強化学習を用いた文圧縮手法

松本 悠　　斎藤 博昭
慶應義塾大学大学院 理工学研究科
{matumoto, hxs}@nak.ics.keio.ac.jp

1 はじめに

元の文書から文を抜き出し，要約を作成するいわゆる抽出的文書要約において，文をそのまま抜き出すのではなく，抜き出して圧縮を行う文圧縮手法が重要である．これは，文書要約タスクにおいて字数制限が存在するため，そのまま文を抜き出すよりも，抜き出した文を圧縮し，字数を減らすことによって要約により多くの情報を加えることができるためである．文書要約は文書自体を要約することであるのに対し，文圧縮は個々の文の要約を生成するタスクと捉えることもできる．

文圧縮の代表的な手法としては，文に含まれる単語の集合を考え，この集合から不要と思われる単語を取り除いた部分集合を圧縮結果とする手法である [1]．文圧縮における研究では，圧縮結果に対しスコア関数を定義しスコアが最大となるような文を圧縮結果として出力する手法が主である．圧縮の際に係り受け関係を重視し，係り受け木の部分木を削除することによって，スコア関数の最大化，また文法的に正しい圧縮文を出力することを目指した手法 [2] や，文圧縮を整数計画問題に落とし込み最適化を行う手法 [3] などが挙げられる．近年では，後者の整数計画法を用い，単語だけでなく元文の構造を考慮し，スコアに組み込んで最適化を行う手法 [4] が成功をおさめている．このように，多数の情報をスコアに組み込むことでよりよい圧縮文が得られることが報告されているが，一方で整数計画法による文圧縮は NP 困難であり，上記のように多数の情報を利用したい場合は解法に工夫が必要となる．

そこで本稿では，機械学習の一種である強化学習を用いて文圧縮を行う手法を提案する．強化学習を自然言語処理に用いた研究として，文書要約を強化学習問題として定式化した研究 [6] が挙げられる，それによると，強化学習を用いた文書要約の精度が，整数計画法による結果を上回ったと報告されている．文圧縮の強化学習問題への定式化についても，実験を通じて有用であることを示した．

2 提案手法

本節では文圧縮，強化学習について説明し，提案手法について述べる．本提案手法における強化学習システムの実装については Ryang ら [6] の Automatic Sumarization using Reinforcement Learning を参考にした．

2.1 文圧縮

前節でも述べたように，文圧縮とは各文の要約であると捉えられる．つまり，単純な単語の削減等で文を圧縮しても良い文は得られない．良い圧縮文を得るためには，以下の 2 点に着目すべきである．

1. 元文に含まれる最も重要な情報を残す
2. 圧縮文は文法的に正しい

1 については元文と内容の食い違いがあると，文を要約したとは考えられず，また 2 については極端に文法がおかしい文が出力された場合，可読性に欠けるためよい圧縮文とはならない．

次に，文圧縮についての定式化を行う．S を m 個の単語からなる文，$\hat{S}$ を S の圧縮文とする．さらに T を S に含まれる単語の集合とし $\{t_i \in T | 1 \leq i \leq m\}$ とする．さらに $x_i \in \{0, 1\}$ を単語 t_i が $\hat{S}$ に含まれているかどうかを表す変数とする．単語 t_i の重要度を求める関数を $f(t_i)$ とすると，圧縮文 $\hat{S}$ のスコア $Score(\hat{S})$ は

$$Score(\hat{S}) = \sum_{t_i \in T} x_i f(t_i) \quad (1)$$

$$\text{s.t.} \sum_i x_i \leq K \quad (2)$$

と表せるせる，ただし K は圧縮文に含めてよい単語数の上限である．この $Score(\hat{S})$ が最大となる圧縮文 $\hat{S}$ を求めることを目標とする．ただしこの $Score(\hat{S})$ は単語の重要度のみを考慮しているため，文法的に正しくなるとは限らないため，実用的ではない．単語の重要度に加え，n-gram の情報や，係り受け関係などを考慮すべきであるがこれについては 2.5 で述べる．

2.2 強化学習

強化学習とは，ある環境下におかれたエージェントが試行錯誤を通じ現時点での状態から取るべき行動を決定し，環境から得られる報酬をもとに，最も多くの報酬が得られる方策を学習していく機械学習の一種である．エージェントは状態 s_t から行動 a_t を取ることで，状態 s_{t+1} に遷移し報酬 r_{t+1} を受け取る．本稿において状態は圧縮文に含める単語集合を指し，行動はまだ選択されていない単語集合から，どの単語を圧縮文に含めるか，または終了状態に移行するかを指す．これについては 2.4 で詳しく述べる．

2.3 TD(λ) 学習

TD(λ) 学習 [7] は強化学習アルゴリズムの一つである TD 学習の拡張である．TD 学習では，TD 誤差と呼ばれる値を 0 に近づけていくよう学習を進めていく．現在の状態を s_t とし，この状態の価値を $V(s_t)$ で表す．V は状態価値関数と呼ばれ，現在の状態を評価する関数である．TD 学習では，次のような式を用いて状態価値関数を更新する．

$$V(s_t) \leftarrow V(s_t) + \alpha(r_{t+1} + \gamma V(s_{t+1}) - V(s_t)) \quad (3)$$

上式において，$r_{t+1} + \gamma V(s_{t+1}) - V(s_t)$ が TD 誤差である．ここで α は学習率，γ は割引率，r_{t+1} は状態 s_{t+1} に遷移したときに環境から受け取る報酬である．TD 学習は 1 ステップごとに状態価値関数を更新していくのに対し，$TD(\lambda)$ 学習では，eligibility trace と trace decay parameter と呼ばれるものを用い，n ステップ後までの評価値を考慮できるような学習アルゴリズムである．ここで，eligibility trace $e_t(s)$ は

$$e_t(s) = \begin{cases} \gamma\lambda e_{t-1}(s) & (s \neq s_t) \\ \gamma\lambda e_{t-1}(s) + 1 & (s = s_t) \end{cases} \quad (4)$$

で表される．λ は trace decay parameter である．ここで，δ_t を TD 誤差とすると．TD(λ) 学習における状態価値関数の更新は

$$\Delta V(s_t) = \alpha\delta_t e_t(s), \forall s \in S \quad (5)$$

で表せる．本提案手法では，この TD(λ) 学習を用いて文圧縮を行う．

2.4 行動

文圧縮におけるエージェントの行動は，圧縮文に含めると決定した単語集合 (現在の状態) に，まだ含めていない単語集合 (可能な行動の集合) から単語を選ぶ (行動の選択) ことで，圧縮文に含める単語を増やしていく (次の状態に遷移)．エージェントは行動を選択する際に，圧縮を終了する (終了状態) という行動を選択できる．圧縮を終了した場合，環境から報酬を受け取る．また行動を選択した際，圧縮に含める単語数が上限を超えていた場合，強制的に圧縮を終了させペナルティを与える．文圧縮を開始してから終了するまでの一連の流れをエピソードと呼ぶ．エピソードを繰り返すことで，最も多く報酬がもらえるような方策を学習していくことを目標とする．図 1 に 1 エピソードの流れを示す．

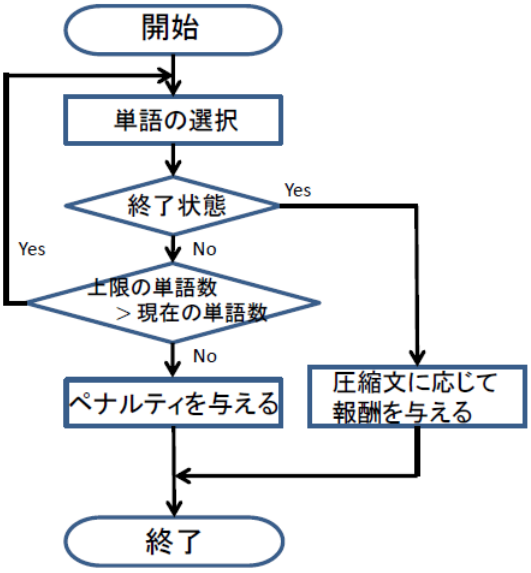

図 1: エピソードの流れ

行動の選択にはボルツマン選択を用いる．状態 s から可能な行動の集合を $A(s)$ とし，行動 $a \in A(s)$ を取ったときの行動価値と呼ばれる値を $Q(s,a)$ とする．ボルツマン選択では状態 s のときに行動 a を取る確率 $\pi(s,a)$ を

$$\pi(s,a) = \frac{\exp(\frac{Q(s,a)}{\tau})}{\sum_{p \in A(s)} \exp(\frac{Q(s,p)}{\tau})} \tag{6}$$

とする．ここで τ はいわゆる温度のようなパラメータであり，エピソードを重ねるごとに 0 に収束するようにしていく．ボルツマン選択では，学習の初期段階においては確率 $\pi(s,a)$ でいろいろな行動を試すが，学習の終盤においては，$\pi(s,a)$ は $Q(s,a)$ の大きいものほど 1 に近づくため，行動価値が大きくなるような行動を貪欲的に選択するようになる．

2.5 報酬

環境はエージェントが行動選択において終了状態を選択した際に，報酬またはペナルティを与える．本稿においては以下に列挙する情報をもとに報酬を与える．

- 含まれる単語の品詞，固有表現
- 元文に含まれていた bigram が圧縮文に含まれているか
- 元文で係り受け関係にあった単語どうしが圧縮文に含まれているか

2.1 で述べたように，含まれている個々の単語についてのスコアを考慮するだけでは文法的におかしい圧縮文が出力される可能性が大きい．そこで，元文に含まれている bigram や係り受け関係にある単語に対してスコアを与えることで，より文法的に正しい圧縮文が期待できる．本稿において，圧縮文 $\hat{S}$ に与える報酬関数 $Score(\hat{S})$ は

$$\begin{aligned} Score(\hat{S}) = & \sum_{t_i \in \hat{S}} f_{tok}(t_i) \\ & + \sum_{\substack{t_i \in \hat{S} \cap \{START\}, \\ t_j \in \hat{S} \cap \{END\}}} f_{bgr}(t_i, t_j) \\ & + \sum_{\substack{t_i \in \hat{S} \cap \{ROOT\}, \\ t_j \in \hat{S}}} f_{dep}(t_i, t_j) \end{aligned} \tag{7}$$

を用いる．式 (7) は [4] におけるスコア関数を参考にした．$f_{tok}(t_i)$ は単語 t_i についてスコアを与える関数，$f_{bgr}(t_i, t_j)$ は単語 t_i，t_j が元文でこの順番で出現している場合にスコアを与える関数，$f_{dep}(t_i, t_j)$ は単語 t_i，t_j が元文で係り受け関係にある場合にスコアを与える関数である．

圧縮文が上限の単語数を超えた場合は，報酬ではなくペナルティを与える．エージェントにペナルティを与えることで，単語数の上限を超える圧縮文の出力を抑制する働きを持つ．

3 実験

3.1 実験データ

文圧縮に用いるデータは newswire コーパス [1](NW コーパス) を用いる．このコーパスは元の英文と，1 名のアノテータが圧縮した文が一対一に対応したものである．アノテータは (1) 単語の削減のみで圧縮を行う，(2) 圧縮文の文法は正しくなるようにする，(3) 重要と思われる箇所を残す，という基準のもとに圧縮文を作成した．

3.2 評価について

本実験では，ngram-F1 値を評価値として用いる．Clarke ら [3] の整数計画法による手法を Thadani[5] が再実装したものをベースラインとした．また実験状況を同じにするために，NW コーパス内の 2 単語未満の文，110 単語より多い文を除去した計 1619 文を用いた．最後に報酬関数のパラメータ決定のため，データセットを学習セット (953 文)/検証セット (63 文)/テストセット (603 文) に分割した．

3.3 結果

本実験による結果を表 1 に示す．ngram-F1 値における n は n=1,2,3 で実験を行った．強化学習において，1 文の圧縮には 100 エピソードをかけて行い，圧縮文

表 1: 実験結果

	n=1 (%)	n=2 (%)	n=3 (%)
ベースライン	66.66	51.59	39.33
提案手法	72.10	53.16	42.99

[1] http://jamesclarke.net/research/resources/

の単語数の上限は，アノテータが作成した各圧縮文の単語数とした．行動選択におけるボルツマン選択の温度は現在のエピソード数を k として $\tau = 0.987^{k-1}$ とし，学習率 α は $\alpha = 0.001 \times 101/(100 + k^{1.1})$ とした．最後に割引率 γ は $\gamma = 0.95$，trace decay parameter λ は $\lambda = 1.0$ とした．

また，出力された圧縮文の一例を表 2 に示す．入力文

表 2: 出力の一例

入力文	When Los Angeles hosted the Olympics in 1932 , Kurtz competed in high platform diving.
アノテータ	When Los Angeles hosted the Olympics , Kurtz competed in high diving.
ベースライン	When Los Angeles hosted Olympics in 1932 , in high platform diving.
提案手法	When Los hosted the Olympics in 1932 , Kurts competed platform diving.

は圧縮される前の文，アノテータは人手による入力文の圧縮，ベースラインはベースライン手法による入力文の圧縮結果で [4] より引用した．最後に提案手法は本提案手法によって入力文を圧縮したものである．文法的な間違いはほとんど見受けられないが，Los Angeles の Angeles を選択できなかった点など，改善すべき点があると考えられる．

4 考察

今回の ngram-F1 値での評価において，文圧縮の強化学習問題への定式化は有用であったと考えられる．しかしながら，ベースラインと比較して n=2,3 においては提案手法による F1 値の向上幅はそれほど高くない．これは今回の報酬関数において，元文と同じ bigram があるかどうか，また元文と係り受け関係にあった 2 単語が存在するかという単純な特徴を報酬として用いたためであると推測できる．今後は言語モデルや Thadani ら [4] のように文の構造等に着目したスコアを考え，報酬関数を設計していくことが精度向上に繋がると考えられる．

また今回提案手法の実装においては，Ryang ら [6] のシステムを参考にしたため，強化学習の他のアルゴリズムで実装を行うことや，状態価値関数について，より文圧縮における報酬関数に適したものを設計していくことが必要であると考えられる．

最後に，今回の実験においては元文の長さに関わらず強化学習のエピソード数を 100 に固定したが，実行時間の問題等を考えると文の長さに応じてエピソード数を変化させ，またそれに応じた温度や学習率などのパラメータ調整が必要であると考えられる．これに関してはさらなる実験を行う必要がある．

5 おわりに

本稿では強化学習を用いた文圧縮手法を提案し，実験において整数計画法を用いたベースライン手法と比較することにより，その有用性について示すことができた．今後は考察でも述べたように，文構造等を考慮した報酬関数を設計，さらには強化学習の枠組みに文法等にペナルティを課すなどより実用的な文圧縮システムを構築していく予定である．さらに，実装したシステムを文書要約システムに組み込むことで，自然な要約を行うシステム構築まで目指していきたい．

参考文献

[1] Kevin Knight and Daniel Marcu. Summarization Beyond Sentence Extraction: a Probabilistic Approach to Sentence Compression. Artificial Intelligence. 139(1):91-107. 2002.

[2] Katja Filippova and Michael Strube. Dependency Tree Based Sentence Compression. In Proceedings of INLG, pages 25-32. 2008.

[3] James Clarke and Mirella Lapata. Global Inference for Sentence Compression: An Integer Linear Programming Approach. Journal for Artificial Intelligence Research, 31:399-429, 2008.

[4] Kapil Thadani and Kathleen McKeown. Sentence Compression with Joint Structural Inference. In Proceedings of CoNLL, pages 25-32. 2013.

[5] Kapil Thadani. Approximation Strategies for Multi-Structure Sentence Compression. In Proceedings of ACL, pages 1241-1251. 2014.

[6] Seonggi Ryang and Takeshi Abekawa. Framework of Automatic Text Summarization Using Reinforcement Learning. In Proceedings of the EMNLP, pages 256-265. 2012.

[7] R.S. Sutton and A.G. Barto. Reinforcement Rearning: An introduction, The MIT Press. https://webdocs.cs.ualberta.ca/ sutton/book/ ebook/the-book.html. 1998.