超球面上での最適輸送に基づく文類似性尺度

横井 祥 [1,2] 高橋 諒 [1,2] 赤間 怜奈 [1,2] 鈴木 潤 [1,2] 乾 健太郎 [1,2]
[1] 東北大学 [2] 理化学研究所
{yokoi, ryo.t, reina.a, jun.suzuki, inui}@ecei.tohoku.ac.jp

1 はじめに

計算機によってふたつの文の意味的な類似性 (semantic textual similarity; **STS**) を人間と同様に計算できるようになれば，損失関数や自動評価尺度がより“正確な”値を返せるようになり，結果として種々の自然言語処理アプリケーションの性能向上が期待できる [19, 22].

STS の計算の指針のひとつは，二文に含まれる単語同士をなんらかの方法で対応づけることで二文の意味の重複の度合いを測るというものである [17]. とくに近年は**単語ベクトルのアラインメント**に基づく手法が多数提案されている [6, 21–23]. これらの手法は予測の可読性が高くまた文の構造的な特徴を考慮できるなど良い特徴を多く備えるが，一方で精度面では汎用的な**文ベクトル**を用いるアプローチに後れをとっている [4, 7].

本稿では，アラインメントに基づく STS 手法に残された精度面での課題を，単語ベクトルの大きさと向きに着目することで解決する．はじめに，単語ベクトルのノルムと方向ベクトルには単語の重要度と意味が分けてエンコードされていることを確認する．その上で，既存手法はこれらの情報を“混ぜて”取り扱ってしまっていることを指摘し，これらを別々に扱うための最適輸送に基づく新しい尺度を提案する．提案法は単語の方向ベクトルを超球面上で回転させてアラインメントをとるため，これを **Word Rotator's Distance (WRD)** と名付ける (§4).

さらに，高精度を達成している文ベクトル推定の成果を取り入れるため，一旦構成された文ベクトルを単語ベクトルの集合に再分解する方法を提案する．こうして得られた単語ベクトルには元々の学習済み単語ベクトルよりも重要度や意味の情報が強くエンコードされており，また WRD の入力としてそのまま利用できる (§5). 実験では WRD と文ベクトル分解のふたつの提案手法を組み合わせることで，複数のベンチマークテストで教師なし STS の過去最高精度を達成することを示す (§6).

1.1 問題設定・表記

STS は与えられたふたつの文の意味的な類似性を推定するタスクである [5]. 評価尺度としてはシステムの予測スコアと人手評価の間の (順位) 相関係数が用いられる[*1]. また本稿では [4, 7] にならって**教師なし**の設定を取り扱う.

形式的には，単語数 n, n' の 2 文 s, s' が与えられ

$$s = \{w_1, \dots, w_n\},\ s' = \{w'_1, \dots, w'_{n'}\}, \tag{1}$$

その類似度 $\mathrm{sim}(s, s')$ を推定する．各単語 w_i に対応する単語ベクトルを太字 $\boldsymbol{w}_i \in \mathbb{R}^D$ で表す.

2 関連研究

STS を計算するアプローチには大きく二種類ある.

ひとつめは与えられた二文の意味の重複度を計算するアプローチで [17]，近年は特に**単語ベクトルのアラインメント**に基づく手法が主流である．アラインメントの道具としては，注意機構 [21]，ファジィ集合 [23]，**Earth Mover's Distance (EMD)** [6, 9, 22] などが用いられる．とりわけ EMD は，確率分布間の距離としての精緻な理論が付いており [18]，構文木などの構造情報の組み込みも容易であることから [1]，とくに文生成の自動評価尺度として注目を集めている [22]．我々も単語ベクトルアラインメントの道具として EMD を採用する (§4).

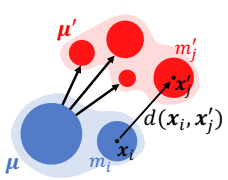

図1: Earth Mover's Distance.

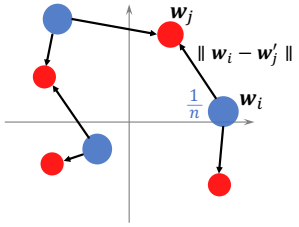

図2: Word Mover's Distance.

ふたつめは汎用的な**文ベクトル**を計算し，それらのコサイン類似度で STS を推定するアプローチである．文ベクトルの作り方には，生コーパスから深層文エンコーダを学習するもの [8]，言い換えコーパスを用いて「足し合わせに最適な」単語ベクトルを再学習するもの [12]，学習済み単語ベクトルから潜在文ベクトルを推定するもの [4, 7] がある．本稿では潜在“文ベクトル”を推定する手法 [4, 7] に着目し，これを“単語ベクトル”アラインメントに援用する方法を示す (§5).

3 準備：最適輸送コスト

3.1 Earth Mover's Distance (EMD)

直感的には，EMD は小分けに配置された荷物 (たとえば市内のいくつかの納豆工場に置かれた出荷予定の納豆) を別の配置 (たとえばホテルや小売店にそれぞれ必要量が入荷された状態) に移し替えるための最小の輸送コストのことである.

形式的には，EMD は以下を入力にとる (図1).

1. ふたつの確率分布 $\boldsymbol{\mu}$ (初期配置)，$\boldsymbol{\mu}'$ (輸送後の配置)：
$$\boldsymbol{\mu} = \left\{(\boldsymbol{x}_i, m_i)\right\}_{i=1}^{n}, \quad \boldsymbol{\mu}' = \left\{(\boldsymbol{x}'_j, m'_j)\right\}_{j=1}^{n'}. \tag{2}$$
ここで $\boldsymbol{\mu}$ は「各点 $\boldsymbol{x}_i \in \mathbb{R}^D$ に，**確率質量** (荷物の量) $m_i \in [0, 1]$ を対応させた確率分布」の意. $\sum_i m_i = 1$. 図1では各ペアを円で表し，その位置はベクトル $\boldsymbol{x}_i$，$\boldsymbol{x}'_j$ を，その大きさは重み m_i，m'_j を表す.
2. 任意の 2 点 $\boldsymbol{x}_i$，$\boldsymbol{x}'_j$ 間の輸送コストを定める**距離尺度** d：
$$d\colon \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}. \tag{3}$$

■ 最適化. EMD は，与えられたふたつの分布の間の輸送コストを最小化する最適化問題で定義される[*2]. 実用上はこの最

[*1] システムは文ペアの類似度の相対的な大小だけを予測できれば良い.

[*2] EMD の計算方法は以下の通り：

$$\mathrm{EMD}(\boldsymbol{\mu}, \boldsymbol{\mu}'; d) := \min_{\boldsymbol{T} \in \mathcal{C}(\boldsymbol{\mu}, \boldsymbol{\mu}')} \sum_{i,j} \boldsymbol{T}_{ij}\, d(\boldsymbol{x}_i, \boldsymbol{x}'_j) \tag{4}$$

$$\mathcal{C}(\boldsymbol{\mu}, \boldsymbol{\mu}') := \{\boldsymbol{T} \in \mathbb{R}_+^{n \times n'}; \boldsymbol{T}\mathbf{1} = \boldsymbol{m}, \mathbf{1}^\top \boldsymbol{T} = \boldsymbol{m}'\}. \tag{5}$$

適化を実装する必要はなく，高速なソルバを利用すればよい[*3]．

■ 副作用：アラインメント． 図3から分かる通り，上記の最適化問題を解く過程で，ふたつの分布の点同士にアラインメントが張られる (輸送計画行列が求まる)[*4]．実際 EMD はアラインメントの道具としても活用されており [16]，単語ベクトルアラインメントに利用される [6, 9, 22] 理由もここにある．

3.2 Word Mover's Distance (WMD)

Word Mover's Distance (WMD) は，EMD を利用した STS 計算手法の草分けである [9]．直感的には，WMD はまず各文を「単語ベクトルの集合 (離散分布)」と見なして，その間の EMD (最適輸送コスト) を測る (図2)．形式的には，各文 s，s' を単語ベクトルに一様な重みを持たせた離散分布とする：

$$\boldsymbol{\mu}_s := \left\{(\boldsymbol{w}_i,\, 1/n)\right\}_{i=1}^{n},\ \boldsymbol{\mu}_{s'} := \left\{(\boldsymbol{w}'_j,\, 1/n')\right\}_{j=1}^{n'}, \quad (6)$$

図2では各単語を円で表し，その位置はベクトル $\boldsymbol{w}_i$，$\boldsymbol{w}'_j$ を，その大きさは重み $1/n$，$1/n'$ を表す．次にふたつの分布の間の EMD をユークリッド空間で測る：

$$d_{\mathrm{E}}(\boldsymbol{w}_i, \boldsymbol{w}'_j) := \|\boldsymbol{w}_i - \boldsymbol{w}'_j\| \quad (7)$$

$$\mathrm{WMD}(s, s') := \mathrm{EMD}(\boldsymbol{\mu}_s, \boldsymbol{\mu}_{s'}; d_{\mathrm{E}}). \quad (8)$$

この定式化は直観的で，経験的な効果も確認されているが，単語の重要度と意味を混ぜて取り扱ってしまっているという問題をはらむ．これについて次のセクションで詳しく述べる．

4 Word Rotator's Distance

以下, 各単語ベクトル $\boldsymbol{w}_i$ のノルムと方向ベクトルを $\lambda_i \in \mathbb{R}$, $\boldsymbol{u}_i \in \mathbb{R}^D$ で表す：$\lambda_i := \|\boldsymbol{w}_i\|$，　$\boldsymbol{u}_i := \boldsymbol{w}_i/\|\boldsymbol{w}_i\|$.

4.1 単語ベクトルの長さと単語の重要度

単語ベクトルのノルム (長さ) には単語の重要度がエンコードされていると考えられる．以下にその理由を述べる．

単語ベクトルを足し合わせて作った文ベクトルが文の意味をよく表現することは**加法構成性**としてよく知られており，STS でもこれらの間のコサイン類似度が良好な性能を示す．

$$\boldsymbol{s}_{\text{ADD}} = \sum_{w_i \in s} \boldsymbol{w}_i, \quad \boldsymbol{s}'_{\text{ADD}} = \sum_{w'_j \in s'} \boldsymbol{w}'_j. \quad (9)$$

一見すると (9) は個々の単語ベクトルを均等に扱っているように見える．しかし実際には単語ベクトルのノルムの分散は大きく [2]，結果として，ノルムの大きな (長い) 単語ベクトルが文ベクトルの支配的な要素となり，逆にノルムの小さな単語ベクトルはほとんど無視されることになる．文の "良い表現" を作ることのできる加法構成がこうしたノルムによる重み付けを暗黙的に実行している以上，単語ベクトルのノルムには各単語の相対的な重要度がエンコードされていることが期待される．実際，ノルムによる重み付けの効果を排除した加法構成で文ベクトルを構成すると，

$$\boldsymbol{s}_{\text{ADD W/O NORM}} = \sum_{w_i \in s} \boldsymbol{w}_i/\lambda_i = \sum_{w_i \in s} \boldsymbol{u}_i \quad (10)$$

文類似度評価タスクにおける性能が大幅に低下した (表1)．

	ADD (9)	ADD W/O NORM (10)
GloVe	**54.16**	46.25
word2vec	**72.43**	63.20
fastText	**70.40**	56.31

表1: 文ベクトル間のコサイン類似度と人手評価の相関係数 (Pearson's $r \times 100$). 行毎に最良の数値を**太字**で示す．評価データは STS-B (dev) [5].

	GloVe			word2vec		
	COS	L2	DOT	COS	L2	DOT
MEN	**80.49**	73.36	**80.79**	**78.20**	62.31	74.46
MTurk287	**69.18**	60.87	**69.50**	**68.37**	49.43	66.60
MC30	**78.81**	75.22	76.77	**78.87**	69.88	76.57
RW	**47.28**	40.37	45.64	**53.39**	31.70	48.66
RG65	76.90	70.75	**77.79**	**76.17**	71.30	72.58
SimLex999	**40.84**	35.16	38.99	**44.19**	32.24	43.28
WS353-SIM	**79.57**	69.03	**79.54**	**77.39**	55.82	74.89

表2: 人手評価との順位相関係数 (Spearman's $\rho \times 100$). 利用している単語ベクトル・評価データ毎に最良の数値 ± 0.5 を**太字**で示す．

また先行研究でも，固有名詞 (文の意味に強く寄与すると考えられる) のノルムが，機能語 (寄与しないと考えられる) のノルムに比べて大きい傾向があることが確かめられている [15]．

4.2 単語ベクトルの向きと単語の意味

ふたつの単語の意味の違いを計算するために，本稿では単語ベクトル同士のなす角 (コサイン) を用いる

$$\cos(\boldsymbol{w}, \boldsymbol{w}') = \langle \boldsymbol{w}, \boldsymbol{w}' \rangle / (\lambda\lambda') = \langle \boldsymbol{u}, \boldsymbol{u}' \rangle. \quad (11)$$

(11) でノルムは無視されることに注意されたい．すなわち本稿では，単語ベクトルのノルム (長さ) には重要度がエンコードされており (§4.1)，これを差し引いた方向ベクトル (向き) に単語の意味がエンコードされていると考える．

単語ベクトルの類似度の計算手法としてコサイン類似度はもっとも自然で標準的であるが，STS の先行研究ではその他の尺度もしばしば利用される．たとえば WMD ではユークリッド距離 [9]，注意機構を利用した文類似度尺度では内積 [21] と，ノルムを加味した尺度が利用される．単語類似度の評価セットでこれらの尺度を比較すると，コサイン (すなわちノルムを無視する場合) がもっとも高性能であることが確認できた (表2)．

4.3 WMD の問題点

以上より，WMD にはふたつの問題があることがわかる．

■ 単語の重み付け． EMD が確率質量を介して各点 (各単語ベクトル) の重みを考慮でき (2)，各単語ベクトルの重要度はノルムにエンコードされているにも関わらず (§4.1)，WMD はこれを無視して各単語ベクトルを一様に重み付けしている (6)．

■ 意味的類似度の計算． EMD が距離尺度を介して点 (単語ベクトル) 間の非類似度を考慮でき (3)，単語ベクトルの意味的類似度はコサインで測ることができるにも関わらず (§4.2)，WMD では重要度と意味が混ざったユークリッド距離を用いている (8)．たとえば "動物" ベクトルと "ペルシャ猫" ベクトルのように意味は近いがその具体性や重要度が大きく異なる単語ベクトル対について，類似度が低く見積もられてしまう[*5]．

4.4 Word Rotator's Distance

以上の考察を踏まえ，単語ベクトルのノルムと方向ベクトルを分けて活用する新しい文類似度尺度を提案する (図3)．直感

行列 $\boldsymbol{T} \in \mathcal{C}(\boldsymbol{\mu}, \boldsymbol{\mu}')$ は $\boldsymbol{\mu}$ から $\boldsymbol{\mu}'$ へ荷物を過不足なく移し替える輸送計画行列で，$\boldsymbol{T}_{ij}$ は位置 $\boldsymbol{x}_i$ から位置 $\boldsymbol{x}'_j$ に移す荷物の量を表す．$d(\boldsymbol{x}_i, \boldsymbol{x}'_j)$ は単位量を二点間で輸送するコストなので，$\mathrm{EMD}(\boldsymbol{\mu}, \boldsymbol{\mu}'; d)$ はふたつの分布 $\boldsymbol{\mu}$，$\boldsymbol{\mu}'$ を距離空間 $(\mathbb{R}^D, d)$ 上で移し替える最小コストとなる．

[*3] 本稿の実験では POT ライブラリを用いた：https://github.com/rflamary/POT/.

[*4] たとえば，$\boldsymbol{x}_i$ と $\boldsymbol{x}'_j$ が近ければ ($d(\boldsymbol{x}_i, \boldsymbol{x}'_j)$ が小さければ)，これらの 2 点は対応付けられる (輸送量 $\boldsymbol{T}_{ij}$ は大きくなる)．

[*5] 内積を用いた場合も，ノルムと方向ベクトルを混ぜて用いる弊害が起きる．たとえば一方のベクトルのノルムが大きければ (具体性や重要度が高ければ) なす角が大きくとも (意味的類似度が低くても) 高い意味的類似度を持つと見積もられてしまう．

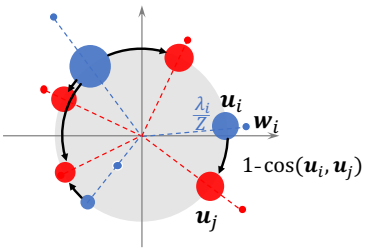

図3: Word Rotator's Distance.

的には，提案法はまず各文を「単位超球面上に射影された単語ベクトル集合 (離散分布)」と見なし，超球面上で EMD を計算する．超球面上で方向ベクトルのアラインメントは回転にあたるため，提案法を **Word Rotator's Distance (WRD)** と名付ける．

形式的には，まず各文 s, s' をノルムで重み付けされた方向ベクトルの集合，すなわち単位超球面上の離散分布とみなす：

$$\boldsymbol{\nu}_s := \left\{(\boldsymbol{u}_i,\ \lambda_i/Z)\right\}_{i=1}^{n} \quad \left(Z := \sum_i \lambda_i\right) \tag{12}$$

$$\boldsymbol{\nu}_{s'} := \left\{(\boldsymbol{u}'_j,\ \lambda_j/Z')\right\}_{j=1}^{n'} \quad \left(Z' := \sum_j \lambda'_j\right), \tag{13}$$

図3では各単語を円で表し，その位置はベクトル $\boldsymbol{u}_i$, $\boldsymbol{u}'_j$ を，その大きさは重み (ノルム) λ_i/Z, λ'_j/Z' を表す．次に，コサイン距離を輸送コストとして EMD を計算する

$$d_{\cos}(\boldsymbol{u}_i, \boldsymbol{u}'_j) := 1 - \cos(\boldsymbol{u}_i, \boldsymbol{u}'_j) \tag{14}$$

$$\mathrm{WRD}(s, s') := \mathrm{EMD}(\boldsymbol{\nu}_s, \boldsymbol{\nu}_{s'}; d_{\cos}). \tag{15}$$

これにより WRD は，WMD と異なり，EMD と単語ベクトルとの間に次の適切な対応付けがなされる．

- 確率質量 (各点の**重み**) ↔ ノルム (単語の**重要度**)
- 距離尺度 (各点間の**距離**) ↔ なす角 (単語間の**非類似度**)

5 文ベクトル推定が誘導する単語ベクトル変換

5.1 潜在文ベクトルの推定

潜在文ベクトルから単語ベクトルたちが生成される生成モデル [3] を考えれば，逆に，与えられた単語ベクトルの集合から潜在文ベクトルを推定できる．これらの間のコサイン類似度は STS において精度面で大きな成功を収めている [4, 7].

5.2 潜在文ベクトルの推定が誘導する単語ベクトル変換

これまで提案されてきた文ベクトルの推定手法およびその組み合わせは，次の形式で一般的に書き表すことができる．

$$\mathrm{Encode}(s) = f_3\left(\frac{1}{n}\sum_{w\in s} \alpha_2(w) f_1(\boldsymbol{w})\right). \tag{16}$$

ここで，関数 f_1, α_2, f_3 はそれぞれ次の役割を担う．

- f_1: 単語ベクトルの "ノイズ" の除去 [7, 11]
- α_2: 単語ベクトルを拡大・縮小 [4, 7]
- f_3: 文ベクトルの "ノイズ" の除去 [4]

過去に提案された f_3 がいずれも線形変換であることに注意すると，(16) は次のように書き換えることができる：

$$\mathrm{Encode}(s) = \frac{1}{n}\sum_{w\in s} f_{\mathrm{VC}}(\boldsymbol{w}) \tag{17}$$

$$f_{\mathrm{VC}}(\boldsymbol{w}) := f_3\left(\alpha_2(w)\cdot f_1(\boldsymbol{w})\right). \tag{18}$$

すなわち，精度面で成功を収めている文ベクトルの推定手法は，まず各単語ベクトル $\boldsymbol{w}$ を f_{VC} で変換し，その後これを足し合わせる (加法構成) 手法だと見なすことができる．

	ADD (9)	ADD W/O NORM (10)
GloVe	**54.16**	46.25
GloVe + A	**68.30**	59.62
GloVe + AW	**76.68**	59.62
GloVe + **VC**(AWR)	**79.13**	63.60

表3: 人手評価との相関係数 (Pearson's $r \times 100$)．行毎に最良の数値を**太字**で示す．評価データは STS-B (dev) [5].

	GloVe	GloVe + A	GloVe + **VC**(AWR)
MEN	80.49	**82.43**	**82.26**
MTurk287	69.18	**72.77**	69.32
MC30	78.81	77.99	**80.67**
RW	47.28	**54.75**	**54.34**
RG65	**76.90**	75.18	**76.89**
SimLex999	40.84	46.74	**49.83**
WS353-SIM	79.57	**80.97**	79.32

表4: 人手評価との順位相関係数 (Spearman's $\rho \times 100$)．評価データ毎に最良の数値 ± 0.5 を**太字**で示す．

本稿では単語ベクトル変換 f_{VC} の具体例として以下を用いる．

- f_1: all-but-the-top (**A**) [11], conceptor negation (**C**) [10], 次元毎正規化 (**N**) [7]
- α_2: SIF weighting (**W**) [4]
- f_3: common component removal (**R**) [4]

また，たとえば **A**, **W**, **R** に誘導されるベクトル変換を **VC**(AWR) のように記す．パラメータは全て引用先論文のまま用いた．

5.3 変換された単語ベクトルの長さと向き

文ベクトルの推定法もある種の加法構成であり (17)，しかも単純な加法構成よりも高精度に STS が解けることが分かっている．したがって，$f_{\mathrm{VC}}(\boldsymbol{w})$ のノルムには元の $\boldsymbol{w}$ のノルムよりも単語の重要度がよりよくエンコードされていることが期待される．実際，変換され単語ベクトル $\{f_{\mathrm{VC}}(\boldsymbol{w})\}$ を用いて加法構成と，ノルムによる重み付けの効果を排除した加法構成 (10) をを比較すると，ノルムによる重み付けが STS の推定に強く貢献していることが確認できた (表3).

また，単語ベクトルの向きには単語の意味がエンコードされていたが，この傾向も f_{VC} によって強まることが単語類似度タスクによって確かめられた (表4).

5.4 単語ベクトル変換と WRD の統合

変換された単語ベクトル $f_{\mathrm{VC}}(\boldsymbol{w})$ は提案手法 WRD の入力としてそのまま利用できる．WRD は単語ベクトルのノルムと方向ベクトルを直接的に活用する方法であり，ノルムと方向ベクトルを改善する f_{VC} によって提案手法の性能がさらに向上すると期待できる．

6 実験

6.1 アブレーション実験

はじめに，ふたつの提案手法それぞれの効果について確かめるアブレーション実験をおこなった (表5). WMD から WRD に変更することで文類似度尺度としての性能が一貫して向上し，ノルムと方向ベクトルを分けて用いる効果 (あるいは混ぜて用いることのデメリット) が確認できた．

また，文ベクトル推定法が誘導する単語ベクトル変換 f_{VC} を利用することで文類似度尺度としての性能が一貫して向上した．ノルムと方向ベクトルの改善が WRD の性能の改善に直結

ストップワード除去	WMD	**WRD**	WMD ✓	**WRD** ✓
GloVe	62.56	**64.66**	**71.34**	**71.13**
GloVe + A	65.74	**68.83**	**75.19**	**75.19**
GloVe + AW	63.34	**77.21**	74.41	**76.44**
GloVe + **VC**(AWR)	61.42	**79.20**	72.81	**78.60**

表5: 人手評価との相関係数 (Pearson's $r \times 100$). ストップワードの除去をする場合としない場合で，行毎に最良の数値 ± 0.5 を**太字**で示す．全体で最良の結果に**下線**を付す．評価データは STS-B (dev) [5].

	STS'15	STS-B	Twitter
GloVe – 文ベクトル			
GloVe†	56.08	41.45	29.56
GloVe + WR† [4]	64.66	65.61	40.24
GloVe + UP [7]	76.1	71.5	–
GloVe – 単語ベクトルアラインメント			
WMD GloVe† [9]	67.29	61.67	41.15
DynaMax GloVe [23]	70.9	–	–
BERTScore GloVe† [21]	71.38	57.44	52.37
WRD GloVe + **VC**(CWR) **(提案法)**	76.25	75.21	48.75
WRD GloVe + **VC**(NWR) **(提案法)**	**76.74**	74.62	54.10
fastText – 単語ベクトルアラインメント			
WRD fastText + **VC**(CWR) **(提案法)**	**76.34**	**76.29**	51.97
WRD fastText + **VC**(NWR) **(提案法)**	**76.83**	**76.02**	**55.04**
その他 – 文ベクトル			
Sent2Vec [12]	–	75.5*	–
Skip-Thought‡ [8]	46	–	–
ELMo (All layers, 5.5B)‡ [14]	68	–	–

表6: 人手評価との相関係数 (Pearson's $r \times 100$). 評価データ (列) 毎に最良の結果 ± 0.5 を**太字**で示し，最良の結果に**下線** を付す．(†) は我々による再現実験，(‡) は Perone et al. [13] からの引用，(∗) は STS Wiki[*8]からの引用，ほか記号のない既存手法の数値は元論文からの引用．

することが分かる．

6.2 ベンチマークテスト

次に，STS システムの性能評価のためのベンチマークデータを用いて，提案手法と既存のベースライン手法とを比較した．

■ データセット． 3 種類の STS 評価用データを用いた．

- STS'15 (SemEval 2015, STS task; [8, 14, 23] との比較用)
- STS-B (SemEval 2012–2016, STS task の抜粋・集約版) [5]
- Twitter (SemEval 2015, Paraphrase & STS task) [20]

■ ベースライン手法． 単語ベクトルアラインメントに基づく 3 種類の既存手法と比較した．

- WMD [9][*6]
- ファジィ集合に基づく手法 [23]
- 注意機構に基づく手法 [21]

さらに，以下の**文ベクトル**を構成する手法と比較した．

- 文ベクトル推定に基づく手法 [4, 7]
- 「足し合わせに最適な」文ベクトルを推定する手法 [12]
- 深層文エンコーダに基づく手法 [8]
- 文脈付き単語ベクトルの加法構成 [14][*7]

■ 結果． 実験結果は表6の通り．まず，単語ベクトル間のアラインメントを計算する手法群の中で，提案手法がもっとも良好な精度を示した．各種既存手法は，単語ベクトルにエンコードされた重要度と意味を "混ぜて" 取り扱っており，このことが提案手法の優位性に繋がったと考えられる．

[*6] 効果の確認されているストップワード除去も実施した．

[*7] 我々の知る限り，今のところ BERT やその亜種は教師なし STS において良好な結果を示せていない．[21] の BERT/RoBERTa へ適用が最も有望であるが，ベンチマークタスクでの結果が大きく劣るため表から省いた．

さらに提案法は，最も盛んに取り組まれている STS-B を含む全てのベンチマークデータで過去の最良の教師なし STS 手法を上回った．とりわけ文ベクトル推定に基づく手法 [7] を上回った点に着目したい．我々が用いた単語ベクトル変換はあくまで文ベクトル推定法から誘導されたものであり，この点で既存手法に対する優位性はない．差が生じているのは単語ベクトルのアラインメントの有無であり，STS 計算におけるアラインメントの重要性を強く示唆するものと考えられる．

7 おわりに

単語ベクトルのアラインメントという直感的な指針に基づく STS 計算は，精度面では文ベクトル推定手法に後れをとっていた．本稿ではこの理由が「単語の意味と重要度を混ぜて利用しているから」だと仮説を立て，単語ベクトルとノルムと方向ベクトルを別々に扱う手法を提案した (WRD)．さらに，文ベクトル推定手法が単語ベクトル変換手法と捉えられることを示し，WMD との統合方法を提案した．ふたつの提案手法により，複数のベンチマークタスクで過去最良の精度を達成した．

EMD に基づく手法は，句の単位や係り受けといった構造情報を容易に組み込むことができる [1]．今回は EMD 自体には工夫を加えず "素の" 状態で既存手法に対する優位性を確認したが，今後，様々な構造情報を考慮しその効果を確認したい．

謝辞． 本研究は JST CREST (課題番号 JPMJCR1513) の支援を受けたものです．

参考文献

[1] D. Alvarez-Melis et al. "Structured Optimal Transport". In: *AISTATS*. Vol. 84. 2018, pp. 1771–1780.
[2] N. Arefyev et al. "How much does a word weigh? Weighting word embeddings for word sense induction". In: *arXiv:1805.09209* (2018).
[3] S. Arora et al. "A Latent Variable Model Approach to PMI-based Word Embeddings". In: *TACL* 4 (2016), pp. 385–399.
[4] S. Arora et al. "A Simple but Tough-to-Beat Baseline for Sentence Embeddings". In: *ICLR*. 2017.
[5] D. Cer et al. "SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Crosslingual Focused Evaluation". In: *SemEval*. 2017, pp. 1–14.
[6] E. Clark et al. "Sentence Mover's Similarity: Automatic Evaluation for Multi-Sentence Texts". In: *ACL*. 2019, pp. 2748–2760.
[7] K. Ethayarajh. "Unsupervised Random Walk Sentence Embeddings: A Strong but Simple Baseline". In: *Rep4NLP*. 2018, pp. 91–100.
[8] R. Kiros et al. "Skip-Thought Vectors". In: *NIPS*. 2015, pp. 3294–3302.
[9] M. J. Kusner et al. "From Word Embeddings To Document Distances". In: *ICML*. Vol. 37. 2015, pp. 957–966.
[10] T. Liu et al. "Unsupervised Post-processing of Word Vectors via Conceptor Negation". In: *AAAI*. 2019, pp. 6778–6785.
[11] J. Mu and P. Viswanath. "All-but-the-Top: Simple and Effective Postprocessing for Word Representations". In: *ICLR*. 2018.
[12] M. Pagliardini et al. "Unsupervised Learning of Sentence Embeddings Using Compositional n-Gram Features". In: *NAACL*. 2018, pp. 528–540.
[13] C. S. Perone et al. "Evaluation of sentence embeddings in downstream and linguistic probing tasks". In: *arXiv:1806.06259* (2018).
[14] M. E. Peters et al. "Deep Contextualized Word Representations". In: *NAACL*. 2018, pp. 2227–2237.
[15] A. M. J. Schakel and B. J. Wilson. "Measuring Word Significance using Distributed Representations of Words". In: *arXiv:1508.02297* (2015).
[16] J. Solomon et al. "Entropic Metric Alignment for Correspondence Problems". In: *ACM Transactions on Graphics* 35.4 (2016), 72:1–72:13.
[17] M. A. Sultan et al. "DLS$@$CU: Sentence Similarity from Word Alignment". In: *SemEval*. 2014, pp. 241–246.
[18] C. Villani. *Optimal Transport*. 1st ed. Vol. 338. Springer Berlin Heidelberg, 2009.
[19] J. Wieting et al. "Beyond BLEU:Training Neural Machine Translation with Semantic Similarity". In: *ACL*. 2019, pp. 4344–4355.
[20] W. Xu et al. "SemEval-2015 Task 1: Paraphrase and Semantic Similarity in Twitter (PIT)". In: *SemEval*. 2015, pp. 1–11.
[21] T. Zhang et al. "BERTScore: Evaluating Text Generation with BERT". In: *arXiv:1904.09675* (2019).
[22] W. Zhao et al. "MoverScore: Text Generation Evaluating with Contextualized Embeddings and Earth Mover Distance". In: *EMNLP*. 2019, pp. 563–578.
[23] V. Zhelezniak et al. "Don't Settle for Average, Go for the Max: Fuzzy Sets and Max-Pooled Word Vectors". In: *ICLR*. 2019.