数は初期値よりも大きくなっている。またそれらの値は累積寄与率には関係なく、常に近い値を示している。さらにその値は累積寄与率が 90%のときに決定したクラスタ数と似た値である。

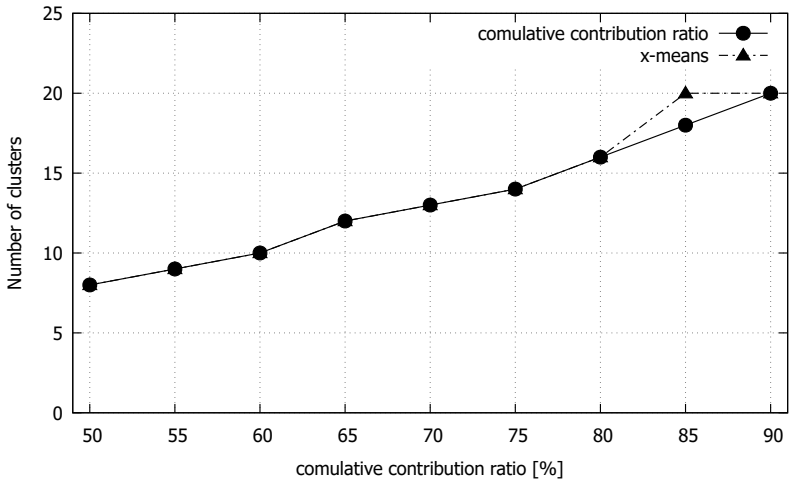

Figure 4.22 Number of clusters of synopsis (Fairytale, Horror, Love).

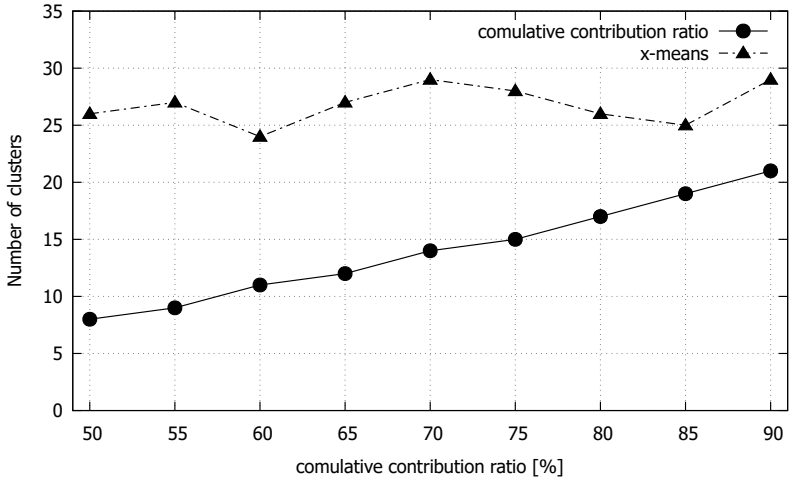

Figure 4.23 Number of clusters of chapter one (Fairytale, Horror, Love).

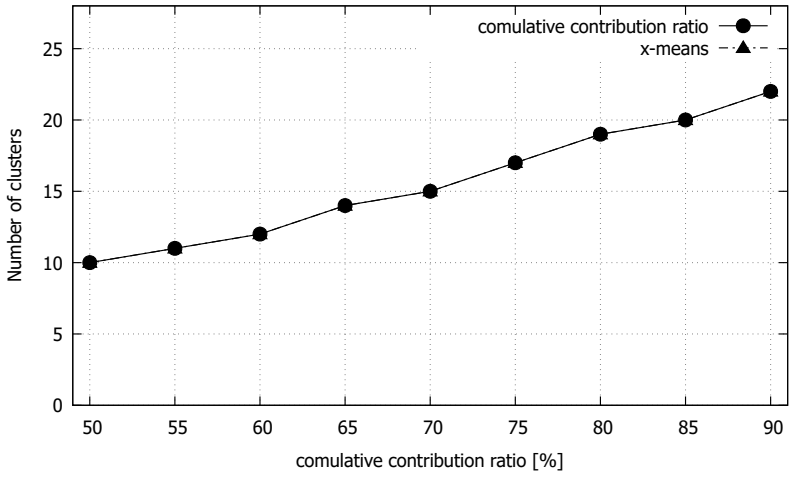

Figure 4.24 Number of clusters of synopsis (Science Fantasy, Love, Mistery).

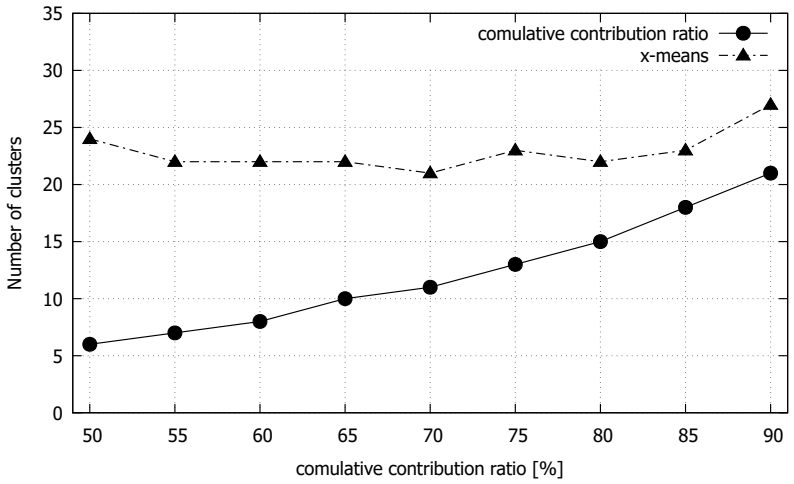

Figure 4.25 Number of clusters of chapter one (Science Fantasy, Love, Mistery).

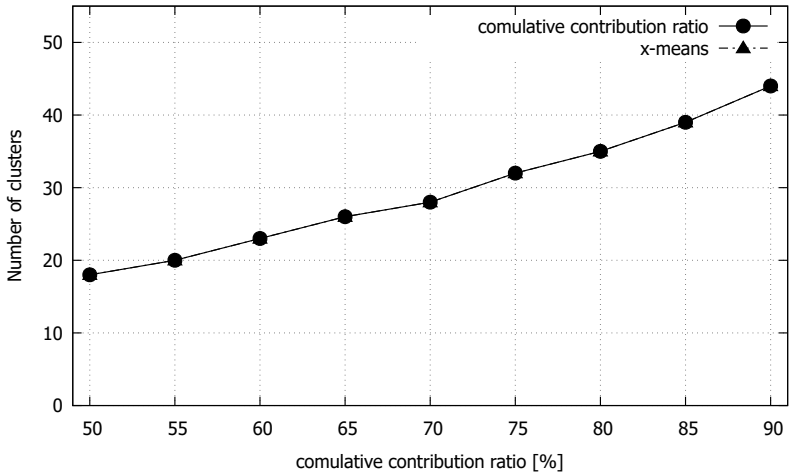

Figure 4.26 Number of clusters of synopsis (6 genres).

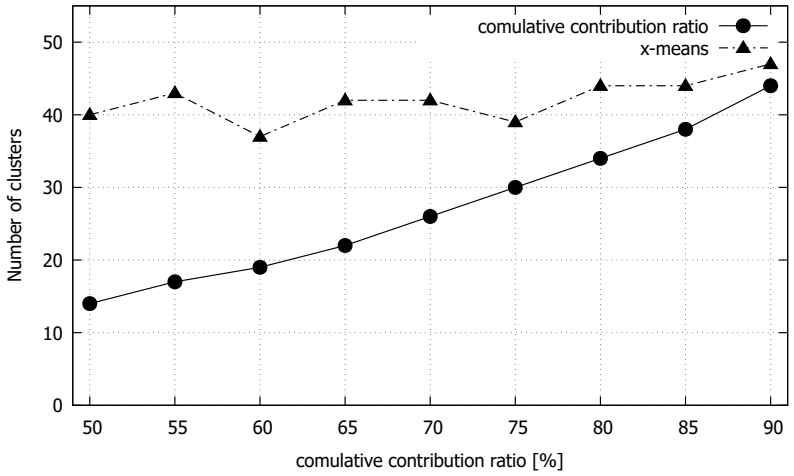

Figure 4.27 Number of clusters of chapter one (6 genres).

4.9.5　x-means 法による全自動文書分類

x-means 法に初期値を与えずに実行した結果を Table 4.3 に示す。なお、値は小数点第 4 位を四捨五入している。

初期値を与えた場合の結果と比べて、パターン 1 以外は早く計算が終了していることが確認できる。パターン 2 は特に、初期値を与えた x-means 法や k-means 法の計算時間の約 1/2 となっている。パターン 1, 3, 5 の小説のあらすじに対する分類を行った際、クラスタ数が 3 つと少ない数になったが、正解率は特別高くも低くもなく、初期値を与えた際の結果と似た値を得た。またパターン 2, 4, 6 のクラスタ数は、初期値を与えた場合の x-means 法の最終的なクラスタ数に近い値を示している。

Table 4.3　Fully automatic dimension reduction by x-means.

	Pattern1	Pattern2	Pattern3	Pattern4	Pattern5	Pattern6
Process time [s]	2.798	64.884	4.972	49.953	6.603	140.361
Number of clusters	3	28	3	25	3	40
Accuracy	0.467	0.467	0.6	0.4	0.3	0.35

4.10　考察

4.10.1　正解率

実験結果より、全ての実験パターンにおいて正解率と累積寄与率の関係がないことが明らかになった。またジャンル数が増えると、正解率は大幅に下がった。ジャンル数が少ない場合、1 つのジャンルのクラスタができていれば消去法のような形で他のジャンルのクラスタも生成できるが、ジャンル数が多ければそれは叶わないことが予想されるため、このような結果になったと考える。

4.10.2　各手法の比較

全体を通して、潜在意味解析よりクラスタリングによる手法の方が正解率が高くなった。潜在意味解析は特異値分解により行列を分解し、特異値の小さいものを削ることによって削減しているにすぎず、単語の意味を考慮したものにはなっていない。対して他 2 つの手法は、Word2vec による単語ベクトルを利用して単語をクラスタリングし、次元削減に利用しているため、このような結果になったといえる。しかし、パターン 1 では

x-means 法の正解率の平均値が最も低くなった。このとき、Figure 4.22 より、k-means 法と x-means 法のクラスタ数はほとんど変わっていない。よってこの正解率の違いは、初期値依存性によるものであると考えられる。

また k-means 法、x-means 法の 2 つの手法は、一つの文書に含まれる情報量が多いパターンの方が、正解率の平均値が高かった。しかし全体的に正解率が上がったのではなく、特に高くなる場合が生まれることによって、平均値が高くなった。異なるジャンル同士で距離が近くなった Horror_6 と Love_10 の文書を見比べてみると、前者には「消防車」「嫌悪」「泥棒」、後者には「警察」「醜悪」「暴力」といった似た単語が多く用いられていた。ジャンルは異なるが、どちらもダークな雰囲気を持つ作品で、距離が近くなったことに納得のいく内容であった。よって、この 2 つの手法は作品の情報量が多いほど、類似度計算を有利に行うことができると考える。

k-means 法と x-means 法の正解率の差はあまり大きくは見られなかった。両者の差は主にクラスタ数にあるが、累積寄与率によってクラスタ数を変化させても正解率にあまり影響がなかったように、クラスタ数と正解率に明白な関係性は無いように考える。

4.10.3 計算時間

x-means 法と k-means 法の計算時間に大きな差はなかった。この計算時間というのは k-means++法を適用してから次元削減後の行列が作成されるまでであったため、k-means 法、x-means 法のみの計算時間を改めて測定した。結果、k-means 法より x-means 法の方がどの実験パターンでも計算時間が長かった。x-means 法は k-means 法の 2 倍以上の計算量が必要になるため、このような結果になったと考えられる。

x-means 法に初期値を与えなかった場合、正解率は初期値を与えた場合と同等の値を示し、さらに計算時間は短縮されていた。単語数が少ないとき、クラスタ数はどのパターンでも 3 つとなっていたため、計算時間が短縮されたことは明白である。一方単語数が多いとき、クラスタ数は初期値を与えた場合のクラスタ数と似た値を取っていたにもかかわらず、計算時間は短くなっていた。k-means++法の計算を行う時間を調べたところ、平均して 0.5 秒ほどであったため、k-means++法を省いた影響ではない。この理由を説明するために、最適なクラスタ数を 8 つと仮定して、それぞれのアルゴリズムを考える。x-means 法に初期値として 5 つのクラスタ中心を与えた場合、5-means 法を行った後、5 つのクラスタそれぞれで 2-means 法を適用する。この時点で k-means 法を 6 回行ってい

る。対して初期値を与えなかった場合は、ランダムに選定した2つのクラスタ中心をもとに2-means法を行った後、それぞれのクラスタで2-means法を行う。これに再度2-means法を適用すれば、クラスタ数は8つとなる。ここでk-means法の回数は5回である。このように、初期値を与えないx-means法の方が計算量が少なくなる場合があるため、計算時間が短くなったのではないかと考えられる。

4.10.4 x-means法によって得られたクラスタ数

少ない単語群に対し、初期値を与えたx-means法によってクラスタリングした際、クラスタ数は初期値からほとんど増加しなかった。これは、初期値を与えなかったx-means法においてクラスタ数がとても少なかったことと関係していると考えられる。また多くの単語群に対し、初期値を与えたx-means法によってクラスタリングした際は、クラスタ数はどの累積寄与率でもほぼ一定の値をとった。この値は初期値を与えていないx-means法の結果と似た値であった。これによって、x-means法のアルゴリズムによって導出されるクラスタ数は安定することを確認できた。

4.10.5 正解率とデンドログラムの関係

デンドログラムにより、文書間の距離が近いほど正解率が高くなるという結果が得られた。全体的に文書間の距離が遠い場合、ある程度離れている文書であっても同じクラスタに分類されることがあるため、このような結果になったと考えられる。またデンドログラムによってクラスタの中身を見ることによって、正解率が高いとき、同じジャンルの文書は同じクラスタに分類されていることを確認できた。

第5章　結論

本研究では、Word2vec と x-means 法による次元削減法を提案し、潜在意味解析や k-means 法との比較によってその有用性を検証した。

実験の手順を説明する。まず WebAPI を用いて、小説のタイトル、あらすじ、そして本文の第一章を取得した。これに対して形態素解析、Word2vec による単語ベクトルの取得、Tf-Idf の導出を行った。その後、潜在意味解析、k-means 法、x-means 法を用いて次元削減を行い、得られた文書行列を利用して階層的クラスタリングによって文書を分類した。この結果を正解率、デンドログラム、次元削減行列作成までの計算時間、そしてクラスタ数という形で出力した。以上の作業を、ジャンル数などの条件を変更した 6 つのパターンで行い、結果を比較した。

正解率の妥当性をデンドログラムにて確認したうえで正解率を比較すると、k-means 法、x-means 法による次元削減法は潜在意味解析より有効性が高いという結果が得られた。これら 2 つの手法は単語の意味を考慮したうえでクラスタリングを行い、次元削減に利用しているため、小説のあらすじや本文の分類には適した手法であったといえる。k-means 法、x-means 法の正解率の差はあまりなく、両者とも文書あたりの情報量が多いほど正解率が高くなるという結果を得た。また k-means++法を用いた x-means 法と、クラスタ数を決定する必要のない純粋な x-means 法を比較すると、両者の正解率に大きな差は見られなかった。単語数が多いパターンではクラスタ数も近く、さらに計算時間は x-means 法のみである方が短かった。このように x-means 法による次元削減は、手軽に計算コストを抑えながらも、k-means 法と同程度の正解率を得られるため、k-means 法と同等、あるいはそれ以上の有用性をもつ手法であるといえる。

実験を行う上で、2 つの懸念点が浮かび上がった。1 つは、初期値依存性による結果の変動である。k-means 法と x-means 法の結果を比較するにおいて、初期値依存問題は大きく評価指標を狂わせたといえる。k-means++法を利用することで多少は問題を克服できたが、それでもなお計算を行うたびに結果が変化するため、信頼性に疑問の残る結果となった。初期値依存問題を解決するために考案された手法は、k-means++法以外にもいくつか存在する。それらの手法を用いることで、より良い評価を行うことができるかもしれない。2 つ目は、評価指標として正解率を用いることへの不安である。実験パターンによっては、累積寄与率によって正解率が大きく変動した。この理由を突き止めるた

め、同じクラスタに分類された異なるジャンルの文書の中身を確認すると、主観ではあるが同じクラスタに分類されたことに納得のいく内容であった。しかし正解率を評価指標としている以上、これは間違った分類であるといわざるを得ない。よって文書間類似度の評価として、文書のジャンル分類を用いることは最善であるとはいえない。

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) MeCab: Yet Another Part-of-Speech and Morphological Analyzer. https://taku910.github.io/mecab/ (2023 年 10 月 4 日アクセス).
2) 水野貴明, Web API: TheGood Parts, オライリージャパン, 2014.
3) 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/ (2023 年 10 月 15 日アクセス).
4) 株式会社ヒナプロジェクト, 【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〜, MIERUCA AI, 2019. https://mieruca-ai.com/ai/lsa-lsi-svd/ (2023 年 1 月 23 日アクセス).
5) PyClustering, 2019. https://pyclustering.github.io/docs/0.9.0/html/index.html (2023 年 11 月 20 日アクセス).
6) 石岡恒憲, クラスタ数を自動決定する k-means アルゴリズムの拡張について, 大学入試センター 研究開発部, 2000. http://www.rd.dnc.ac.jp/%7Etunenori/doc/xmeans_euc.pdf (2023 年 10 月 3 日アクセス).
7) 山内長承, Python によるテキストマイニング入門, オーム社, 2017
8) 西川司優, Word2Vec を用いたモデルの違いによる文書分類の差, 岐阜工業高等専門学校電気情報工学科卒業研究報告, 2023.
9) 鈴木正敏, 日本語 Wikipedia エンティティベクトル, 東北大学, 2018. https://www.cl.ecei.tohoku.ac.jp/ m-suzuki/jawiki_vector/ (2023 年 10 月 4 日アクセス).
10) 武藤昇吾, 単語間の距離を用いたクラスタ分析による文書の類似度計算, 岐阜工業高等専門学校電気情報工学科卒業研究報告, 2023.
11) scikit-learn: machine learning in Python. https://scikit-learn.org/stable/ (2023 年 10 月 28 日アクセス).