ARG WI2 No.12, 2018
WI2-2018-06

Word2Vec出力側重みを用いた単語ベクトルの評価と応用

内田 脩斗[†,a]　吉川 大弘[†,b]　古橋 武[†,c]

†名古屋大学大学院工学研究科

a) *uchida@cmplx.cse.nagoya-u.ac.jp* b) *yoshikawa@cmplx.cse.nagoya-u.ac.jp*

c) *furuhashi@cmplx.cse.nagoya-u.ac.jp*

概要　Word2Vec を用いて単語の意味関係をベクトルへ埋め込む分散表現型単語ベクトルが近年注目を集めている．さらに，この単語ベクトルは，構文解析や文書分類など言語処理分野において幅広く利用されるようになり，その有効性も報告され始めている．一般的に，単語ベクトルとして利用されるのは Word2Vec 学習ネットワーク上の入力側重みであり，同時に生成される出力側重みは利用されない．これに対し著者らは，対となる出力側重みの有用性に着目し，本稿において，前者と後者を併用した単語ベクトルを提案する．また，実際に単語ベクトルの性能評価実験を行い，その有効性を検証するとともに，提案した単語ベクトルを応用した文書分類実験を行い，分類性能が向上することを示す．

キーワード　Word2Vec，分散表現，出力側重み，単語ベクトル，文書分類

1　はじめに

インターネットの普及に伴って大量の電子文書が日々生成されている現代において，テキストデータの自動解析手法や情報抽出技術は，様々な場面での応用が期待されている．一般的に，これらの言語処理技術は単語を原子単位（要素）として取り扱っている．広く普及している単語の表現手法として，個々の単語に固有のインデックスを与えることで単語を表現する One-hot 表現がある．この手法は非常にシンプルで分かりやすい反面，各単語が独立であることを前提としているため，同義語や類似語が全く関係のない単語として扱われることがある．また，1 単語に 1 次元を割り当てるため，ボキャブラリが増えると高次元となり扱いづらくなるという問題がある．これを解決するための手法が数多く研究されているが [1][2]，Micolov らが発表した Word2Vec[3] は，大規模コーパスから教師なし学習を行うことで，語義の似た単語が類似した重みを持つ分散表現と呼ばれる単語ベクトルを生成することができる．これにより，従来では困難であった単語ベクトル間での意味の演算が可能となり，学習された分散表現に対し，

$$vector(Paris) - vector(France) + vector(Italy)$$

により算出されるベクトルが $vector(Rome)$ に，

$$vector(king) - vector(man) + vector(woman)$$

により算出されるベクトルが $vector(queen)$ に，それぞれ近くなるという性質を持っている．

この Word2Vec は，ニューラルネットワークを利用した学習構造をしており，ネットワーク上の入力層-隠れ層間の重みである上述の分散表現と，それと共に生成される隠れ層-出力層間の重み，“出力側の重み”があるが，後者については一般的に利用されることはない．しかし，この出力側の重みは単語に対して分散表現とは異なる意味関係を捉えていると考えられ，その有用性があると思われる．そこで本稿では，分散表現と出力側の重みの捉えている意味関係の違いについて検討し，両者を併用した単語ベクトルを提案する．

2　Word2Vec

本章では．Word2Vec の学習モデルについて説明する．

Word2Vec は，言語処理でよく用いられる分布仮説（同じ文脈で出現する単語は同じ意味を持つこと）[4] に基づいており，文脈上のある単語に対して，周辺に現れやすい単語を予測することをモデル化した構造をしている．

図 1 は Word2Vec の学習モデルを表した図である．また，入力層と出力層の次元数はボキャブラリ数，隠れ層は埋め込み次元数である．

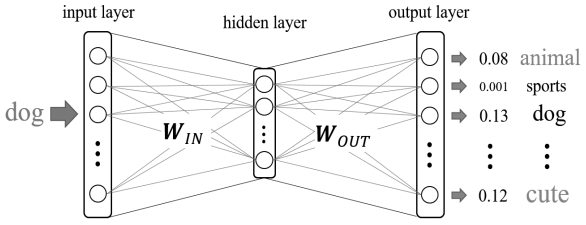

図 1 Word2Vec 学習モデル図

Word2Vec はニューラルネットワークを用いて分散表現を獲得する．具体的には，入力層に入力された単語に

対して，その単語の周辺に出現しやすい単語の出現確率が大きくなるように各層の重み（入力側重み W_{IN}，出力側重み W_{OUT}）を更新する．例えば，ある文脈上において,「dog」という単語の周辺に「animal」や「cute」という単語が出現したとすると，入力層の「dog」に対応した要素に 1，それ以外の要素に 0 が入力され,「animal」,「cute」に対応した出力層の出力が 1 に近づくように W_{IN} と W_{OUT} が更新される．

W_{IN} は周辺単語の情報を元に，単語の意味関係を学習している．つまり,「dog」や「cat」という単語の周辺に「animal」や「cute」という単語が共通して出現することで類似した重みを学習し，

$$vector(dog) \simeq vector(cat) \simeq vector(rabbit)$$

に近づくように，個々の次元に単語の意味関係を埋め込んでいる．また，W_{IN} の各行ベクトルが個々の単語ベクトルに対応しており，Word2Vec では W_{IN} を分散表現として利用することを前提とした学習モデルとなっている．

一方，W_{OUT} はある単語の周辺に出現する単語を予測するための重みである．つまり,「dog」や「cat」という単語の周辺に「animal」や「cute」という単語が現れる確率を大きくするため，入力単語のベクトルと周辺単語のベクトルの内積値が大きくなるように重みを学習する．これより，W_{OUT} は単語ベクトルを共起しやすい単語に展開する共起単語ベクトルであると捉えることができる．つまり，W_{OUT} においては，

$$vector(animal) \simeq vector(cute) \simeq vector(bark)$$

のような共起関係を学習する傾向にある．また，W_{OUT} の各列ベクトルが個々の共起単語ベクトルに対応している．ただし，W_{OUT} は W_{IN} を獲得するために生成される副産物と考えられており，一般的に W_{OUT} を単語ベクトルとして利用することはない．W_{OUT} を利用した従来研究は数少ないが，Mitra ら [5] は，W_{IN} と W_{OUT} の双方から得られる共起情報を利用した文書検索手法を提案している．しかし，W_{OUT} を単語ベクトルとして，独立して用いる本研究とは異なっている．

上述したように，W_{IN} と W_{OUT} では学習される重みの性質が異なっており，通常利用されない W_{OUT} にも有用性があると考えられる．

3 提案手法

本章では，W_{IN} と W_{OUT} の性質を利用した新たな単語ベクトルの生成手法を提案する．

3.1 連結型単語ベクトル

複数の分散表現を結合する手法のひとつに，Yin らが提唱した連結型単語ベクトル [6] がある．具体的には，複数のコーパスを用いて独立に分散表現を獲得し，ベクトルの次元を拡張してそれぞれの分散表現を連結する．つまり，次元数が 100，50，300 の 3 種の分散表現を生成したとき，次元数 $k = 100 + 50 + 300 = 450$ となる．これにより，単語ベクトルの表現力の拡張と単語のカバレッジの向上が期待できるとされている．本稿では，この手法を W_{IN} と W_{OUT} に適用し，W_{CONC} を生成する．これにより，W_{IN} と W_{OUT} の捉えているそれぞれの特徴を含有したベクトルが生成されると考えられる．

3.2 加算平均型単語ベクトル

ここでは，W_{IN} と W_{OUT} の重みの関係性に注目した手法を提案する．2 章で述べた通り，W_{IN} と W_{OUT} には，共起関係を含有した重みが学習されている．実際には，W_{IN} と W_{OUT} との内積をとると，その上位に来る単語のペアは，同一の単語のペアとなる（$\overrightarrow{dog_{IN}}$ と $\overrightarrow{cat_{OUT}}$ よりも，$\overrightarrow{dog_{IN}}$ と $\overrightarrow{dog_{OUT}}$ の内積の値が大きい）傾向が確認されている．その結果を表 1 に示す．表 1 は，W_{IN} と W_{OUT} の内積上位単語ペアにおける同一となった単語ペアの割合を示している．Top-n は上位何単語まで許容するかを表すパラメータである．ただし，対象とした単語は 4 章で使用するデータセット内の単語に限っている．

表 1 内積上位単語ペアの単語一致率

Top-n	Match rate [%]
1	91.2
2	94.2
3	95.1

これより，W_{IN} と W_{OUT} の同一単語ベクトル（$\overrightarrow{dog_{IN}}$ と $\overrightarrow{dog_{OUT}}$ など）は類似したベクトル構成になっていることが考えられ，この性質を利用し，(1) 式を用いて，新たな単語ベクトル W_{MEAN} を生成する．

$$W_{MEAN\_word} = \frac{1}{2}(\overrightarrow{word_{IN}} + \overrightarrow{word_{OUT}}) \quad (1)$$

W_{MEAN_word} は，新しく生成される単語ベクトルを表している．また，加算平均型単語ベクトルでは，ベクトルの次元数に変化はない．

4 実験

本章では，[2] で紹介されている手法を用いて，単語ベクトルの性能評価を行う．

4.1 統語論的意味関係テスト

統語論的意味関係テストとは，単語ベクトルがどの程度単語の意味関係を捉えられているかを評価する方法である．

本実験では，マイクロソフト社にて公開されている

MSR Word Relatedness Test Set[1] を用いた．上記のデータセットには，「$good : better, rough : \_\_\_$」，「$sell : sold, win : \_\_\_$」のような文法的な意味関係を反映した単語セットが計 8000 セット含まれている．

4.2 評価方法

1. $a : b = c : d$ という関係性の単語セットであるとき，d を未知とする．
2. 各々の分散表現 W 内の $\vec{a}$，$\vec{b}$，$\vec{c}$ を利用し，$\vec{y} = \vec{b} - \vec{a} + \vec{c}$ を算出する．
3. $\vec{y}$ と $\vec{d}$ の Cos 類似度を算出することで，単語ベクトルの性能を計測する．

また，単語ベクトルが存在しない単語が含まれている単語セットの場合は，除外して評価を行う．

4.3 Word2Vec 学習コーパス

Word2Vec の学習コーパスには英語 Wikipedia を利用した．また，各種パラメータは，$window = 5$，$size = 300$ を使用した．「$window$」は前後何単語を教師データとするかを指定するオプション，「$size$」は学習する単語ベクトルの次元数を指定するオプションである．Word2Vec の実装は Python の gensim ライブラリを使用し，また，W_{OUT} はライブラリ内の syn1neg に保存されているものを使用した．

4.4 実験結果

図 2 に，統語論的意味関係テストの結果を示す．実際に評価可能な単語セット数は計 6820 セットであった．なお今回は，対象単語はデータセット内出現単語に限っている．Top-n は，上位何単語まで許容するかを表すパラメータである．よって，Accuracy は，Top-n 単語内に正解単語が出現する割合を表している．また，Word2Vec は内部にランダム性を保持しているため，今回は 5 試行平均での結果を表示している．

図 2 より，W_{IN} と W_{OUT} では W_{IN} の方が精度が高いことが確認できる．W_{OUT} は共起関係を含有しているものの，単語ベクトルとしての性能は W_{IN} よりも低いと考えられる．また，W_{CONC} は W_{IN} と W_{OUT} の中間を推移していることが確認できる．これにより，W_{CONC} は両者の異なる意味関係の捉え方を融合したベクトルになっていると考えられる．また，W_{MEAN} が，最も単語ベクトルとしての性能が高いことが確認でき，提案手法の有効性を示唆している．

以上の結果を考察するために，等分散の検定（両側検定）を行った．W_{IN} と W_{MEAN} の 4.4 で利用した 930 単語を対象とし，それぞれで総当りによる単語ベクトル間の Cos 類似度を算出し，分散を算出したところ，$Var_{IN} = 0.00780$，$Var_{MEAN} = 0.0125$ となった．これより，$F = Var_{MEAN}/Var_{IN} = 1.61$ である．また，F 分布は自由度 $({}_{930}C_2 - 1, {}_{930}C_2 - 1)$ に従い，有意水準 5%において $F_{0.025} = 1.006$ である．よって，$F > F_{0.025}$ となり，帰無仮説は棄却される．また，Var_{IN} と Var_{MEAN} の関係性より，W_{MEAN} の単語ベクトルのばらつきは W_{IN} のばらつきよりも大きい，すなわち，単語ベクトル間の距離や角度が大きくなったといえ，単語ベクトル同士の意味的な区別がしやすくなったことが，性能が向上した要因の一つとなったと考えられる．

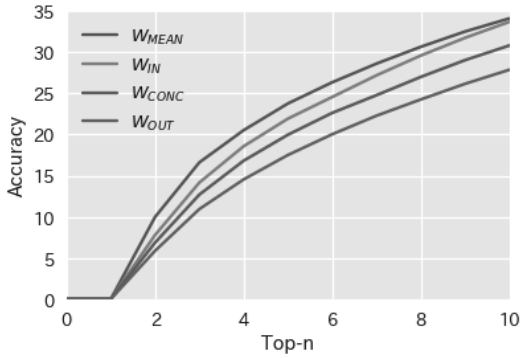

図 2 統語論的意味関係テスト

5 文書分類への応用

本章では，提案手法を文書分類に応用し，分類精度への貢献を検証する．文書をベクトル化する手法は，[7] で紹介されている文書内単語の平均ベクトルを用いた．

5.1 分類対象データセット

実験で使用した分類対象データセットを表 2 に示す．livedoor ニュースコーパスは日本語のテキストデータとなっており，ウェブサイト [2] からダウンロードして利用できる．また，Reuters 21578 は英語のテキストデータとなっており，[8] の著者のウェブサイト [3] からダウンロードして利用できる．

表 2 分類対象データセット

データセット	学習文書数	テスト文書数	クラス数
livedoor	4420	2947	9
Reuters 21578	5485	2189	8

5.2 Word2Vec 学習コーパス

Word2Vec の学習コーパスには，日本語 Wikipedia と英語 Wikipedia，さらに，各データセットの学習文書を

[1] https://www.microsoft.com/en-us/research/project/recurrent-neural-networks-for-language-processing/

[2] https://www.rondhuit.com/download.html

[3] http://web.ist.utl.pt/acardoso/datasets/

表 3 各データセットにおける文書分類精度

Dataset(Word2Vec)	W_{IN}[%]	W_{OUT}[%]	W_{CONC}[%]	W_{MEAN}[%]
livedoor(Wiki)	87.47	87.41	87.73	**87.81**
livedoor(学習文書)	91.07	91.20	91.55	**91.83**
Reuters(Wiki)	93.22	92.94	93.33	**93.58**
Reuters(学習文書)	93.84	93.88	94.03	**95.31**

利用した．諸条件は 4.3 と同様である．

5.3 分類器

分類器には SVM を用いた．SVM は RBF カーネルを用い，ハイパーパラメータはライブラリのデフォルト値である $C = 1$ と $\gamma = 1/$次元数 (300) を用いた．

5.4 実験結果

表 3 に，各データセットと，Word2Vec 学習コーパス別に，個々の単語ベクトルを用いて文書分類を行った際の分類精度を示す．数値は 5 試行の平均であり，分類精度は，テスト文書に対して正しく分類された割合である．

表 3 より，W_{CONC} と W_{MEAN} では，W_{IN}（従来手法）よりも精度の向上が確認できる．W_{CONC} の精度が向上した原因として，一般的に単語ベクトルの次元数は，大きい方が分類精度が向上する傾向があることが挙げられる．W_{CONC} は他の単語ベクトルの倍の次元数を持つため，分類精度の向上につながったと考えられる．また，どのデータセットにおいても，W_{MEAN} の精度が最も高いことが確認できる．一般的に，文書分類の精度と単語ベクトルの性能には相関があるため，4.4 で示した通り，W_{MEAN} が単語ベクトルとしての性能が高いことで，文書を的確に特徴づけることが可能となり，分類精度の向上につながったと考えられる．

また，Word2Vec の学習コーパスを学習文書にすることで，全体的に精度が向上している．これは，文書特有の単語や表現を学習することが可能であるためと考えられる．また，W_{IN} と W_{OUT} を比較すると，Word2Vec の学習に Wikipedia と学習文書を利用した時で大小関係が入れ替わっていることが確認できる．学習文書を Word2Vec の学習に利用した場合，W_{OUT} の単語ベクトルに共起性，つまり，文書情報が含有されると考えられる．よって，分類文書の文書傾向が一致したことにより，W_{IN} よりも文書を的確に表すことが可能となり，精度の向上につながったと考えられる．

6 まとめ

本稿では，Word2Vec の学習過程で生成される出力側の重み W_{OUT} に注目し，入力側の重み W_{IN} と併用した単語ベクトルを提案した．実験の結果，提案する単語ベクトルの性能が，従来分散表現として用いられてきた W_{IN} よりも高いことを示した．さらに，文書分類に応用した実験においても，提案する単語ベクトルの有効性が確認できた．W_{OUT} は，W_{IN} の生成と同時に，常に副産物的に生成されるため，W_{MEAN} の生成に必要な追加データや時間的なコストがほぼ 0 であるというメリットがある．さらに，ベクトルの次元数を増やすことなく単語ベクトルの性能を向上させられるため，計算コストも抑えることが可能であると考えられる．

今後の課題として，他のデータセットにおける有効性の検証や，W_{OUT} のより有効的な活用方法について検討することが挙げられる．

参考文献

[1] Andrew, L., Maas and Andrew Y. Ng.: A Probabilistic Model for Semantic Word Vectors, In NIPS Workshop on Deep Learning and Unsupervised Feature Learning, 2010.

[2] Mikolov, T., Yih, W.T., Zweig, G.:Linguistic Regularities in Continuous Space Word Representations, NAACL HLT 2013.

[3] Mikolov, T., Chen, K., Corrado, G., et al.: Efficient estimation of word representations in vector space, arXiv preprint arXiv:1301.3781, 2013.

[4] Hinton, G.E., McClelland, J.L. and Rumelhart, D.E.: Distributed representations, In: McClelland, J.L., Rumelhart, D.E. (Eds.), Parallel Distributed Processing: Explorations in the Microstructure of Cognition. MIT Press, Cambridge, MA, pp. 77109, 1986.

[5] Mitra, B., Nalisnick, E., Craswell, N., et al.: A dual embedding space model for document ranking, arXiv preprint arXiv:1602.01137, 2016.

[6] Yin, W and Schutze, H.: Learning word meta-embeddings, In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, Berlin, Germany, pages 13511360, 2016.

[7] Liu, R., Wang, D. and Xing, C.: Document classification based on word vectors, in ISCSLP ’ 14, 2014.

[8] Cardoso-Cachopo, A.: Improving Methods for Single-label Text Categorization, PdD Thesis, Instituto Superior Tecnico, Universidade Tecnica de Lisboa, 2007.