ARG WI2 No.11, 2017
WI2-2017-18

多目的遺伝的アルゴリズムを用いた言語概念ベクトルの類似度計算による小説要約システムの性能評価

今野 勇気　　荒木 健治

北海道大学大学院情報科学研究科

y-konno@eis.hokudai.ac.jp　　*araki@ist.hokudai.ac.jp*

概要　現代では数多くの小説が存在し，また毎年多くの小説が発表されている．膨大な小説の中から人々が好みの小説を探す手段の一つとなる，小説の自動要約システムを提案する．文書の要約は文抽出による手法が主流であり，本提案システムでも小説中の重要文抽出によって小説の要約を試みている．また重要文の抽出を文の組み合わせ最適化問題とみなし要約を行う．本提案システムは，言語概念ベクトルを用いた意味的な類似度計算などの複数の関数を使用し，多目的最適化を行うものである．言語概念ベクトル化には word2vec と fasttext を用い，多目的最適化を多目的遺伝的アルゴリズムの一種である Non-dominated Sorting Genetic Algorithm-Ⅱ(NSGA-Ⅱ)を用いてシステムを構築した．本稿では，小説要約システムの構築及び，ROUGE を用いた要約システムの性能評価を行った結果について述べる．また小説の要約を生成する際に問題となるネタバレを回避するための予備実験の結果についても述べる．

キーワード　文書要約，多目的遺伝的アルゴリズム，言語概念ベクトル

1 はじめに

電子書籍やインターネット上の小説投稿サイト等の登場により，最近ではより小説に触れる機会が増えている．また特に小説投稿サイトによって，誰でも簡単に小説を発表できることで世の中に小説が溢れている．それらの膨大な小説の中から好みの小説を探すという行為は多くの時間や手間を要する．あらすじと呼ばれる小説の内容を短くまとめたものが存在するが，これを読めば簡単に短い時間でその小説がどんな内容であるかを把握することができる．しかし，全ての小説にあらすじが存在するわけではない．そこで本稿では，小説のあらすじを生成できるシステムの構築を目指す．

あらすじとは小説中の重要な部分の集合であるので，あらすじを生成するためには小説の重要な部分がどこであるかを判断する必要がある．これを文の組み合わせ最適化問題として重要文抽出を行うことにより実現する．これを厳密に解くという解法を取ると，小説のように多くの文を含んでいる対象では，文の組み合わせが膨大になり，最適化を行うための処理時間がかかりすぎるという問題がある．また良い要約には様々な要因が絡んでいると考え組み合わせ最適化に多目的遺伝的アルゴリズムの一種である Non-dominated Sorting Genetic Algorithm-Ⅱ (NSGA-Ⅱ)[1]を用いた．目的関数には word2vec[2]や fasttext[3]といった単語をベクトルに変換するツールを使用し，文同士の類似度計算を行う関数や，要約文の長さに関する関数を設定した．

2 関連研究

多目的遺伝的アルゴリズムを用い，重要文を抜き出すことで要約を行う手法には小倉らの研究[4]がある．小倉らの研究では，多目的遺伝的アルゴリズムの目的関数に単語の出現頻度，文の位置，JS ダイバージェンス，要約文の文字数等の情報を使用した関数を設定し要約を行っている．

言語概念ベクトルを用いて算出した文書間類似度を使用して要約を行う手法として住田らの研究[5]がある．住田らの研究では，要約前の文書と要約文の言語概念ベクトルを作成し，ベクトル同士の類似度が最も高い要約文を探索する手順を整数線形計画問題として解いている．

本研究では，小倉らの研究を参考にしつつ，言語概念ベクトルを使用することで表層的だけではない意味的な類似度の高い要約文生成を目指す．

3 NSGA-Ⅱ

本研究では，小説の要約というタスクを文の組み合わせ最適化問題とみなし，重要文抽出を行うことで要約を行う．また，遺伝的アルゴリズムの一種である NSGA-Ⅱを用いることで文の組み合わせ最適化問題を解く．NGSA-Ⅱの概要図は図1の通りである．

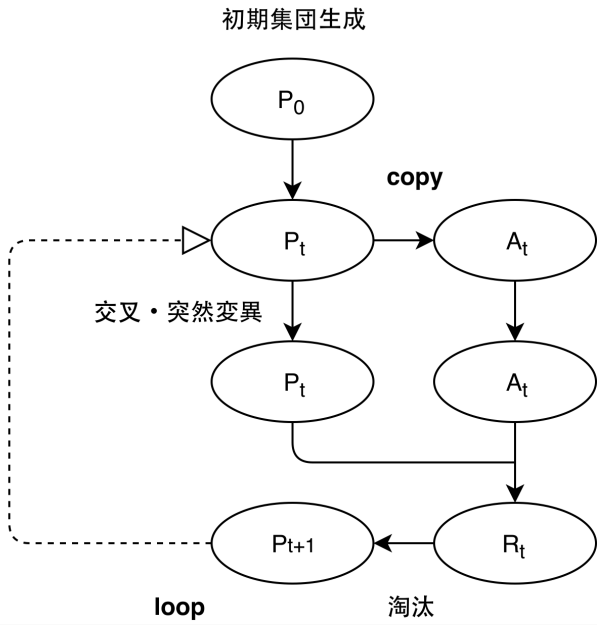

図 1 NSGA-Ⅱのアルゴリズム概要

3.1 個体の表現方法

各個体は小説の総文数の遺伝子を持っており，i 番目の遺伝子は小説の i 番目の文 s_i に対応する．また i 番目の遺伝子が”1”の場合は s_i は要約に含まれ，i 番目の遺伝子が”0”の場合は s_i は要約に含まれない．この小説の総文数個の遺伝子を持つ各個体が要約候補となる．

3.2 初期集団生成

まず遺伝的アルゴリズムでは初期集団 P_0 の生成を行う．本研究では各世代の制限個体数を 100 個体とし，初期集団 P_0 も 100 個体用意した．この際，要約長の制約を超えた個体に対し，遺伝子が”1”であるものからランダムに 1 つずつ”0”に更新し下記の式(1)を満たす個体を生成する．

$$\sum_{s_i \in S} |s_i| \le L \qquad (1)$$

ここで，S は要約候補に含まれる文の集合，L はあらかじめ設定された要約長である．

3.3 適応度の付与

適応度を持たない個体に対して，**4** 章で説明する目的関数を元に算出される適応度を付与する．

3.4 交叉・突然変異

t 世代目の集団 P_t をアーカイブ A_t にコピーする．その後 P_t に対して遺伝的操作を行なっていく．まず交叉を行う．交叉は 2 点交叉を行う．これは親となる 2 個体に対し，ランダムに 2 点交叉点を決め，交叉点に挟まれた部分の遺伝子を入れ替えた子個体 2 体を生成する操作である．また，交叉率は 10%とした．

次に突然変異を行う．突然変異は，個体の遺伝子からランダムに1つ選び，その遺伝子の”0”，”1”を反転させるという操作である．また突然変異率は 5%とした．また，交叉，突然変異が終了した後，個体が式(1)を満たすように要約長の制約を超えた個体の遺伝子が”1”であるものをランダムに 1 つずつ”0”に更新し，$R_t=P_t \cup A_t$ となる新たな集団 R_t を生成する．

3.5 ランク付け

i　R_t の各個体に対して，支配されている個体の数を数える．なお，最大化問題における支配されている個体の数とは，ある個体に対して各目的関数の数値が全て大きい個体の数である．

ii　支配されている個体が 0 である個体をランク r とする(初期値は r=1)．

iii　ランクが付与された個体を取り除く．ランクが付与されていない個体が残っている場合には，r＝r+1 とし i ～iii を繰り返す．

3.6 混雑度計算

R_t の各個体に対して以下の式(2)で与えられる混雑度を付与する．

$$d_i = \sum_{m=1}^{M} \frac{f_m^{i+1} - f_m^{i-1}}{f_m^{\max} - f_m^{\min}} \qquad (2)$$

ここで，d_i はランク r 内でソートされた個体群の i 番目の個体の混雑度，f_m^i は i 番目の個体の m 個目の適応度である．またランク r 内でソートを行った際の順位が最大のものと最小のものは適応度を無限大とする．この混雑度計算は全てのランクの全ての個体に対して行う．

3.7 淘汰

R_t は $R_t=P_t \cup A_t$ より制限個体数である 100 個体以上を有しているのでここで淘汰をおこなう．R_t 内の個体のうちランク r(初期値は r=1)の個体を制限個体数を超えない条件下で P_{t+1} に追加する．r=r+1 とし，上記の操作を繰り返し，初めて P_{t+1} が制限個体数を超えた場合，ランク r の個体を混雑度順にソートし，P_{t+1} が 100 個体になるように P_{t+1} に個体を追加する．

あらかじめ設定した世代数になるように **3.3~3.7** を繰り返し，設定された世代数に達した時 NSGA-Ⅱを終了する．

4 目的関数

目的関数には word2vec や fasttext といった言語概念ベクトルを生成するツールを用いた．fasttext とは，単語のベクトルを，単語を分割した n-gram のベクトルを足し合わせたものとして扱うツールである．これにより，活用形を考慮したベクトル生成，略語，未知語に対応しやすい等の利点がある．また今回の研究では word2vec，

fasttext どちらも同じ条件で学習を行った. 学習データとして 2017 年 9 月 7 日 16 時 30 分にダウンロードした日本語 Wikipedia データを使用, 学習方法は skip-gram, 学習の高速化手法は階層的ソフトマックス[2]を用い, 各パラメータについては, 学習に使用する前後の単語数は 5 単語,単語ベクトルの次元数は 300, 出現頻度が 15 回未満の単語は学習に使用しないと設定した.

4.1 内容の包括性

1 つ目の目的関数は, 要約前の文書に対する要約候補の包括性に関する関数である. 要約では, 元の文書の内容をできるだけ損なわず文字数を減らすことが重要である. 要約の前の文書と要約候補のベクトルのコサイン類似度を計算することによって意味的に類似した要約候補のスコアを高くする. まず, 要約前の文書, 要約候補のベクトルを以下の式(3), (4)に示す.

$$\vec{v}_D = \frac{\sum_{w_i \in D} \vec{w}_i}{|D|} \tag{3}$$

$$\vec{v}_S = \frac{\sum_{w'_i \in S} \vec{w}'_i}{|S|} \tag{4}$$

ここで, D と S はそれぞれ要約前の文書の単語集合と要約候補の単語集合であり, $\vec{w}_i$は単語 w_i のベクトルである. 式(3), (4)より1つ目の目的関数は以下の式(5)となる.

$$f_1 = \cos\left(\vec{v}_D, \vec{v}_S\right) = \frac{\vec{v}_D \cdot \vec{v}_S}{\|\vec{v}_D\|\|\vec{v}_S\|} \tag{5}$$

4.2 要約候補の冗長性

2 つ目の目的関数は, 要約候補文の冗長性に関する関数である. 要約文は文字数が少ないため, 冗長性を減らし, より多くの情報を含んだものが理想である. 要約候補中の各文同士のベクトルのコサイン類似度を計算することによって要約候補の冗長性を計算する.

$$f_2 = 1 - \frac{\sum_{s_n, s_m \in S} \cos\left(vec\left(s_n\right), vec\left(s_m\right)\right)}{|S| C_2} \tag{6}$$

ここで S は要約候補の文集合, $vec(s_n)$は文 s_n の文ベクトルである.

4.3 要約文字数

3つ目の目的関数は, 要約文字数に関する関数である. 遺伝的アルゴリズムでは, 遺伝的操作によって要約候補の文字数の変動が大きいため, 文字数が多い個体に高いスコアが付与されるように式(7)を設定した.

$$f_3 = \frac{\sum_{s_i \in S} |s_i|}{L} \tag{7}$$

ここで, S は要約候補の文集合, L はあらかじめ設定された要約長である.

5 評価実験

5.1 実験設定

青空文庫[6]から収集した 11 作品に対して実験を行った. 評価指標には ROUGE[7]の中の ROUGE-1 値を用いた. なお, ROUGE-1 とはシステム要約の 1-gram と参照要約の 1-gram がどの程度一致しているかという要約の評価指標の一種である. また, 評価指標に ROUGE を用いるにあたって必要な参照要約を Wikipedia に掲載されている各作品の「あらすじ」という項目の文書とした.

5.2 比較手法

まず, word2vec を用いて単語のベクトル化を行った手法と, fasttextを用いて単語のベクトル化を行った手法の 2 種類を提案手法とする. 比較手法として今野らの手法[8]を用いた. この手法も, 要約前の文書と要約候補の類似度を計算し最適解を探索するものである. 目的関数に Dice 係数を用いたもの, Jaccard 係数を用いたものの 2 つの手法が提案されているが, 今回は文献[8]で実験結果が最良であった Jaccard 係数を用いた手法を比較手法に用いた.

5.3 実験結果

今回の実験では, それぞれの手法において世代数を 250, 500, 750, 1000と変化させ結果を比較した. 図2は世代数を変化させた場合の各手法の ROUGE-1 値の変化を示したグラフである.

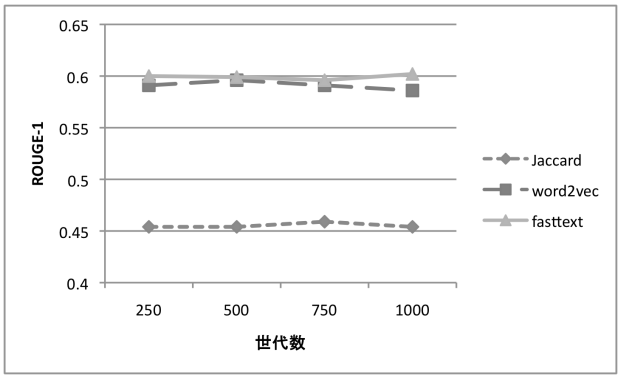

図 2 評価実験結果グラフ

今回の実験では世代数による ROUGE-1 値の変化はあまりなく, どの世代数においても Jaccard 係数を目的関数に用いた手法を大きく上回っていることが確認された.

また, それぞれの手法で最も 11 作品に対する

ROUGE-1 値の平均が高かった世代数と ROUGE-1 値を表 1 に示す.

表 1 評価実験の結果

	Jaccard	word2vec	fasttext
世代数	750	500	1000
ROUGE-1	0.459	0.596	0.602

表 1 の通り, 今回用いた手法の中で最も良い ROUGE-1 値を記録したのは, 目的関数に fasttext を用いた手法で, 1000 世代に設定されたものであった. また比較手法の Jaccard 係数を目的関数に用いた手法に比べ 0.143 ポイントの向上を確認した. また, word2vec を目的関数に用いた手法も, 500 世代に設定された時の ROUGE-1 値が 0.596 ポイントと fasttext との間に大きな差は見られなかった.

5.4 考察

表1, 図2から提案手法のROUGE-1値が比較手法を大きく上回っている. これは比較手法が Jaccard 係数という文書の表層的な情報しか扱っていないのに対し, 提案手法では文間の意味的な類似性に着目しているためであると考えられる. ここから要約において文間の意味的な類似性を考慮することが重要であると考えられる.

word2vec を目的関数に用いた手法と fasttext を目的関数に用いた手法の ROUGE-1 値は fasttext を目的関数に用いた手法の方が若干ではあるが高くなっている. これは, fasttext の特徴である subword の影響が大きいと考える. なお, subword とは, 単語を学習する際, 単語を n-gram に分割し, n-gram のベクトルの合計を元の単語のベクトルとする手法である. 例えば今回実験で用いた小説の中に「セロ弾きのゴーシュ」があるが, word2vec で作品中の単語をベクトル化すると 79 単語に対してエラーが発生してしまう. しかし, fasttext でベクトル化を行った際にはエラーが 47 個に減っている. より多くの単語をベクトル化することができた結果, より文の意味に近いベクトル同士の比較が行えたのではないかと考えられる.

fasttext でもベクトル化することができなかった単語には主に 2 種類の原因があった. まず 1 つ目は, 小説中に漢字であるべき単語がひらがなで表記されている部分がある点である. 例えば, 一般的に「一生懸命」と表記されている単語が小説中では「一生けん命」と表記されている. また,「飛びたち」と小説中で出現した結果, 形態素解析ツール MeCab[9]で上手く解析できず, 一単語として扱われていた等の問題起こっていた. この原因でベクトル化ができなかった単語は word2vec で 18 単語, fasttetx で 11 単語あった. 2 つ目は, 学習データの影響で擬態語や擬音語にとても弱いという点である. 今回用いた word2vec, fasttext のどちらの手法にも学習データに Wikipedia のダンプデータを使用した. この影響で「のそのそ」,「がさがさ」といった擬態語や擬音語をベクトル化できない事態が多数見受けられた. この原因でベクトル化できなかった単語は word2vec で 14 単語, fasttext で 11 単語存在した. この問題には, コーパスを変更, または拡充することによって対応していきたいと考えている.

6 ネタバレに関する予備実験

6.1 関連研究

今回は小説の要約において重要な要素であるネタバレを防ぐ手法について予備実験を行った. ネタバレとは, 小説や映画等の物語の結末, スポーツ等の試合結果など, コンテンツに触れる前に知ってしまうと楽しみが減ってしまうような情報である. ネタバレに関連する研究は盛んに行なわれている.

田島らの研究[10]では, Twitter においてリアルタイムでアニメを視聴した人のツイートにより, リアルタイムでアニメを視聴できない人がネタバレの被害を受けることを防止するため, ネタバレツイートの判定手法を提案している. 田島らは, ツイート中に含まれる単語の形態素解析, 構文解析結果から単語のベクトルを生成し, 機械学習によってネタバレツイートを判定している. 田島らの研究では, アニメの登場人物をネタバレ判定において重要な要素としてあげており Wikipedia からその情報を得ている. しかし, 小説は Wikipedia に掲載されていることが少ないので, この手法を応用することはできない.

また, 前田らの研究[11]では, レビュー文書におけるストーリー文書のネタバレ記述を発見する手法の検討がなされている. 前田らはネタバレ文には特徴的な単語が含まれており, それらの単語は小説中の後半に多く分布していると報告している. 前田らの研究から, 小説中の後半に多く分布する単語は特徴的な単語であると考え, 単語の分布から小説中のネタバレ部分を判定する手法の検討を行った.

6.2 手法概要

前田らの研究では, ネタバレ文に多く出現すると考えられる単語は文書中の後半に多く出現し, 品詞は名詞, 動詞が多いと報告されている. そこでまず, 小説中の後半に多い名詞, 動詞を抽出する. 小説を文数によって前半, 後半に分割し, 前半, 後半の名詞, 動詞をそれぞれ抽出する. また, あらかじめ小説中の名詞, 動詞の平均出現頻度を取得しておく, 平均出現頻度以上の出現頻度の単語のうち, 後半に比べ前半に多く出現する単語に対し式(8)で, 前半に比べ後半に多く出現する単語に対し式(9)でスコアを付与する.

表 2 評価者 3 名がネタバレ文と評価した文の一例

「モルグ街の殺人」のネタバレ文一例	
水夫が覗きこんだとき、その巨大な動物はレスパネエ夫人の髪の毛(ちょうど梳いていたので解いてあった)をつかんで、床屋の手ぶりをまねて、彼女の顔のあたりに剃刀を振りまわしていた。	猩々は、扉がうち破られるすぐ前に、避雷針を伝って部屋から逃げ出したにちがいない。

$$Score_F(w) = \frac{First(w)}{First(w) + Latter(w)} \quad (8)$$

$$Score_L(w) = \frac{Latter(w)}{First(w) + Latter(w)} \quad (9)$$

ここで First(w)は単語 w の前半での出現頻度, Latter(w)は単語 w の後半での出現頻度である. ま

た, 小説中の各文には以下の式(10)でスコアを付与する.

$$Score_s = \frac{\sum_{w_L \in L_s} Score\ (w_L) - \sum_{w_F \in F_s} Score\ (w_F)}{|L_s| + |F_s|} \quad (10)$$

ここで, Score(w_L), Score(w_F)は単語 w_L, w_Fのスコア, L_sは文 s に含まれる単語のうち, 後半に多く出現する単語集合, F_sは文 s に含まれる単語のうち, 前半に多く出現する単語集合である. また, これら重要であると考えられる単語の他にも, word2vec, fasttext を使用し単語集合を拡充した. 後半に多く出現する単語集合 L や前半に多く出現する単語集合 F 中の単語に対し, word2vec や fasttext を使って類似単語とその類似度を取得する. これらの情報から式(11)で類似単語のスコアを計算し, 新たに L または F に加える.

$$Score_B = Score(w) * Sim(w, w') \quad (11)$$

ここで, Score(w)は単語 w のスコア, Sim(w,w')は単語 w と w の類似単語 w'との類似度である. また, 今回は類似度が 0.6 以上の単語のみを類似単語として L または F に追加した.

6.3 実験設定

実験を行うにあたり手法を 4 つ用意した. 手法 1 は, 後半に多く出現する単語集合 L 中の単語が文中に一つでも存在する場合にネタバレ文として判定する手法. 手法 2 は式(8)から(10)を用いて前半に多く出現する単語集合 F と後半に多く出現する単語集合 L から文のスコアを計算し, 文のスコアが負になった場合にネタバレ文として判定する手法である. 手法 3 と手法 4 は, 手法 2 に加え, 式(11)も用いて F, L に類似単語を追加する手法である. また, word2vec を用いて類似単語を追加する手法を手法 3, fasttext を用いて類似単語を追加する手法を手法 4 とする.

また, 今回は「モルグ街の殺人」[12]を対象にしてネタバレ文の判定を行った. ネタバレ文は被験者 3 人に対してアンケートを取り 2 人以上がネタバレであると判定した文とした. また実験に際し, ネタバレ文を「これから作品を読む人が聞いたら楽しみが減ってしまう内容」と定義した. また, 被験者は 3 名とも 20 代理系大学院生男性である. アンケートの結果評価者 3 名がネタバレ文と判定した文の一例を表 2 に示す. また, アンケートの結果をまとめたものを表 3 に示す. アンケートの結果, 小説全文 852 文に対しネタバレ文は 42 文となった.

表 3 ネタバレ文に関するアンケート結果

	1 人目	2 人目	3 人目	結果
文数	106	46	11	42

6.4 結果

今回行った 4 つの手法による実験の結果は表 4 の通りである.

表 4 ネタバレに関する予備実験結果

	手法 1	手法 2	手法 3	手法 4
適合率	0.059	0.077	0.093	0.10
再現率	1.0	0.76	0.68	0.76
F 値	0.0065	0.14	0.16	0.18

表 3 より, 手法 4 が最も F 値が高くなっていることがわかる. また, 手法 1 に比べ手法 2 の F 値が高いことから, ネタバレ判定に後半に出現する単語だけではなく, 前半に多く出現する単語を考慮することが有効であることがわかる. さらに, 手法 2 に比べ手法 3, 手法 4 の F 値が高いことから, word2vec や fasttext によるツールを用いて類似語を追加する手法の有効性も確認できた. しかし, 全体的に適合率が非常に低く, F 値が低い結果となった.

6.5 考察

どの手法に関しても適合率が低い結果となった. 今回実験に用いた「モルグ街の殺人」は総文数が 852 文に対し, 最も適合率が良かった手法 4 に関しても, ネタバレ文と判定された文が 301 文あった. これはスコア付けの方法に問題があると考えている. 具体的には, 後半に多く出現する単語全てにスコアを付与してしまうと, 重要ではない単語にもスコアを付与してしまい, ネタバレをしていない文に対しても負のスコアをつけてしまうということである.

表 5 に前半に多く出現した単語, 後半に多く出現した単語の例を示す.

表 5 前半, 後半に多く出現した単語まとめ

前半に多く出現した単語	後半に多く出現した単語
証人, 証言, 死体 一, 二, 個,	殺す, 動機, 猩々 やる, 人間

「モルグ街の殺人」は, ある殺人事件に対して主人公である語り手が証人の証言をもとに犯人を推理する小説である. 前半部分では, 主人公が被害者の関係者や, 殺人現場付近に居た証人たちの証言を聞き, 後半は推理の結果, 犯人の手口や経緯を披露するというのが大まかな流れである. その流れを踏まえると, 証人, 証言, 死体といった単語が前半に多く出現し, 後半に殺す, 動機, 猩々といった単語が多く出現しているのは理想的な結果である. しかし, 数字, 単位, 普遍的な動詞, 名詞などあまり重要ではない単語も多く出現していた. 対策としては MeCab の解析結果が一般名詞, 固有名詞の単語にのみスコアを付与する, スコアの付与方法を前後半の出現頻度の差異だけでなく, 小説全体での出現文の位置情報を考慮するなどの方法が考えられる.

しかし, 手法 3, 4 の結果から類似単語を追加する手法に関しては有効性を確認することができたのは大きな収穫であったと考えている.

7 おわりに

本稿では, NSGA-Ⅱを用いた組み合わせ最適化による小説の要約システムの実装, 及び評価実験を行った. 評価実験より, 提案手法の ROUGE-1 値が比較手法に対して最大で 0.141 ポイント向上したことを確認した. また, この結果より, 要約における言語概念ベクトルの有効性を確認した.

後半のネタバレに関する予備実験では, 手法を 4 つ用意し, 予備実験を行った. 予備実験の結果から, ネタバレ判定における類似語の追加手法の有効性を確認した. また, 考察からスコア付けの式を見直す必要性があるという今後の課題を発見することができた.

本研究の最終目標はネタバレを含まない小説の自動要約である. 当面の目標は, ネタバレ判定手法のF値を実用レベルまで向上させることである. その後, 本稿の提案手法と組み合わせ, ネタバレを含まない小説の自動要約システムの実装を目指す予定である.

参考文献

[1] Deb, K., Pratap, A., Agarwal, S. and Meyarivan, T.: A fast and elitist multi-objective genetic algorithm: NSGA2, Proc. of IEEE Transactions on Evolutionary Computation, Vol. 6, No. 2, pp. 149-172, 2002.

[2] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S. and Dean, J.: Distributed representations of words and phrases and their compositionality, Proc. of Advances in Neural Information Processing Systems, vol. 26, pp. 3111-3119. 2013.

[3] Bojanowski, P., Grave, E., Joulin, A. and Mikolov, T.: Enriching word vectors with subword information, ArXiv preprint arXiv:1607.04606, 2016.

[4] 小倉由佳里, 小林一郎: 多目的遺伝的アルコリスムを用いた組み合わせ最適化による要約生成, 言語処理学会第 21 回年次大会発表論文集, pp. 585-588, 2015.

[5] 住田恭平, 二宮崇: 言語概念ベクトルを用いた文書間類似度に基つく複数文書自動要約, 言語処理学会第 22 回年次大会発表論文集, pp. 513-516, 2016.

[6] 青空文庫, http://www.aozora.gr.jp/.

[7] Lin, C. Y. and Hovy, E.: Automatic evaluation of summaries using n-gram co-occurrence statistics, Proc. of the 4th Meeting of the North American Chapter of the Association for Computational Linguistics and Human Language Technology, pp.150-157, 2003.

[8] 今野勇気, 荒木健治: 遺伝的アルゴリズムを用いた文書間類似度による小説要約システムの性能評価, 第 10 回 Web インテリジェンスとインタラクション研究会資料, pp. 19-20, 2017

[9] Kudo, T., Yamamoto, K., Matsumoto, Y.: Applying Conditional Random Field to Japanese Morphological Analysis, Proc. of the 2004 Conference on Empirical Method in Natural Language Processing, pp. 230-237, 2004.

[10] 田島一樹, 中村聡史: Twitterにおけるアニメのネタバレツイート判定手法の提案, 第 8 回データ工学と情報マネジメントに関するフォーラム, B5-4, 2016

[11] 前田恭祐, 土方嘉徳, 中村聡史: ストーリー文書を用いたレビュー文書でのネタバレ検出に関する一検討, 2016 年度人工知能学会, 3O4-OS-04b-2in1, 2016

[12] エドガー・アラン・ポー: モルグ街の殺人事件, http://www.aozora.gr.jp/cards/000094/files/605_20934.html