B0TB2225

卒業論文

分布意味論における
係り受け関係を用いた句ベクトルの構成的生成

村岡雅康

2014年3月27日

東北大学
工学部 情報知能システム総合学科

分布意味論における
係り受け関係を用いた句ベクトルの構成的生成*

村岡雅康

内容梗概

近年、句や文の意味をそれらを構成する単語から計算する構成意味論の研究に注目が集まっている。句や文の意味を計算し、それらの間に成り立つ類似度を計算機が扱うことができるようになることは、自然言語処理の応用上非常に有用である。現在構成意味論に基づく手法が様々提案されているが、3 単語以上の句や文の意味の計算が考慮されていない、構文解析など特定のタスクを解くことに主眼を置いているために句や文の意味が正しく計算される保証がない、パラメータ数が膨大になってしまうために学習が難しいなどの問題がある。これらの問題に対処するため、本稿では係り受け関係に基づくニューラルネットワークモデルを提案し、この有用性を評価実験により示す。また、学習により得られた重み行列を可視化することで、係り受け関係ごとに個別に重みを学習することが有用であることを定性的に示す。

キーワード

自然言語処理, 分布意味論, 構成意味論, 機械学習, 係り受け関係, ニューラルネットワーク

*東北大学 工学部 情報知能システム総合学科 卒業論文, B0TB2225, 2014 年 3 月 27 日.

目 次

図 目 次

表 目 次

1　はじめに

計算機による自然言語の理解において、言葉の意味を正しく計算できることは大きな目標の一つである。単語の場合、多くの研究では分布仮説 [1] と呼ばれる仮説に基づいたアプローチ (分布意味論) が取られている。分布仮説とは、任意の単語は、同じ文脈で出現する単語の分布からその単語の意味が推定可能であるとする仮説である。例えば、以下のような文を考える。

知人が北海道のお土産に「き花」を買ってきた。

上記の文において、「き花」という単語の意味が分からなかったとしても、それと共起している単語から意味をある程度推測できる (上述の例では、「き花」とは北海道のお土産の一つであることが推測できる)。さらに、この未知語が出現する文章を大量に収集すれば、その意味をより精密に推測できるようになる。また、「着物」と「和服」、「シューズ」と「スニーカー」など似た文脈を持つ単語は似た意味を持つと推定できる。このような方法を用いれば、計算機も単語の意味を扱うことができる。

しかし、同様の方法で句や節などを 1 単語とみなし、句や節の意味を捉えようとした時、新たな問題が生じる。それは、句や節が長くなるほどそれらの出現頻度が指数関数的に減少し、正しく意味を推測できなくなるというものである。そこで近年、句や節、文の意味をそれらを構成する個々の単語の意味から計算するというアプローチ (構成意味論) の研究に注目が集まっている。現在様々な手法が提案されているが [2, 3, 4, 5, 6, 7, 8, 9, 10]、後述するように 3 単語以上の句や文の意味の計算が考慮されていないため 2 単語からなる句の意味は計算できるがそれ以上の句や文の意味が計算できない (再帰性がない)、Socher ら [10] の手法のように特定のタスクを解くことに主眼を置いているため句の意味が正しく計算される保証がないなどの問題がある。さらに、既存研究では修飾関係や目的語関係など明らかに性質の異なる合成に対して、異なる方法で意味の計算を高精度かつ再帰的に行うことができない。これらの問題に対処するため、本稿では係り受け関係に基づくニューラルネットワークモデルを提案し、その有用性を評価実験により示す。

本稿の構成は以下の通りである。まず 2 節では単語ベクトルを構築する方法お

よび単語から構成的に句や文のベクトルを生成する既存のモデルを紹介する。3節で提案手法を説明し、4節でモデルの学習方法および評価実験の方法・結果を述べる。最後に5節で本研究の全体の総括を行う。

2　関連研究

本節では単語の意味を表すベクトルの代表的な構築方法および構成的に句や文のベクトルを生成する既存研究について述べる。

2.1　単語ベクトルの構築

分布意味論では、単語の意味はd次元空間上の一点、すなわちベクトルで表す。その構築方法は、以下の2種類に大別できる。

・共起頻度を用いる方法 [2, 11] - この方法は各単語についてテキストデータ中で共起する単語の統計をとり、ノイズ除去・スパースネス解消のため主成分分析 (Principal Component Analysis, PCA) 等で次元圧縮して得られたものを単語ベクトルとする方法である。

・ニューラルネットワークを用いた言語モデルで学習する方法 [12, 13] - この方法は言語モデルを学習する過程でニューラルネットの誤差逆伝搬により単語ベクトルを学習する方法である。

2.2　構成的な句ベクトルの生成

まず、構成性の原理および問題の定式化を説明し、その後単語ベクトルから構成的に句や文のベクトルを生成するモデルの代表例を紹介する。

構成性の原理 [14] とは、句や文の意味はそれらに含まれる単語の意味から構成されるという考え方である。例えば、「鮭をくわえた熊の木彫り」という句の意味は、語順を無視すれば、「熊」「鮭」「くわえる」「木彫り」の4単語から構成されると考える。句や文の意味を表すベクトルをそれを構成する単語のベクトルから生成することは構成性の原理に基づいている。

以降の説明を容易にするため、「2つの単語ベクトルから句ベクトルを生成する」という問題を数学的に定式化する。2つの単語ベクトル$\boldsymbol{u}$と$\boldsymbol{v}$をあるモデルfに入力として与え、モデルfは句ベクトル$\boldsymbol{p}$を出力する。これは次式で表せる:

$$\boldsymbol{p} = f(\boldsymbol{u}, \boldsymbol{v}) \tag{1}$$

表 1: 既存モデルの代表例

モデル	数学的表現	パラメータ
add[2, 3]	$w_1\boldsymbol{u} + w_2\boldsymbol{v}$	$w_1, w_2 \in \mathbb{R}$
Fulladd[4, 5]	$W\begin{bmatrix}\boldsymbol{u}\\ \boldsymbol{v}\end{bmatrix}$	$W \in \mathbb{R}^{d\times 2d}$
RNN[6]	$\sigma\left(W\begin{bmatrix}\boldsymbol{u}\\ \boldsymbol{v}\end{bmatrix}\right)$	$W \in \mathbb{R}^{d\times 2d}$
Lexfunc[7, 8]	$A_u\boldsymbol{v}$	$A_u \in \mathbb{R}^{d\times d}$
Fulllex[9]	$\sigma\left(W\begin{bmatrix}A_v\boldsymbol{u}\\ A_u\boldsymbol{v}\end{bmatrix}\right)$	$W \in \mathbb{R}^{d\times 2d}$ $A_u, A_v \in \mathbb{R}^{d\times d}$

σ : 活性化関数 (tanh などのシグモイド関数)

ただし、関数 f は単語ベクトル $\boldsymbol{u}, \boldsymbol{v}$ から句を表すベクトル $\boldsymbol{p}$ を計算する演算を表す。また、入力と出力のベクトルの次元を等しくすることで再帰的な生成が可能となり、3 単語以上からなる句や文のベクトルを生成できる。

表 1 に、これまでに提案された代表的なモデルを示す。

・加算 (**add**) モデル [2, 3]:二つのベクトルを重みを用いて線形結合するモデル。

・全加算 (**Fulladd**) モデル [4, 5]:2 つの単語ベクトルを線形変換するモデル。

・再帰的ニューラルネット (**RNN**) モデル [6]:重み行列で 2 単語を線形変換した後、活性化関数 (tanh などのシグモイド関数) を用いて非線形変換を行うモデル。

・**Lexfunc** モデル [7, 8]:従属語 (例:形容詞) を行列、主要語 (例:名詞) をベクトルで表現し、それらを乗じることで新たにベクトルを得る。直感的理解としては、例えば、修飾語 (従属語) は被修飾語 (主要語) に対して、何らかの性質を変換する働きを持つと考えると、ベクトル空間上では行列、すなわちベクトル空間の変換として表現されるということである。

・行列ーベクトル再帰的ニューラルネット (**Fulllex**) モデル [9]:Lexfunc,RNN の一般化で単語は意味を表すベクトルと他の単語に与える作用としてはたらく行列の組として表現され、ニューラルネットによって合成される。

表 2: 既存手法の問題点

モデル	表現力	再帰性	学習のしやすさ
add[2, 3]	×	○	○
Fulladd[4, 5]	×	○	○
RNN[6]	×	○	○
Lexfunc[7, 8]	○	×	△
Fulllex[9]	○	○	×
提案手法	○	○	○

この他に乗算 (**Mult**) モデルや伸張 (**Dil**) モデル [2, 3] などが提案され、Dinu ら [15] はそれらの精度を同一条件下で比較し、Lexfunc モデルが最も優れていると述べている。その理由として、Lexfunc は言語学的な根拠、すなわち単語間に存在する関係 (例えば、修飾-被修飾関係、動詞-目的語関係など) を考慮した合成であることを挙げているが、単語の表現形式を統一していない (形容詞などの従属語を行列、名詞などの主要語をベクトルで表現している) ため、再帰的なベクトルの生成ができないという問題がある。一方、再帰的なベクトルの生成が可能な Fulllex モデルも RNN モデルも非線形変換を行う表現力の高いモデルではあるが、Fulllex モデルは各単語がベクトルと行列で表現されているため学習パラメータが膨大になり学習が難しい。また、RNN モデルは全ての単語の合成を一つの重み行列で行っているため、全ての単語の組み合わせに対応できるベクトルの合成を行うには自由度が不足し、Lexfunc に劣っている。以上をまとめると表 2 のようになる。

従って、本稿では Fulllex や RNN と同様に再帰的な生成が可能かつ、パラメータ数が Fulllex ほど多くなく、Lexfunc と同様に言語学的根拠に基づいた合成を行うモデルを提案する。具体的には合成時に使用する重み行列を係り受け関係によって使い分けるニューラルネットワークモデルを提案する。本研究に類似したモデルとして、Socher ら [10] が提案した品詞毎に重みを使い分けるモデルがあるが、このモデルは正しい構文構造を推定するという目的で設計され、合成された

ベクトルが正しい句や文の意味を表しているとは限らない。つまり、正しい構文構造を推定するために合成されたベクトルからその合成が構文構造として正しいかどうか (例えば、... 前置詞 限定詞 名詞 ... のような単語列に対して先に前置詞と限定詞の合成をするのではなく、先に限定詞と名詞を合成すること) を識別するのだが、この識別を行うためにはその合成されたベクトルに合成前の要素の統語情報が含まれてさえいればよく、それが正しい句や文の意味であるとは限らない。そこで本研究では句の意味を表す教師データを用いることで、この問題の解消が期待できると考えた。

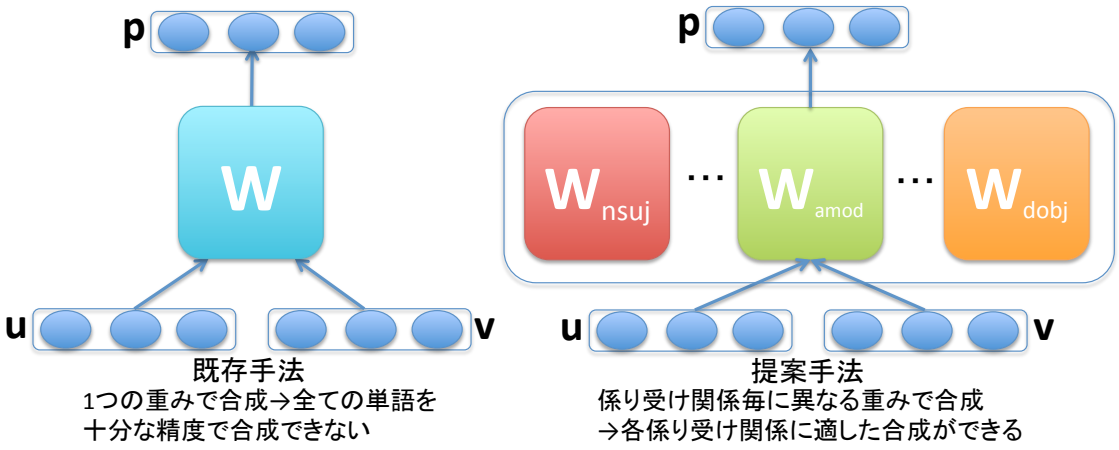


図 1: 既存手法と提案手法の比較

3 提案手法

一般的なニューラルネット (表 1、2 の RNN) による句ベクトルの生成は次式で表される (図 1 左):

$$\boldsymbol{p} = f(\boldsymbol{u}, \boldsymbol{v}) = \sigma\left(W \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{v} \\ b \end{bmatrix}\right) \tag{2}$$

ただし、$\boldsymbol{u}, \boldsymbol{v}$ は d 次元列ベクトル、W は $d \times (2d+1)$ 行列、b はバイアス項 (スカラ) である。また、$\sigma(.)$ はシグモイド関数であり要素毎に適用する。本研究では tanh を用いた。このモデルでは、形容詞による名詞の修飾や主語と動詞の合成、動詞と目的語の合成等、明らかに性質の異なる合成を 1 つの重みで行うため、それらすべてを十分な精度で合成することは困難である。これに対し提案手法は、2 つの単語ベクトルに加えてそれらの間に成り立つ係り受け関係 r も入力として使用する。この関係 r によって重み行列 W_r を決定する (図 1 右):

$$\boldsymbol{p} = f(\boldsymbol{u}, \boldsymbol{v}, r) = \sigma\left(W_r \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{v} \\ b_r \end{bmatrix}\right) \tag{3}$$

ここで $W_r \in \mathbb{R}^{d\times(2d+1)}, b_r \in \mathbb{R}$ は係り受け関係の種類だけ用意する。これにより、係り受け関係毎に異なる性質の合成を異なる重み行列で学習するため、既存のモデルでは達成できなかったより精密な合成が可能となる。

このモデルが正しい句のベクトルを出力するためには、入力となる単語ベクトル $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^{d\times1}$ が与えられたとき、それらによって構成される句の意味を正しく表すベクトル $\boldsymbol{p} \in \mathbb{R}^{d\times1}$ を出力するための関数 $f : \mathbb{R}^{(2d+1)\times1} \longrightarrow \mathbb{R}^{d\times1}$、具体的にはそのパラメータである重み行列 $\{W_r \in \mathbb{R}^{d\times(2d+1)}\}$ を学習しなければならない。これは次のような最適化問題を解くことで達成される。
訓練データ集合 $\{((\boldsymbol{u}_i, \boldsymbol{v}_i), \boldsymbol{t}_i)|i = 1...N\}$($\boldsymbol{t}_i$:教師句ベクトル) に対して定義される以下の誤差関数 $J(\theta)$ を最小化する:

$$J(\theta) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}||\boldsymbol{p}_i - \boldsymbol{t}_i||^2 + \frac{\lambda}{2}||\theta||^2 \tag{4}$$

ただし、θ は学習パラメータの全てを表し、$\theta = \langle W_r, b_r|r : \text{係り受け関係}\,\rangle$、$\lambda$ は L2 正則化項のパラメータであり、本研究では $\lambda = 10^{-4}$ とした。勾配の計算は一般的なニューラルネットワークモデルと同様に誤差逆伝播法を用いる。以降、図 2 で表されるフィードフォーワードニューラルネットワークを例として、誤差逆伝搬の手順を述べる。図 2 において、$k, k+1$ 層間を結ぶ重み行列 $W^{(k)} \in \mathbb{R}^{d^{(k+1)}\times d^{(k)}}$ の誤差逆伝搬法による更新式は次式で表される:

$$W^{(k)'} = W^{(k)} - \alpha\boldsymbol{e}^{(k+1)}\boldsymbol{p}^{(k)^T} \tag{5}$$

ただし、α は学習率、$\boldsymbol{e}^{(k+1)} \in \mathbb{R}^{d^{(k+1)}\times1}$ は $k+1$ 層における誤差、$\boldsymbol{p}^{(k)} \in \mathbb{R}^{d^{(k)}\times1}$ は k 層における出力、T は転置を表す。誤差 $\boldsymbol{e}^{(k+1)}$ は $k+1$ 層が中間層か出力層かによって次の 2 つの場合がある:

$$\boldsymbol{e}^{(k+1)} = \begin{cases} \frac{\partial J(\theta)}{\partial \boldsymbol{p}^{(k+1)}}\frac{\partial \boldsymbol{p}^{(k+1)}}{\partial W^{(k)}\boldsymbol{p}^{(k)}} & \text{(出力層のとき)} \\ (\boldsymbol{e}^{(k+2)^T}W^{(k+2)})^T\frac{\partial \boldsymbol{p}^{(k+1)}}{\partial W^{(k)}\boldsymbol{p}^{(k)}} & \text{(中間層のとき)} \end{cases} \tag{6}$$

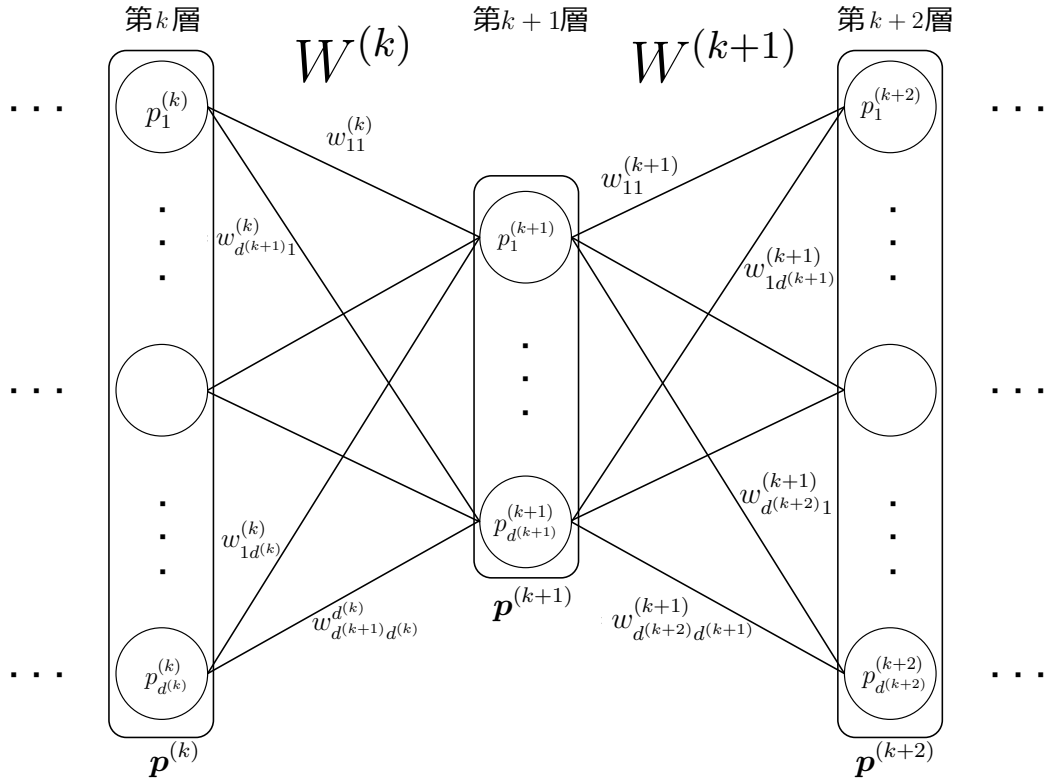


図 2: 一般的なフィードフォーワード型ニューラルネットワーク

本研究では中間層は存在しないので、$\boldsymbol{e}^{(k+1)} = (\boldsymbol{p}^{(k+1)} - \boldsymbol{t})(1 - \boldsymbol{p}^{(k+1)^2})$、$\boldsymbol{p}^{(k)} = \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{v} \\ b_r \end{bmatrix}$ となり、結局、更新式は次式で表される:

$$W_r' = W_r - \alpha(\boldsymbol{p} - \boldsymbol{t})(1 - \boldsymbol{p}^2) \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{v} \\ b_r \end{bmatrix}^T \tag{7}$$

また、重み行列の更新は 1 入力ごとに逐次的に行った。このような学習方法を確率的勾配降下法 (Stochastic Gradient Desdent, SGD) と呼ぶ。

4　評価実験

本節では、まず入力および教師ベクトルとなる単語・句ベクトルの作成方法およびモデルの学習方法を説明し、その後学習されたモデルの評価実験について述べる。

4.1　単語・句ベクトルの作成

単語・句ベクトルは Dinu らの論文 [15] を参考とし、以下のような手順で作成した。

1. ukWaC[16]†,Wikipedia(2009)[16]†,ClueWeb09‡の計約 38 億単語からなる大規模テキストデータから内容語 (名詞、形容詞、動詞、副詞) の頻度統計を求め、その結果の名詞・形容詞・動詞のみからなる上位 1 万語を語彙 V とする。
2. 語彙 V に含まれる単語のみで構成される 2 単語の句 (形容詞+名詞、名詞+名詞、動詞+名詞) の頻度を上述の大規模テキストデータから求める。
3. 語彙 V に含まれる各単語および出現頻度 1000 以上の句のそれぞれに対し、大規模テキストデータ中で同一文内かつ前後 50 語以内に共起する内容語の頻度統計を求め、単語および句と文脈語との共起行列を作る。
4. 共起行列の各要素の値を PMI(相互情報量)[17] に変換する。
5. 共起行列を EM アルゴリズムを用いた PCA[18][1]で $d = 200$ 次元に圧縮する。

これにより 10,000 種類の単語ベクトルおよび 17,433 種類の句ベクトルが得られた。

4.2　定量評価

評価実験は Mitchell ら [2][2]によって作成されたデータセットを用いて行った。

このデータセットは ⟨ 句 1, 句 2, 類似度 ⟩ の 3 つ組を 1 事例とし、2 つの句の意味的類似度を 7 段階で人手で付与してある。3 種類の品詞の組み合わせ (形容詞+

†http://wacky.sslmit.unibo.it/
‡http://lemurproject.org/clueweb09/
[1]大規模データに対応するためオンラインアルゴリズムに拡張した (付録参照)。
[2]http://homepages.inf.ed.ac.uk/s0453356/share

名詞, 名詞+名詞, 動詞+名詞) 毎に 108 事例ある。例えば、vast amount と large quantity は類似度 7、hear word と remember name は類似度 1、といった正解事例が収められている。評価は、モデルが生成した 2 つの句ベクトルのコサイン類似度と正解データの類似度とのスピアマン順位相関係数を計算し、モデルが生成した句ベクトルの類似度が人手による判断とどれだけ近くなるかを測定する。相関係数が高いほど、人間に近い類似性判断ができたことになり、モデルの性能が高いことを意味する。本研究では、Mitchell ら [2] と同様に、人手の類似度は平均値ではなく、それぞれ別のデータ点として計算した。

4.3 モデルの学習

評価データに含まれる句を除いた 16,845 種類の句を、それぞれテキストデータ中の出現回数の 1000 分の 1 回だけ重複して出現させた合計 $N = 175,899$ 句を訓練データとした。その他のハイパーパラメータは次のように設定した。

・学習率 $\alpha = 1.0 \times 1.1^{-l}$($l$: 現在の反復回数)
・L2 正則化項 $\lambda = 10^{-4}$
・収束条件: $l-1$ 回目の誤差と l 回目の誤差の差が 10^{-6} 未満
・反復回数の上限: 100

重み行列 W_r は 31 種類の係り受け関係に対して定義し、それぞれ学習開始時に次のように初期化した。

$$W_r = 0.01[\boldsymbol{I}_{n\times n}\boldsymbol{I}_{n\times n}\mathbf{0}_{n\times 1}] + \mathcal{N}(\mathbf{0}_{(2d+1)\times 1}, 0.001\boldsymbol{I}_{(2d+1)\times d}) \tag{8}$$

学習は 2.2GHz の計算機サーバ上で 10 並列で行い、学習に要した時間は約 7 時間であった[3]。

4.4 結果・考察

表 3 に評価結果を示す。corpus は句の共起ベクトル、add は表 1 において $w_1 = w_2 = 1.0$ とした加算モデルである。upper-bound は被験者間の相関であり、被験

[3]Python の numpy や multiprocessing のモジュールを使用した。

表 3: カテゴリ別スピアマン相関係数

	形容詞+名詞	**名詞+名詞**	**動詞+名詞**
corpus	0.362	0.432	0.215
add[3, 2]	0.442	0.432	0.404
Fulladd[4, 5]	0.406	0.420	0.366
RNN[6]	0.424	0.416	0.379
提案手法	**0.450**	**0.464**	**0.411**
upper-bound	0.539	0.490	0.505

全ての相関係数の間には統計的に有意な差がある ($p < 0.01$)

者 1 人とその他の被験者の相関を求め、最後にそれらの平均を取ることで算出した。corpus の精度が低いのは、そもそも評価データに含まれる句がコーパス中に出現せず、句ベクトル自体が求まっていないか (その場合の類似度は 0 とした)、出現頻度が小さすぎて十分な情報量を持ったベクトルにならなかったためと考えられる。また、全ての相関係数の間には統計的に有意な差が認められた ($p < 0.01$)。評価結果では add モデルが RNN や Fulladd よりも優れた強いベースラインになっているが、全ての句カテゴリにおいて提案手法が add モデルの精度を上回ったことが確認できた。このことより、句ベクトルの生成において係り受け関係毎に重み行列を個別に学習することは有用であることが確かめられた。

4.5 重み行列の可視化

ここでは実際に学習によって得られた重み行列の可視化を行った結果を示す (図 3、図 4)。実際に学習された重み行列 W_r は 401×200 行列であるため、値をそのまま可視化したとしてもその特徴を直感的に捉えることは容易ではない。そこで値の大小を色相の変化に対応させて可視化することで直感的理解が容易になると考えた。これらの図の見方は、中心から左右に二分した時、左半分は入力となる 2 つの単語ベクトルのうち左の単語ベクトルに対する重みであり、右半分は右の

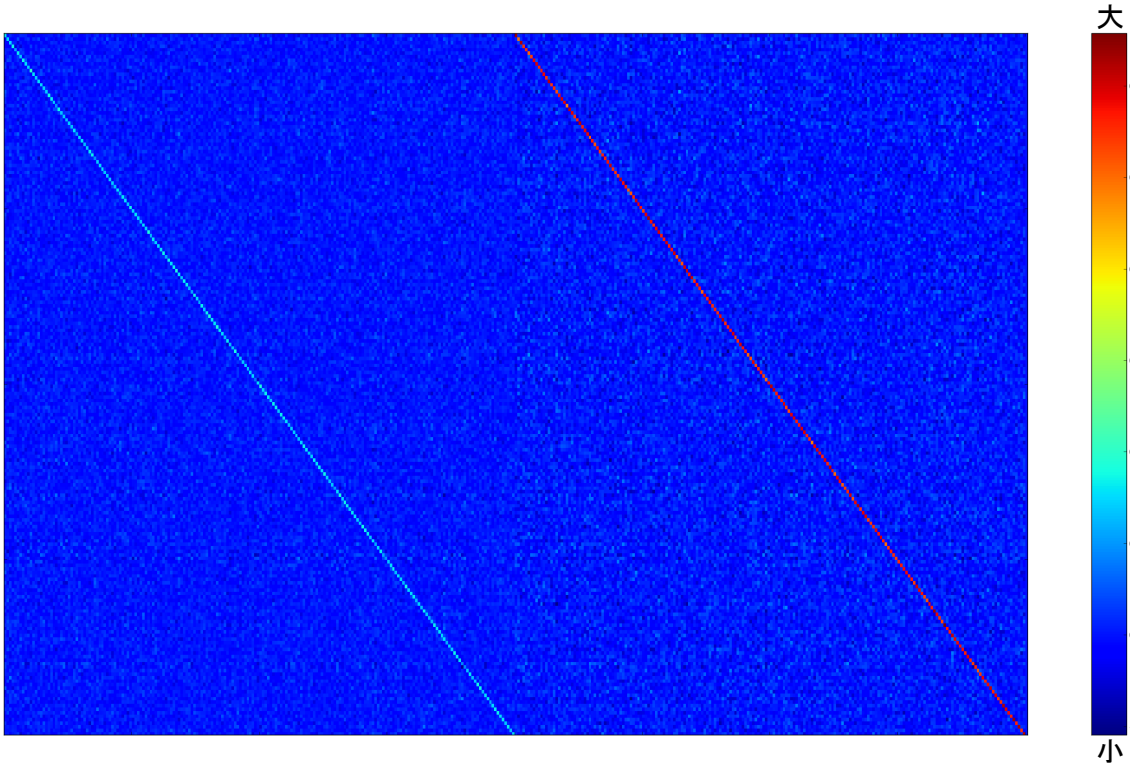
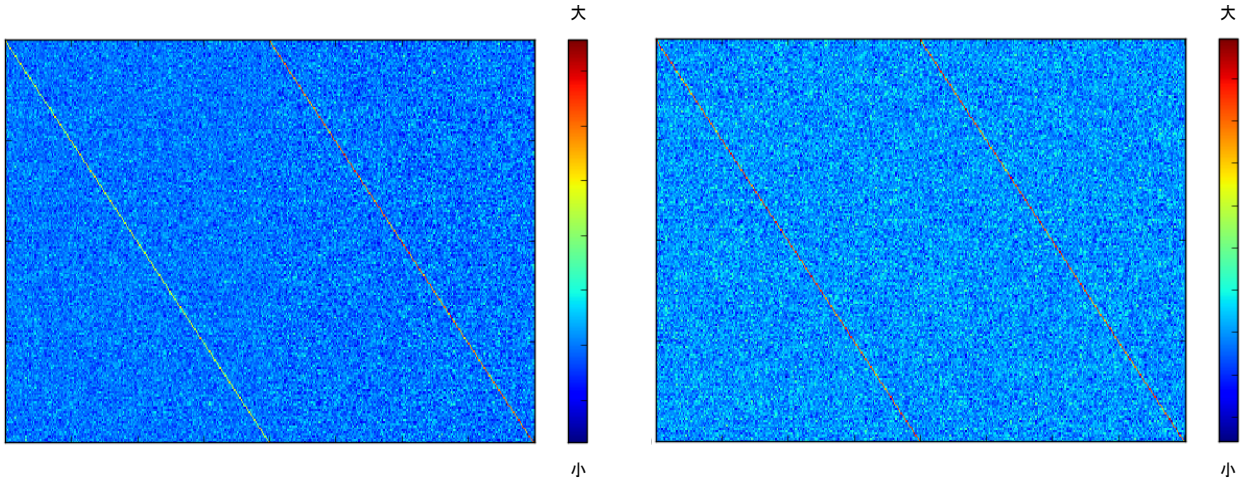
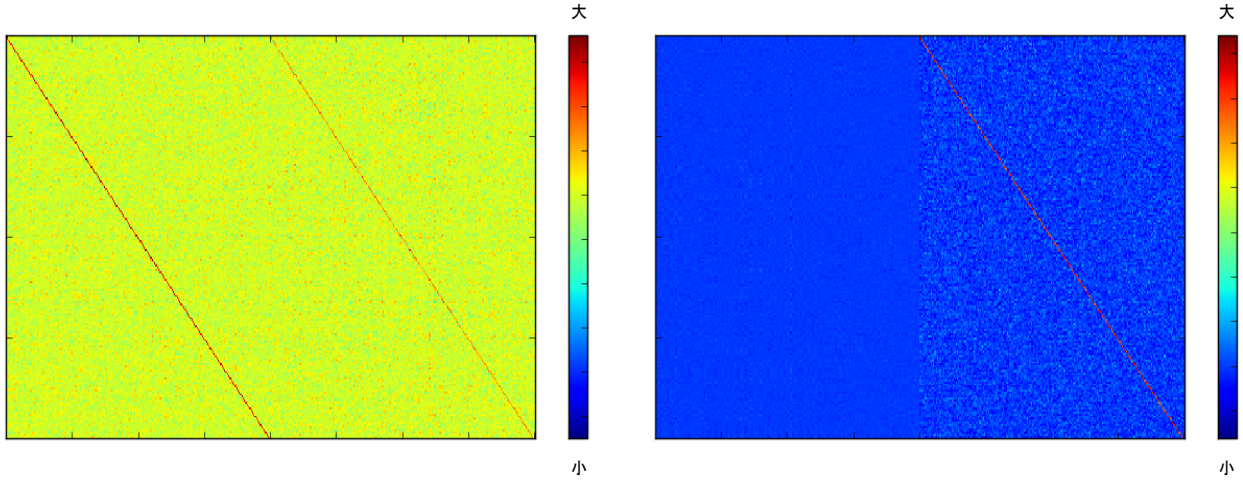


図 3: 既存手法 (RNN) の重み行列

単語ベクトルに対する重みである。図からそれぞれの対角成分が他の成分より値が大きくなっている (水色または赤) ことが確認できる。これは句ベクトルの第 i 要素を生成する時単語ベクトルの同じ第 i 要素からの影響が大きいことを意味する。また、図 3 の既存手法 (RNN) の重み行列に関して、右の対角成分の方が大きい (赤い) のは右の単語ベクトルがその句ベクトルにより貢献しているためである。言い換えれば右の単語ベクトルの方が重要であると言える。これは訓練データ中に「vast amount」や「large number」などのように主辞が後ろにある句が多かったためと考えられる。句の主辞が重要となるのは言語学的にも直感に合う結果である。しかし、実際は後ろに来る単語が必ずしも主辞となるとは限らず (反例「cars and bikes」)、それらの句の合成を既存手法 (RNN) の重みでは高精度に



(a) 修飾関係　　(b) 複合名詞



(c) 主述関係　　(d) 限定詞+名詞

図 4: 提案手法の重み行列

行うことができないと考えられる。それに対し、本手法 (図 4) のように言語的性質に基づいて重みを別々に学習することでこの問題に対処できると考えられる。以下、図 4 の説明を行う。

修飾関係の合成では修飾語 (左の単語) より被修飾語 (右の単語) がより重要 (主辞) であると考えられるため、右の対角成分がより大きな値 (赤) になっている。名詞の複合語の合成では先の反例のように、両方の名詞が同程度に重要であるため、対角成分も両方同じくらいの大きさになっている。主述関係用の重みは主語 (左の単語) をやや強めるような重みとなっている。これは同じ動作 (動詞) を行うにしても、動作主の違いが句の意味により大きな影響を与えると解釈できる。そし

て、特に傾向が顕著に現れているのが限定詞と名詞の合成を行う行列である。この行列は左の単語 (限定詞) に対する対角成分を視認することができない。これは句のベクトルの合成に限定詞は殆ど影響しないことを示している。このようにそれぞれ言語的な直感に合うような重みが学習されており、またそれぞれの重みが持つ傾向が異なることが確認できる。このことから、言語的性質が異なればそれぞれ異なる計算が必要となり、個別に重みを学習することは有用であるといえる。

5　おわりに

本稿では、初めに構成的に句のベクトルを生成する既存研究の代表例を紹介し、それら既存手法がもつ問題点について指摘した。具体的には言語的性質に基づく合成を行っていない、再帰性がない、パラメータの学習が難しいといったものである。それらの問題に対処するモデルとして、係り受け関係ごとに異なる重み行列を用いるニューラルネットワークモデルを提案し、評価実験において人手で作成されたデータセットを用いて、提案手法の有用性を実験的に示した。さらに、学習された重み行列を可視化することで、それぞれが異なる傾向を持つ行列となっていることを確認した。これは単語を合成するときに言語的性質毎にそれぞれ異なる計算が求められることを意味している。今後は、提案手法を拡張し、再帰的合成を行った場合の精度を言い換えのタスク等を通じて調査・検討したい。

謝辞

本研究を進めるにあたり、ご指導いただいた乾健太郎教授、岡崎直観准教授に感謝致します。

本論文の作成にあたり、終始適切な助言を賜った山本風人氏に感謝致します。

日常の議論を通じて多くの知識や示唆を頂いた乾・岡崎研究室の皆様に感謝致します。

参考文献

[1] John R. Firth. *Papers in linguistics 1934-51.* Oxford University Press, 1957.

[2] Mirella Lapata Jeff Mitchell. Composition in distributional models of semantics. *Cognitive Science*, Vol. 34, No. 8, pp. 1388–1429, 2010.

[3] Jeff Mitchell and Mirella Lapata. Vector-based models of semantic composition. In *Proceedings of ACL-08: HLT*, pp. 236–244, 2008.

[4] Emiliano Guevara. A regression model of adjective-noun compositionality in distributional semantics. In *Proceedings of the 2010 Workshop on GEometrical Models of Natural Language Semantics (GEMS '10)*, pp. 33–37, 2010.

[5] Fabio Massimo Zanzotto, Ioannis Korkontzelos, Francesca Fallucchi, and Suresh Manandhar. Estimating linear models for compositional distributional semantics. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pp. 1263–1271, 2010.

[6] Richard Socher, Cliff C. Lin, Andrew Y. Ng, and Christopher D. Manning. Parsing Natural Scenes and Natural Language with Recursive Neural Networks. In *Proceedings of the 26th International Conference on Machine Learning (ICML)*, 2011.

[7] Marco Baroni, Raffaella Bernardi, Ngoc-Quynh Do, and Chung-chieh Shan. Entailment above the word level in distributional semantics. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL '12)*, pp. 23–32, 2012.

[8] Bob Coecke, Mehrnoosh Sadrzadeh, and Stephen Clark. Mathematical foundations for a compositional distributional model of meaning. *CoRR*, 2010.

[9] Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 1201–1211, 2012.

[10] Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng. Parsing With Compositional Vector Grammars. In *ACL*. 2013.

[11] John A. Bullinaria and Joseph P. Levy. Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior Research Methods*, Vol. 39, No. 3, pp. 510–526, 2007.

[12] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *Journal of Machine Learning Research*, Vol. 3, pp. 1137–1155, 2003.

[13] Ronan Collobert and Jason Weston. A unified architecture for Natural Language Processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning (ICML '08)*, pp. 160–167, 2008.

[14] Gottlob Frege. On sense and reference. In *Ludlow (1997)*, pp. 563–584, 1892.

[15] Georgiana Dinu, Nghia The Pham, and Marco Baroni. General estimation and evaluation of compositional distributional semantics models. In *Proceedings of the ACL 2013 Workshop on Continuous Vector Space Models and their Compositionality (CVSC 2013)*, pp. 50–58, 2013.

[16] Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. The wacky wide web: a collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, Vol. 43, No. 3, pp. 209–226, 2009.

[17] Stefan Evert. *The statistics of word cooccurrences : word pairs and collocations*. PhD thesis, Universitt Stuttgart, 2005.

[18] Sam Roweis. EM algorithms for PCA and SPCA. In *Neural Information Systems 10 (NIPS'97)*, pp. 626–632, 1998.

[19] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B*, Vol. 39, No. 1, pp. 1–38, 1977.

付録

ここでは、EM アルゴリズムを用いたオンライン PCA について述べる。

d 次元空間上の 1 点を 1 データとし、それが n データあるとする。このとき、これらから構成される行列 $X \in \mathbb{R}^{d\times n}$ を PCA で k 次元まで次元削減するとき、必要な計算量は $O(n^3)$ である [18]。これは結果に不必要な特異値まで全て求めなければならないことが原因である。ここで、EM アルゴリズム [19] を用いると必要な k 個の特異値のみを求めるだけで次元削減できるためその計算量は $O(kdn)$ となり、これはデータ数 n に関して線形であるため大規模データに対しても容易に計算できる。

導出式は以下のようになる:

1. 行列 $W \in \mathbb{R}^{d\times k}$ を乱数で初期化

$$W = \mathcal{N}(\mathbf{0}, 0.01\boldsymbol{I}) \tag{9}$$

2. 収束するまで次を繰り返す。

$$\begin{cases} \text{E-step: } Z = (W^TW)^{-1}W^T\tilde{X} \\ \text{M-step: } W = \tilde{X}Z^T(ZZ^T)^{-1} \end{cases} \tag{10}$$

ただし、$\tilde{X}$ はデータを平均化したもので、$\tilde{X} = X - \bar{X}$ である。

最終的に Z が次元削減後の行列となる。上式は計算量が $O(kdn)$ であるものの、使用するメモリ空間は $X \in \mathbb{R}^{d\times n}$ であるため、$O(dn)$ である。これは特に本研究のように $k \ll d \ll n$ の場合、影響が顕著となる (本研究では $k = 200, d = 10,000, n = 27,433$)。そこで、これをオンライン型に変換することで以下のようにメモリ空間も $O(kd)$ に抑えることができる。

$$\begin{cases} \text{E-step: } \boldsymbol{z}_i = (W^TW)^{-1}W^T\tilde{\boldsymbol{x}}_i \\ \text{M-step: } W = [\sum_i^n \tilde{\boldsymbol{x}}_i\boldsymbol{z}_i^T][\sum_i^n \boldsymbol{z}_i\boldsymbol{z}_i^T]^{-1} \end{cases} \tag{11}$$

ただし、$\tilde{\boldsymbol{x}}_i \in \mathbb{R}^{d\times 1}$ は平均化された 1 データである。