語順と共起を考慮したニューラル言語モデルによる英文穴埋め

Sentence Completion by Neural Language Models using Word Order and Co-occurrences

森 洸樹　三輪 誠　佐々木 裕
Koki Mori　Makoto Miwa　Yutaka Sasaki
豊田工業大学
Toyota Technological Institute
{sd11092, makoto-miwa, yutaka.sasaki}@toyota-ti.ac.jp

1 序論

英文穴埋め問題は言語モデルを評価する方法として，広く利用されている．英文穴埋め問題の代表的なタスクとして，MSR Sentence Completion Challenge (MSR) [6] という Holmes の文章を対象にした穴埋め問題が提供されている．このタスクでは，N-gram モデルよりもニューラル言語モデルを用いた inverse vector Log-bilinear language model (ivLBL) [4] という手法が高い正答率を出している．また，TOEIC の穴埋め問題についても N-gram モデルを用いた研究が行われている [7] が，ニューラル言語モデルを用いた研究は行われていない．TOEIC の問題では，文法や熟語などの語順が影響するような問題が多く用意されている．これに対して，N-gram モデルは N-gram の頻度データを用いた手法なので，語順に関する知識が必要とする問題は得意とする．しかし，この手法では似た文脈で使われる「今日」と「明日」や「雨」と「雪」などの単語の表記による違いを区別してしまい，1 単語違うだけで全く違う文と判断してしまう．一方，ニューラル言語モデルの一種である vector Log-bilinear language model (vLBL) [4] では，単語をベクトルで表現し，文脈内の単語の共起に関する学習を行う．単語をベクトルで表現することで表記による違いを区別せずに，似た文脈で使われる単語のベクトルには似た特徴を持たせることができる．しかし，語順を無視して学習行うため，その知識を必要とする問題を解くことはできない．そこで本研究では，TOEIC の穴埋め問題をニューラル言語モデルを用いて評価する．さらに語順を考慮したニューラル言語モデルを構築し，そのモデルと vLBL を足すことで語順・共起両方を考慮したモデルを構築することで，正答率の向上を目指す．

2 vLBL

ニューラル言語モデルの一種である vector Log-bilinear language model (vLBL) [4] では，各単語をベクトルで表現し，文中の 1 つの単語を対象として，その単語のベクトルと文脈内の各単語のベクトルの類似度が高くなるように学習を行う．この学習により単語の共起に関する知識を得ることができる．その結果，似た文脈で現れる単語はそれぞれ似た特徴のベクトルを持ち，単語の意味的・統語的な特徴をベクトルで表現することができる．例えば「今日/は/雨/が/降る」と「明日/は/雪/が/降る」のように,「今日」と「明日」や,「雨」と「雪」など似た文脈で使われる単語は似た特徴のベクトルを持つことになる．対象の単語 w_t のベクトルを $\mathbf{w}_t$，w_t から相対的に i 語離れた位置にある単語 w_{t+i} のベクトル $\mathbf{w}_{t+i}$，w_t に対応するバイアスを b，文脈 c を w_t の前後 n 単語として，vLBL における c と w_t の類似度を示すスコア関数を $s_{vLBL}(c, w_t)$ を次式で表す．

$$s_{vLBL}(c, w_t) = \mathbf{c} \cdot \mathbf{w}_t + b_t \tag{1}$$

$$\mathbf{c} = \frac{1}{2n} \sum_{-n \le i \le n, i \neq 0} \mathbf{w}_{t+i} \tag{2}$$

(1) 式で定義したスコア関数を効率よく学習するために，文脈外の単語 $w_{t'}$ を生成し，c とは相違しているとして，逆向きの学習を同時に行う．(これを Negative Sampling(NEG) [3] と呼ぶ)．学習に用いる対象の単語 w_t に対する目的関数 g_t を次式で表す．

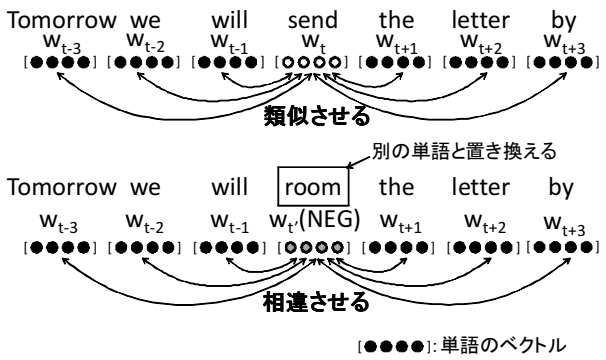

図 1: vLBL による学習

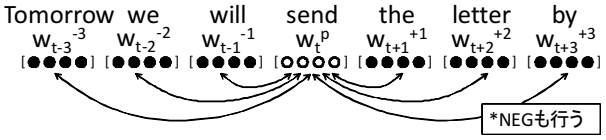

図 2: vLBL(c) による学習

$$g_t = log\sigma(s(c, w_t)) + \sum_{w'_t \sim P_n}^{k} log(1 - \sigma(s(c, w_{t'}))) \quad (3)$$

ただし $\sigma(x)$ はロジスティック関数，P_n は単語の頻度分布，k は NEG で生成する単語数，第 2 項は P_n から k 単語をサンプルすることを示す．式 (1) を式 (3) で用いることで vLBL の目的関数を得る．得られる目的関数を最大化するように各ベクトルを学習することで，対象の単語と文脈内の各単語のベクトルは類似し，文脈外の単語と文脈内の各単語のベクトルは相違するようにする．vLBL による学習の様子を図 1 に示す．

目的関数 g_t を最大化する方法としては，適切に学習率を減衰させ，高速かつ高精度の学習を行う AdaGrad を用いる．AdaGrad では，モデルのパラメータごとの学習率を自動的に調整する．モデルのパラメータベクトル $\boldsymbol{\theta}$ における AdaGrad の更新式を次式で表す．

$$\boldsymbol{\theta}_j = \boldsymbol{\theta}_{j-1} + \frac{\alpha}{\sqrt{\epsilon + \sum_{i=1}^{j} \frac{\|\nabla \mathbf{g}_i\|^2}{(\mathbf{g} \text{ の次元数})}}} \nabla \mathbf{g}_j \quad (4)$$

ただし，j は更新回数，α は初期値，$\nabla \mathbf{g}$ は $\mathbf{g}$ の勾配，ϵ はパラメータの発散を防ぐための小さな正の値である．なお，本来の AdaGrad とは異なり，計算コスト低減のためにパラメータベクトルごとに同じ学習率を用いる [4]．また定期的に学習率を初期化する [5]．

[4] では，w_{t+i} のベクトル $\mathbf{w}_{t+i}$ に w_{t+i} との相対的な位置に依存したベクトル $\mathbf{w}^i_{t+i}$ を重みとして付加した vLBL with position dependent context weights (vLBL(D)) という手法も提案されている．

3 提案手法

vLBL では，文脈内の各単語が対象の単語 w_t に対して相対的にどの位置にあるかを考慮せず，文脈内で共起しやすい単語の類似度が高くなるように学習を行っている．しかし，この学習で得られる単語のベクトルは，語順に関する知識を十分に学習できていない．そこで，文脈内の各単語が w_t との相対的な位置に注目したモデルを構築し，さらにそのモデルと vLBL を組み合わせたモデルを提案する．

3.1 vLBL with position-dependent contexts

vLBL with position-dependent contexts (vLBL(c)) では，文脈内の各単語に w_t との相対的な位置に依存したベクトル $\mathbf{w}^i_{t+i}$ と w_t のベクトル $\mathbf{w}_t$ の類似度が高くなるように学習して，語順に関する知識を得る．例えば “He must have missed the train.” という文で，“missed” を対象として，動詞の活用を正しく判断するには，直前の “have” に注目しなければならない．しかし，vLBL では “must” と “have” を区別をせずに学習を行うため，2 つ前 “must” の影響を受ける．そこで vLBL(c) では直前の “must” と 2 つ前の “must” のように，その単語が w_t からどの位置にあるかを区別して，それぞれに別々のベクトルを割り当てて学習を行う．vLBL(c) における文脈 c と w_t の類似度を示すスコア関数を $s_{vLBL(c)}(c, w_t)$ を次式で表す．

$$s_{vLBL(c)}(c, w_t) = \mathbf{c}^p \cdot \mathbf{w}_t + b \quad (5)$$

$$\mathbf{c}^p = \frac{1}{2n} \sum_{-n \le i \le n, i \ne 0} \mathbf{w}^i_{t+i} \quad (6)$$

式 (5) を式 (3) で用いることで，vLBL(c) の目的関数を得る．vLBL(c) による学習の様子を図 2 に示す．

3.2 vLBL+vLBL(c)

vLBL(c) によって得られる単語のベクトルは語順に関する知識には強くなるが，共起に関する知識には弱くなる．そこで，共起に関する学習を行っている vLBL と組み合わせた手法として vLBL+vLBL(c) モデルを提案する．vLBL+vLBL(c) では，文脈内の各単語のベクトル $\mathbf{w}_{t+i}$ と，対象の単語 w_t との相対的な位置に依存のベクトル $\mathbf{w}^i_{t+i}$ が，それぞれ w_t へ作用するベクトルとして $\mathbf{w}_t$ と $\mathbf{w}^p_t$ を設ける．vLBL+vLBL(c)

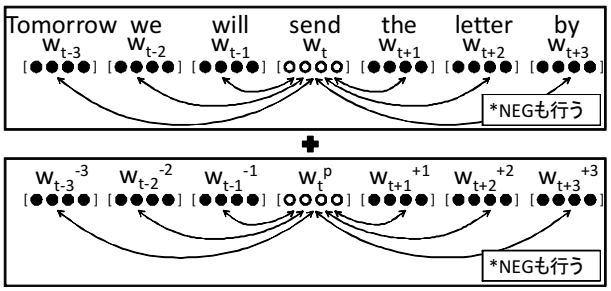

図 3: vLBL+vLBL(c) による学習

における文脈 c と w_t の類似度を示すスコア関数を $s_{vLBL+vLBL(c)}(c, w_t)$ を次式で表す．

$$s_{vLBL+vLBL(c)}(c, w_t) = \mathbf{c} \cdot \mathbf{w}_t + \mathbf{c}^p \cdot \mathbf{w}_t^p + b \quad (7)$$

式 (7) を式 (3) で用いることで，vLBL+vLBL(c) の目的関数を得る．vLBL+vLBL(c) による学習の様子を図 3 に示す．

4 実験設定

2 つの評価データを用いて vLBL，vLBL(c)，vLBL+vLBL(c) を評価した．vLBL(D) は事前実験により，良い結果が得られなかったため，本実験では評価の対象から外した．

一つ目の評価データは，解くのに文法や熟語などの語順が影響するような問題が多く用意されている TOEIC の穴埋め問題である．学習データとして，Wikipedia のデータ (1,735,020,309 語) を全て小文字化して用いた．ただしこのデータで学習するには長い時間が掛かるため，開発用にこの一部のデータ (4,316,154 語) も用意した．また，数回しか出現していない単語については十分な学習が行えないため，学習時間を短縮するために，Wikipedia の全てのデータでは最低 50 回，Wikipedia の一部のデータでは最低 5 回出てきている単語に絞って学習を行った．評価データには，Webanhvan [1] から取得した TOEIC の練習問題を用いた．問題は 4 択の英文穴埋め問題となっており，全 1228 問を開発データ 613 問，テストデータ 615 問に分けて用いた．

二つ目の評価データは，言語モデルの評価によく用いられている，MSR Sentence Completion Challenge(MSR) [6] を用いた手法である．MSR は Holmes の小説をもとに作成した穴埋め問題である．学習データとして，プロジェクト・グーテンベルクにより公開されている 19 世紀の小説 522 冊を対象としている (1,775,155 語)．なお，このデータについても最低 5 回出てきている単語に絞り学習を行った．評価には Holmes の小説をもとに人手で作成された 5 択の穴埋め問題 520 問を用いた．

学習時間の短縮のために，計算は並列化し，ベクトルの更新は 100 回分をまとめてに行うミニバッチを用いた．学習の回数を 20 回，NEG で生成する単語数 k を 5 個，文脈の領域 n を対象の単語の前後 5 単語とした．また，NEG で生成する単語は全単語リスト P_n から抽出するが，抽出する際よく出現する単語 (a, the など) が必然的に選ばれやすくなるため，すべての単語の出現頻度を 3/4 乗してから，抽出する単語を選ぶようにしている [3]．

5 結果と考察

5.1 TOEIC を用いた評価実験

ベクトルの次元数を 100 として，開発データで評価した結果を表 1 に示す．表 1 より，提案手法の 2 つのモデルが vLBL よりも高い正答率を出している．単語のベクトルと位置に依存したベクトルをそれぞれ独立させて学習することで，語順に関する知識を正しく学習できていることが分かる．また，vLBL と vLBL+vLBL(c)，vLBL(c) と vLBL+vLBL(c) をそれぞれ比較すると，vLBL+vLBL(c) の正答率が高くなっている．このことから，vLBL より得た共起に関する知識と vLBL(c) より得た語順に関する知識がいずれも重要であることが分かる．また，用いた学習データを比較すると，Wikipedia の全てのデータを扱った方が正答率は高くなっている．学習に用いるデータの大きさは重要であることが分かる．

次に提案手法の 2 つのモデルを用いて，ベクトルの次元数を 1,000，学習データを Wikipedia の全てのデータとして学習を行い，開発データとテストデータで評価した結果を表 2 に示す．比較対象として，N-gram モデルを用いた [7] のシステムの正答率を示す．[7] のシステムでは，空欄の前後 4 単語と各選択肢を

表 1: データサイズごとの開発データ上での正答率

モデル	次元数	正答率 (%)	
		一部	全て
vLBL	100	40.7	51.2
vLBL(c)	100	45.1	64.4
vLBL+vLBL(c)	100	**46.3**	**68.1**

合わせたキーワードを，Google n-gram で検索し，頻度が大きいものを解答としている．表 2 より， [7] のシステムも高い正答率を出すことができた．N-gram モデルでは表現できていない，共起に関する知識を正しく表現できていることが分かる．

5.2 MSR を用いた評価実験

ベクトルの次元数を 100 と 600 に設定して実験を行った結果を表 3 に示す．比較対象として，Good-Turing smoothing を用いている N-gram モデル [6]，対象の単語から周りの単語を予測する vLBL とは逆のモデルである inverse vector Log-bilinear language model (ivLBL) [4]，Skip-Gram モデル [2] (文脈の領域 n を対象の単語の前後 10 単語とした結果) の結果も示す．表 3 より vLBL(c) と N-gram モデルの結果は低くなっていることから，MSR の問題は語順に関する知識では，良い結果が出ないといえる．しかし，vLBL+vLBL(c) では vLBL とくらべて，ほぼ同じ正答率を出すことができたので，vLBL より得た共起に関する知識を十分に利用できているといえる．

表 2: 評価データごとの正答率

モデル	次元数	正答率 (%)	
		開発	テスト
N-gram モデル [7]	—	73.1	
vLBL(c)	1000	74.7	76.6
vLBL+vLBL(c)	1000	**75.2**	**77.7**

表 3: MSR における正答率

	次元数	正答率 (%)
vLBL	100	49.4
vLBL(c)	100	43.0
vLBL+vLBL(c)	100	50.6
vLBL	600	53.8
vLBL(c)	600	51.3
vLBL+vLBL(c)	600	54.0
N-gram モデル [6]	—	39
ivLBL [4]	100	51.0
ivLBL [4]	600	**55.5**
Skip-Gram [2]	640	48.0

6 結論

穴埋め問題の正答率向上を目的に，語順･共起両方を考慮したニューラル言語モデルを構築した．結果として vLBL+vLBL(c) は，MSR で最も高い正答率を出している ivLBL と同等の正答率を出し，TOEIC で 77.7% と高い正答率を出すことができた．また，これらの結果より，語順･共起両方を考慮している vLBL+vLBL(c) は，単語の特徴を十分に表現できていると考えられる．今後は品詞の情報や，文全体の情報を組み込んで，さらに精確に単語の特徴を表現する予定である．

参考文献

[1] Webanhvan, http://www.webanhvan.com.

[2] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *ICLR2013*. 2013.

[3] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *NIPS 26*, pp. 3111–3119. 2013.

[4] Andriy Mnih and Koray Kavukcuoglu. Learning word embeddings efficiently with noise-contrastive estimation. In *NIPS 26*, pp. 2265–2273. 2013.

[5] Richard Socher, Andrej Karpathy, V. Quoc Le, D. Christopher Manning, and Y. Andrew Ng. Grounded compositional semantics for finding and describing images with sentences. *TACL*, Vol. 2, pp. 207–218, 2014.

[6] Geoffrey Zweig and Christopher J.C. Burges. The microsoft research sentence completion challenge. Technical Report MSR-TR-2011-129, December 2011.

[7] 白石裕次郎, 佐々木裕. 英語穴埋め問題の自動解法. In *NLP2014*, pp. 101–104. 2014.