LSTM を用いた文の分散表現の獲得手法に関する一考察

福田 清人　　森 直樹　　松本 啓之亮
大阪府立大学 工学研究科
{fukuda@ss., mori@, matsu@}cs.osakafu-u.ac.jp

1 はじめに

近年, 計算機の爆発的な性能向上を原動力とした機械学習のブレイクスルーにより, 自然言語や画像といった離散的な記号の分散表現を獲得する試み [1][2] が大きな注目を集めている. 離散的な記号を実数値密ベクトルである分散表現に変換することで, 記号間に存在している意味や用法の類似性をベクトル間の距離として定義することができる. そのため, 離散的な記号の分散表現は, それらが持つ意味の関係性を考慮する必要がある様々なタスクへの応用が期待される.

その中でも特に自然言語処理の分野では, 単語の分散表現を獲得する手法である Word2Vec[3] が有名であり, 有効性が示されている. Word2Vec により得られた分散表現を用いることで単語の関係性を考慮することが可能となった. しかしながら, 1 文や文章の分散表現を獲得する手法 [4][5] は数多く研究されているにも関わらず, Word2Vec のような有効性を持つ手法はまだ報告されていない. そのため, 文書データが持つ情報を文単位で解析することは非常に困難な課題となっている.

以上の観点から, 本研究では文の分散表現の獲得に向けた前段階として, Long short-term memory (LSTM) に基づく Autoencoder を用いた文の分散表現獲得手法を提案する. また, 提案手法と既存の手法により得られた結果を比較することにより, 文の分散表現を獲得するために必要な情報や知見を得ることを目的とする.

2 関連研究

2.1 Word2Vec

Word2Vec[3] は単語の分散表現を得る手法であり, 単語の意味を文中で交換可能かどうかに注目して獲得していく. 分散表現の生成には文脈中の単語の情報を用い, 文中の分散表現を生成したい単語とその周辺の単語を予測するようにニューラルネットワークを学習する. 学習したニューラルネットワークの中間層の重みを分散表現として取得する.

Word2Vec における学習モデルには Skip-gram モデルと Countinuous Bag-of-Words（CBOW）モデルの 2 種類がある. Skip-gram は対象単語からその周辺単語を予測し，この周辺単語予測のエラー率の合計が最小になるように学習する. 一方 CBOW では逆に周辺単語から対象単語を予測するように学習する.

2.2 Doc2Vec

Doc2Vec[4] は上述した Word2Vec の考え方を文章にまで拡張した手法である. Doc2Vec には Distributed Bag-of Words (DBOW) モデルと Distributed Memory (DM) モデルという 2 種類の学習モデルがある. DBOW モデルは文章ベクトルから文章に含まれる各単語の単語ベクトルを予測するモデルであり, Word2Vec の Skip-gram モデルに対応した学習モデルとなっている. DM モデルは文章中の任意の単語ベクトルをその周辺の単語ベクトルおよび文章ベクトルから予測するモデルであり, Word2Vec の CBOW モデルに対応した学習モデルとなっている.

3 提案手法

本研究では LSTM に基づく Autoencoder を用いた文の分散表現獲得手法を提案し, 実際の文から分散表現を獲得する. 図 1 に提案手法の概要を示す.

3.1 入力と出力のデータ形式

提案手法では, 入力および出力は 形態素解析により 1 文を分割した各単語の分散表現である. 以下に提案手法における入力および出力データの生成アルゴリズムを示す.

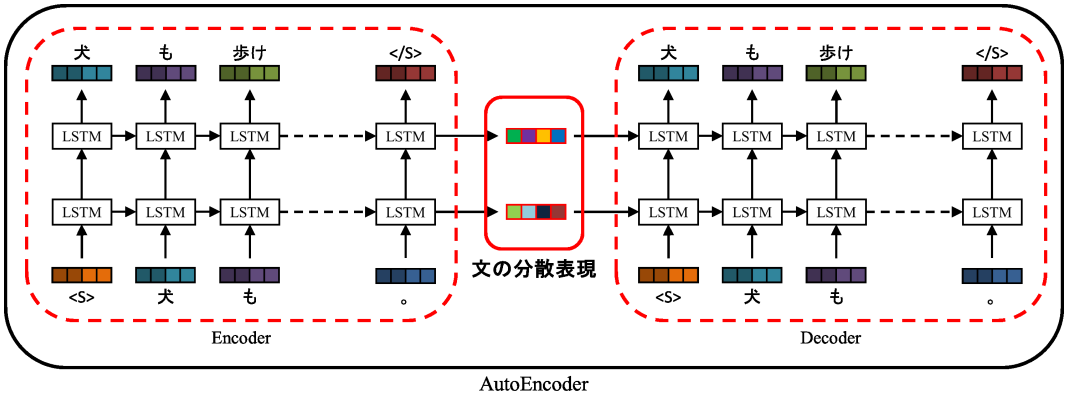

図 1: 提案手法の概要

1. 入力となる 1 文 s を形態素解析することによって N 個の単語列 $w_1 w_2 \cdots w_N$ を取得する.
2. 文頭を表す記号 w_{S} および文末を表す記号 w_{E} をそれぞれ文頭および文末に付与することで単語列を $w_{\mathrm{S}} w_1 w_2 \cdots w_N w_{\mathrm{E}}$ とする.
3. 単語列 $w_{\mathrm{S}} w_1 w_2 \cdots w_N w_{\mathrm{E}}$ の各単語 w に対して, 事前に学習した Word2Vec により分散表現 v_w を獲得する.
4. 入力データを文頭記号 w_{S} から文末 w_N までの分散表現の集合 $\mathcal{X} = \{\boldsymbol{v}_{w_{\mathrm{S}}}, \boldsymbol{v}_{w_1}, \cdots, \boldsymbol{v}_{w_N}\}$ とする. $|\mathcal{X}| = N + 1$ である.
5. 出力データを文頭 w_1 から文末記号 w_{E} までの分散表現の集合 $\mathcal{Y} = \{\boldsymbol{v}_{w_1}, \cdots, \boldsymbol{v}_{w_N}, \boldsymbol{v}_{w_{\mathrm{E}}}\}$ とする. $|\mathcal{Y}| = N + 1$ である.

以下, 単語の分散表現を単語ベクトル, 文の分散表現を文ベクトルと呼称する.

3.2 Encoder と Decoder

提案手法には自然言語処理の分野で有効性が示されている Encoder-Decoder モデルを用いる. Encoder-Decoder モデルにおいて, Encoder の入力と Decoder の出力を同一にし, Autoencoder として用いる. モデル全体を Autoencoder にすることで, Encoder と Decoder を連結する中間表現がその文の特徴を抽出したものになると期待できる. また, 文を扱うため, 時系列データを扱うことができる LSTM を用いて Encoder および Decoder のネットワークを構築する. 今回の提案手法では LSTM を多層構造にする. これは文ベクトルを獲得するうえで文の様々な特徴や情報を保存する必要があるためであり, LSTM を多層構造にすることで各 LSTM 層の隠れ状態ベクトルに異なる情報が保存されることが期待される.

3.3 モデルの学習と分散表現の獲得

Encoder および Decoder への入力および出力を, 学習用の文に対して 3.1 節で示した操作を実行することで生成された単語ベクトルの集合 $\mathcal{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_{N+1}\}$ および $\mathcal{Y} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_{N+1}\}$ とする. ここで, $y_i = x_{i+1}\,(i = 1, 2, \cdots, N+1)$ という関係になっていることに注意を要する.

以下にモデルの学習アルゴリズムを示す.

1. Encoder に入力として $\mathcal{X}$ の i 番目の要素 $\boldsymbol{x}_i$ を, 出力として $\mathcal{Y}$ の i 番目の要素 $\boldsymbol{y}_i$ を, 時系列順に与えていく. Encoder における $\boldsymbol{x}_i$ と $\boldsymbol{y}_i$ の誤差を ℓ_i^{e} とする.
2. 1 文から生成されたデータの入力後の LSTM 層の隠れ状態ベクトルが Decoder における LSTM 層の隠れ状態ベクトルの初期値となる.
3. Decoder には Encoder と同様, 入力として $\mathcal{X}$ の i 番目の要素を, 出力として $\mathcal{Y}$ の i 番目の要素を, 時系列順に与える. Decoder における $\boldsymbol{x}_i$ と $\boldsymbol{y}_i$ の誤差を ℓ_i^{d} とする.
4. Encoder および Decoder でのすべての誤差の合計 $\sum_{i=1}^{N+1} \left(\ell_i^{\mathrm{e}} + \ell_i^{\mathrm{d}}\right)$ をモデル全体の誤差として, 逆誤差伝搬法によりモデルを学習する.

また, 以下に文ベクトルの獲得アルゴリズムを示す.

1. 文ベクトルを獲得したい文 s から 3.1 節の操作を実行することで入力データ $\mathcal{X}$ を生成する.

表 1: Word2Vec の設定

モデル	Skip Gram
高速化手法	ネガティブサンプリング
文脈窓	3
ベクトルサイズ	200
サンプリングサイズ	5
epoch 数	10
最適化手法	Adam
学習率 α	0.000005
頻出頻度の閾値	5
語彙数	559676

2. Encoder に $\mathcal{X}$ の i 番目の要素 $\boldsymbol{x}_i$ を時系列順に入力する.

3. $\mathcal{X}$ の要素をすべて入力後, Encoder における各 LSTM 層の隠れ状態ベクトルを文 s の文ベクトルとして獲得する.

4 実験

本実験では, 提案手法と既存の手法により得られた文ベクトルを用いて対象文と候補となる文集合との類似度を求める. 対象文と最も類似度の高い文を結果として, 得られた文について考察することで文ベクトルを獲得するために必要な情報や知見を得ることを目的とする.

4.1 事前準備

提案手法では, 入力データとして単語ベクトルを獲得する必要がある. 本研究では有効性が示されており, 実装が比較的容易な Word2Vec を単語ベクトルの獲得手法として利用する. 表 1 に Word2Vec の設定を示す. Word2Vec の実装には Chainer を用い, 表 1 に記載していないパラメータについてはデフォルト値を用いた. Word2Vec 学習用のデータとして, 日本語 Wikipedia のテキストデータおよび, 小説投稿サイトである「小説家になろう」[6] から収集した各期間のランキング上位 100 件に含まれる 4566 作品を合わせた約 5.0 GB を用い, 頻出頻度の閾値以下の単語を未知語とした.

4.2 実験条件

表 2 に提案手法および比較手法である Doc2Vec の実験条件を示す. 提案手法の実装には Chainer を,

表 2: 実験条件

$n_{\min}$	10
$n_{\max}$	23
学習データ数	8898705 文
Encoder 構造	2 層 (LSTM, LSTM)
Encoder ユニット数	(200, 200)
Encoder 損失関数	平均 2 乗誤差
Decoder 構造	2 層 (LSTM, LSTM)
Decoder ユニット数	(200, 200)
Decoder 損失関数	平均 2 乗誤差
epoch 数	50
最適化手法	Adam
学習率 α	0.00001
Doc2Vec モデル	DM
文脈窓	3
ベクトルサイズ	200
頻出頻度の閾値	5

Doc2Vec の実装には Gensim を用い, 表 2 に記載していないパラメータについてはデフォルト値を用いた. 提案手法および Doc2Vec の学習データには, 小説投稿サイト「小説家になろう」 から収集した各期間のランキング上位 100 件に含まれる 4566 作品中の文章を 1 文に分解し, 台詞文を除いた $n_{\min}$ 単語以上 $n_{\max}$ 単語以下の文を用いた.

実験に用いる対象文集合と候補文集合には同一のものとし, 提案手法に用いた小説から, “異世界 (恋愛)”, “現実世界 (恋愛)”, “ハイファンタジー”, “ローファンタジー” という 4 ジャンルから各 1 作品をそれぞれ用いた. ここで, 候補文集合を対象文と同一ジャンルの文のみとしており, 候補文集合は対象文を除いたものとなることに注意を要する.

4.3 実験結果と考察

表 3 に実験に用いた各作品ごとの結果の例を示す. ここで, 表 3 の “提案手法 1 層” は第 1 層目の LSTM の隠れ状態ベクトルを, “提案手法 2 層” は第 2 層目の LSTM の隠れ状態ベクトルを用いた結果である. また, 文の類似度は文ベクトル間のコサイン類似度を用いて求めた.

表 3 において, 対象文と最も類似度が高かった文を比較すると, 提案手法を用いた場合は語尾の用法が体言止めなら体言止め, 過去進行形なら過去進行形という風に類似したものとなっている. また, “冷たく白い” という表現がある対象文に対して, “真っ白” という似た意味の表現がある文が類似度の高い文として現れた. これらのことから, LSTM 層の隠れ状態ベクトルを文ベクトルとすることで, 限定的ではあるが文の

表 3: 実験結果の例

手法	対象文	最も類似度の高い文	類似度
提案手法 1 層	誰に突っかかろうと、それは彼女達の自由。	とはいえ、流石は貴族の当主を勤める方々。	0.9942
提案手法 2 層		問題は、そんな彼女がとった行動。	0.9986
Doc2Vec		あの場がどれだけ息苦しかろうと、どうせ全ては自分を置き去りに進んでいくのだ。	0.7700
提案手法 1 層	冷たく白い腕に抱きしめられていた。	視界の全てが真っ白に染まっていた。	0.9972
提案手法 2 層		崩れゆく私の姿を見て、泣いていた。	0.9998
Doc2Vec		私は今、暖かい人生を歩んでいる。	0.5937
提案手法 1 層	それは分かったけど……あれ、何？』	『……それって、俺のことか？』	0.9736
提案手法 2 層		『……それって、俺のことか？』	0.9961
Doc2Vec		……何か、面倒そうだな。	0.4784
提案手法 1 層	このまま自滅してくれるんならそれに越したことはないからな。	俺一人じゃ全部食べるのは無理がある気がするからな。	0.9992
提案手法 2 層		異世界に行きました何て頭の可笑しい事は言えないからな。	0.9999
Doc2Vec		まああんなん見たらそうなるよな。	0.3907

意味や用法といった情報を保持することができると考えられる.

しかしながら, 文全体を見た場合, 意味的に類似度が高いとは言い難い文が選ばれている. 提案手法には, 単純な LSTM や Autoencoder を用いているだけなので, より複雑な構造についての検討が必要であると分かった.

一方, Doc2Vec を用いた場合, 人間が見た場合に提案手法よりも不適な文との類似度が高くなってしまった. Doc2Vec は文章の分散表現を文章中の単語を用いて学習するため, 1 文では学習に十分な量の単語が存在しておらず, Doc2Vec の学習が不十分になってしまうと考えられる. このことから, 文ベクトルを獲得するためには, 文章の分散表現の獲得手法とは異なる手法を用いるべきであることが分かった.

5 まとめと今後の課題

文の分散表現を取得することを最終目標として, 比較的単純な構造で文の分散表現獲得手法を提案手法として実装した. また, 文の分散表現獲得手法と既存の文章の分散表現獲得手法を用いて分散表現から類似度の高い文を獲得し, その結果を考察することで, 適切な文の分散表現を得るために必要な知見を得た.

今後の課題として, 以下に示すことが考えられる.

- 実験結果に対するユーザ評価を含むより詳細な解析
- 他の文の分散表現獲得手法の日本語文への適用
- 文の分散表現を扱うためのより適切なモデルの考察

参考文献

[1] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *Computer Vision and Pattern Recognition*, 2015.

[2] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, Vol. 14, pp. 1532–1543, 2014.

[3] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, Vol. abs/1301.3781, , 2013.

[4] Quoc V. Le and Tomas Mikolov. Distributed representations of sentences and documents. *CoRR*, Vol. abs/1405.4053, , 2014.

[5] Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. Skip-thought vectors. *arXiv preprint arXiv:1506.06726*, 2015.

[6] 小説家になろう - みんなのための小説投稿サイト. `https://syosetu.com/`.