長さの異なる n-gram 同士の関係を考慮した n-gram 埋め込み

山崎 智弘
東芝研究開発センター アナリティクスＡＩラボラトリー
tomohiro2.yamasaki@toshiba.co.jp

1 はじめに

日本の総人口は 2030 年には 1 億 1913 万人、2060 年には 9284 万人にまで減少すると見込まれている [13]。少子化のため若年層が老年層より少ないこともあり、生産年齢人口は総人口より急速に減少する可能性が高い。特に製造業の現場では、ベテランが減少することでさまざまなノウハウが失われるという視点からも深刻な問題として捉えられている。

片や製造業では、日々の業務で起こったトラブルを文書として記録しておき、同じトラブルを二度と起こさないようにする取組みがある。そのため経験が浅い若手であっても、本来なら蓄えられた文書に基づいてノウハウを活用できるはずである。しかし経験が浅いとトラブルに関連する適切な語句が思いつかず過去の事例を見つけられないことがよく起こる。また表面的には異なるが少し視点を変えるとほとんど同じ内容の事例を見逃してしまうなど、現状ではノウハウを十分に活用できているとは言いがたい。

そこで我々は類似事例の検索や事例からの知識抽出を支援するための要素技術として、キーワードや文書間の意味的類似度を求める手法の研究開発を進めている。意味的に類似した表現 (類義表現) が把握できれば誰でも仕様書に類似した事例が見つけられるようになり、設計時のリスク低減につながると考えられる。

画像認識で大きな成功を収めたニューラルネットワーク (NN) は自然言語処理の多くのタスクでも成功を収めつつある。中でもこの分野においてベースとなっている要素技術は Word Embedding(単語埋め込み) と呼ばれる手法 [7, 6, 8] である。対象語と共起する周辺語を文脈と見なし、意味的に類似した単語同士が低次元ベクトル空間において近くに配置されるように、NN を用いてラベルなしコーパスから単語のベクトル表現を学習する。

埋め込みによるベクトルには加法性があるほか、文字 [2] や形態素 [11] の埋め込みも併用することでコーパスにない単語も扱えるようになったこともあり、これまで定量的に捉えることが難しかった単語の意味がうまくモデル化されているものと考えられている。

単語での成功を受け、より大きな単位である文や段落、あるいは文書の埋め込み手法を確立したいと考えるのは自然な流れである。しかし多くの研究者がさまざまな手法を提案しているにもかかわらず未だデファクトが確立されたとは言いがたい。そこで我々は文よりは小さい単位である n-gram を取り上げ、新たな埋め込み手法を提案する。具体的には長さの異なる n-gram 同士を適切にモデル化することを目的とし、NN の層を n-gram の長さごとに分け、それらを n-gram 同士の関係に基づいて接続したネットワークでベクトル表現を学習する。

評価実験としては n-gram 同士のベクトル類似度が意味的類似度を反映できているか分析する。その結果、提案手法による n-gram 埋め込みは長さの異なる n-gram 同士の意味と文法パターンが捉えられていることを示す。

2 関連研究

文の埋め込み手法としては [4] や [3] が知られている。[4] は文中の単語を予測する文ベクトルを学習する。モデルとして分かりやすいが、語順が無視されるほか学習の反復数を増やすと短い文のベクトルが類似しやすいという報告 [1] がある。一方 [3] は、対象文の単語列をベクトルに変換し、前後の文の単語列をベクトルから予測するように学習する。文内の語順だけでなく前後の文の関係も考慮しているため、文の構造がうまくモデル化されていると考えられている。しかし文中の単語の埋め込みベクトルを平均するだけの手法 (平均法) もタスクによってはよい性能が得られるのでよく使われている。逆に言えば平均法を置き換えるほどのデファクトは確立されていない。

n-gram の埋め込み手法としては [12] が提案されている。従来の単語埋め込みでは単語同士の共起情報しか利用しないが、n-gram 同士の共起情報も利用するように拡張することで単語の埋め込みベクトルが改善されることを示した。しかし n-gram の埋め込みベクトルとしては、後述するように自明な包含関係にあるものばかり類似しやすいという問題がある。本論文では長さの異なる n-gram が公平に扱われるように改変を加えることで、意味的類似度を反映した埋め込みベクトルが学習できることを示す。

表 1: 既存手法による類似表現 (数値は cos 類似度)

東芝	、東芝 (.901)	の東芝 (.852)
	東芝の (.848)	は東芝 (.840)
	に東芝 (.826)	・東芝 (.793)
	。東芝 (.780)	東芝が (.778)
であった	であっ(.991)	あった (.929)
	であった。(.912)	あった。(.848)
	であったが (.828)	であったが、(.792)
	あったが (.758)	であったため (.752)

単語についてよりよいベクトル表現を学習しようとする研究としては文脈を双方向 LSTM で扱う手法 [9] がある。同じ単語であっても文脈によって異なる埋め込みベクトルが学習できるほか、既存手法に適用するだけで多くのタスクにおいて大幅に性能向上することを示した。とはいえ [9] によるベクトルは Skip-Gram with Negative Sampling(SGNS)[6] や GloVe[8] によるベクトルと組み合わせて利用されており、従来の埋め込み手法を完全に置き換えるものではない。提案手法への適用は今後の課題とし、本論文では扱わない。

3 提案手法

前述の [12] は n-gram の埋め込みベクトルも学習できることを示した。しかし考慮されているのが 3-gram までなので我々は 5-gram まで広げて実験したところ、表 1 のとおり自明な包含関係にある n-gram のベクトルばかり類似しやすいことがわかった。

単純に [6] を拡張しているので、範囲が重なる n-gram 同士の関係、長さごとの出現頻度の差や窓幅に含まれる個数の差などが影響しているものと考えられる。例えば窓幅 2 で 3-gram まで考慮するものとしよう。小説「雪国」の冒頭文において図 1 に示すように ‘長いトンネル’ という 2-gram を対象としたとき、窓幅 2 なので ‘国境 · · · 抜ける’ の 6 単語が周辺の範囲となる。この範囲に含まれる周辺 1-gram は ‘国境’ から ‘抜ける’ までの 6 個、周辺 2-gram は ‘国境の’ から ‘を抜ける’ までの 4 個 (対象そのものは含まない)、周辺 3-gram は ‘国境の長い’ から ‘トンネルを抜ける’ までの 4 個となり、窓幅が一定のときは短い n-gram の方が含まれる個数が多い。また ‘長いトンネル’ に対する ‘長い’ と ‘トンネル’ のように包含関係にある n-gram は必ず文脈として扱われるため、長さごとの扱いが公平でない。

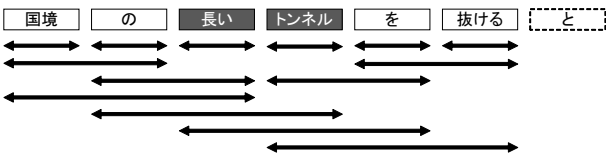

図 1: 窓幅 2 で 3-gram まで考慮するときの ‘長いトンネル’ に対する周辺 n-gram

そこで我々は n-gram の長さごとに層を分け、それぞれの関係に基づいて接続したネットワークで学習する手法を提案する。図 2 に概念図を示す。[12] はすべての n-gram を対象に softmax(あるいは negative sampling) を適用していたが、本手法は分かれている層ごとに適用するようにしてあり、長さごとの出現頻度の差が吸収される。また例えば対象が ‘長いトンネル’ のときそれに隣接する ‘を’ を周辺 1-gram として学習する場合、‘を’ のベクトルを近づけるように正例にするだけでなくそれらを結合した ‘長いトンネルを’ を周辺 3-gram として同時に正例にしたり、あるいは対象に含まれる周辺 1-gram である ‘長い’ と ‘トンネル’ はベクトルを遠ざけるように負例にしたりすることで、長さの異なるものが同時に出現することを反映した学習も可能になる。

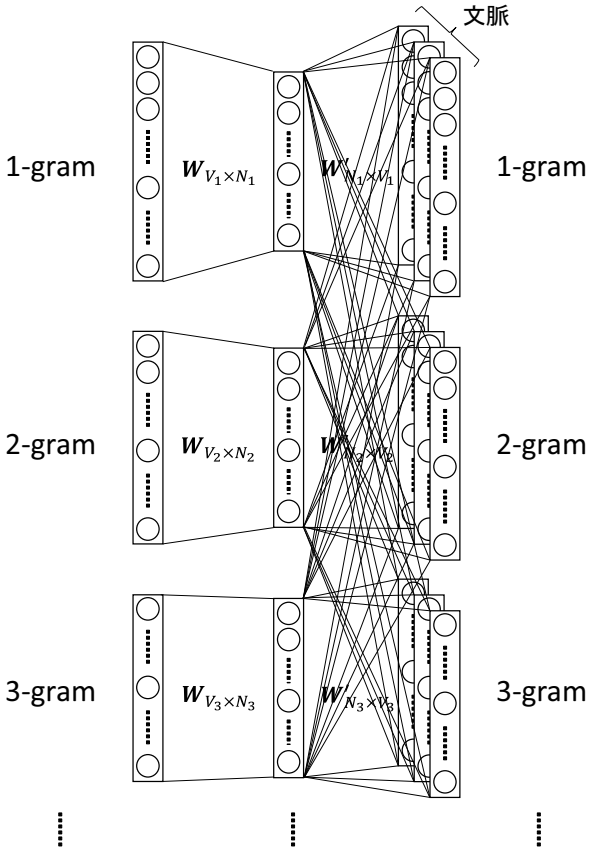

図 2: 提案手法のネットワークの概念図

n-gram 同士の関係としてどこまで考慮するかによってどこまで接続するかが変わるため、具体的なネットワークの構造はさまざまなものが考えられる。しかしネットワークの重み行列の更新は、分かれている層ごとに softmax(あるいは negative sampling) を適用することに注意して一般的な誤差逆伝播法を用いればよい。最終的に図 2 の左側の重み行列 $(W_{V_1\times N_1}, W_{V_2\times N_2}, W_{V_3\times N_3}, \ldots)$ がそれぞれの n-gram の埋め込みベクトルとして得られる。

4 実験と評価

本論文では日本語および英語の Wikipedia のダンプデータから抽出した本文をコーパスとし、5-gram までの埋め込みベクトルを学習した。コーパスの規模と得られた n-gram の個数を表 2 に示す。

表 2: 学習に用いたコーパスの規模

	日本語	英語
バイト数	2.68×10^9	12.5×10^9
行数	8.84×10^6	42.0×10^6
語数	5.18×10^8	23.6×10^8
n-gram 数	1.36×10^7	5.78×10^7

いずれもダンプデータのバージョンは 20181201 である。日本語コーパスは独自の形態素解析エンジンで単語に分割した。英語コーパスは基本的にスペースで分割したが、すべて小文字化したうえで単語の前後にくっついている記号類も単語として分割した。提案手法はベクトルの次元数が n-gram の長さごとに異なっていてもよいが、今回は一律に 300 とした。また多くの埋め込み手法と同じく SGNS を踏襲したハイパーパラメタを持っているので、低頻度のものを無視する閾値を 10、高頻度のものを間引く度合いを 10^{-5}、学習の反復数を 5、negative sample 数を 5 とした。

n-gram 同士の関係としてはさまざまなものが考えられるが、長さごとの扱いを公平にするための改変をなるべく少なくするという観点から、本論文では

A. 対象に隣接するもののみを周辺と見なし、それらを正例にする
B. 対象に隣接するものと包含されるもののみを周辺と見なし、隣接するものを正例、包含されるものを負例にする

の 2 通りを取り上げ、n-gram 同士のベクトル類似度が意味的類似度を反映できているか [12] で窓幅 5、オーバラップありとしたベースラインと比較を行なった。具体的には打消、形容修飾、過去受身、継続、名詞連続などの文法パターンごとに対象 n-gram を 330 件選び、それらとベクトルが類似した n-gram 上位 10 件ずつを抽出し、対象が出現した文において対象と入れ替えても意味が通じる文になるかで妥当性を判定した。

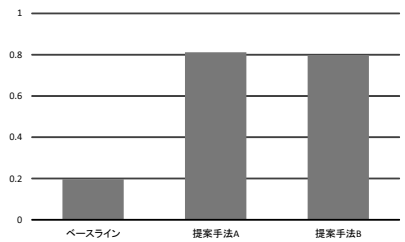

図 3: 類似表現として妥当だったものの割合

図 3 に示すように、妥当だったものの割合はベースライン.197 に対して提案手法 A は.812、提案手法 B は.794 であった。類義表現を抽出するという観点からは妥当なものの割合がベースラインより大きく改善しており、意味と文法パターンがうまく捉えられていることがわかる。

次に、日本語でパターンごとに選んだ対象 n-gram とそれの類似表現の実例を表 3 に示す。3 章の冒頭で述べたようにベースラインでは、例えば ‘できない’ に対する ‘ができない’ や ‘できない。’ のように前後に助詞や句読点がくっついただけのもの、‘でき’ のように対象に含まれるものなど、自明な包含関係にあるものばかりになってしまっていることがわかる。しかし提案手法では ‘出来ない’ や ‘できていない’ のような対象と同じ打消パターンのもの、‘不可能な’ のようなパターンとしては異なるが意味が類似しているものも得られていることがわかる。

2 つの提案手法は対象に含まれるものを負例にするかどうかしか違わないので結果に明確な優劣があるわけではないが、包含関係にあるものを積極的に遠ざけるように学習する提案手法 B の方が、類義表現の抽出という観点からは対象に含まれない単語からなる幅広い表現が得られやすい傾向がある。

続いて、学習された埋め込みベクトルが加法性を持つかどうか単語類推タスク [1] の代表的なデータセットである Google analogy test set で評価した。単語 X の埋め込みベクトルが vec(X) のとき vec(b) − vec(a) + vec(c) との類似度が最も高い vec(d) が正解となる割合を算出したところ、ベースライン.718 に対して提案手法 A は.441、提案手法 B は.082 であった。データセットの正解が単語 (=1-gram) しか用意されていないことを踏まえてもベースラインより大きく悪化しており、本タスクはうまく扱えていないことがわかる。すなわち本手法による埋め込みベクトルは、類似度という局所的な関係は捉えられているが加法性という大域的な関係は捉えられていないことを示している。

5 おわりに

本論文では新たな n-gram 埋め込みの手法を提案した。長さごとに分けた層を n-gram 同士の関係に基づいて接続することでネットワークを構成するものである。いくつかのパターンごとに n-gram を選び、それらとベクトルが類似した n-gram を抽出して分析したところ、提案手法では自明な包含関係にあるものばかりではなく意味的に類似したものが得られることが確認された。

[1] 単語 a、b、c が与えられたときに a に対する b の関係が c に対する d の関係と等しくなる単語 d を求めるタスク

表 3: 手法ごとに得られた類似表現 (数値は cos 類似度)

対象		ベースライン	提案手法 A	提案手法 B
打消	できない	ができない (.937)	出来ない (.952)	出来ない (.946)
		ことができない (.894)	することができない (.859)	不可能な (.796)
		できない。(.873)	できなかった (.826)	されない (.788)
	必要でない	が必要でない (.826)	必要ではない (.882)	充分でない (.819)
		は必要でない (.797)	必要無い (.869)	面倒である (.807)
		に必要でない (.748)	必要とされない (.863)	不要になる (.799)
形容修飾	激しい雷雨	、激しい雷雨 (.841)	強い雨 (.871)	激しい嵐 (.793)
		雷雨 (.821)	濃い霧 (.861)	しばしば洪水 (.787)
		激しい雷雨が (.807)	暴風雪 (.859)	濃い霧 (.767)
	高い品質	高い品質を (.834)	高い生産性 (.839)	高い耐久性 (.792)
		高い品質の (.761)	優れた品質 (.837)	高い生産性 (.791)
		、高い品質 (.754)	安定した品質 (.811)	高い安全性 (.780)
過去受身	書かれた	書かれ (.962)	記された (.793)	記された (.822)
		書か (.947)	書かれている (.783)	作成された (.756)
		に書かれた (.934)	書か (.774)	記述された (.752)
	引っ張られた	に引っ張られた (.806)	押し込まれた (.817)	蹴られた (.774)
		引っ張られ (.796)	叩きつけられた (.817)	掴まれた (.759)
		を引っ張られた (.768)	蹴られた (.813)	当ててしまう (.756)
継続	達成している	を達成している (.968)	達成した (.849)	達成した (.776)
		達成している。(.961)	達成していた (.836)	達成した。(.766)
		達成して (.926)	成し遂げている (.809)	成し遂げていた (.752)
	出版している	出版している。(.952)	刊行している (.880)	刊行している (.869)
		を出版している (.946)	出版していた (.845)	上梓している (.827)
		出版して (.936)	出版した (.818)	出版されている (.799)
名詞連続	ＩＴ技術	ＩＴ技術を (.799)	ウェブ技術 (.833)	コンピュータ技術 (.803)
		、ＩＴ技術 (.747)	人工知能技術 (.831)	ＣＧ技術 (.799)
		のＩＴ技術 (.739)	ネットワーク技術 (.828)	航空機技術 (.798)
	パターン認識	、パターン認識 (.822)	コンピュータビジョン (.870)	フォトリソグラフィ (.744)
		パターン認識、(.760)	計算流体力学 (.859)	知的エージェント (.742)
		、パターン認識、(.749)	ソフトウエア工学 (.853)	特殊高所 (.740)

しかし今回実験した範囲では単語類推タスクに基づく評価でベースラインを上回ることはできず、埋め込みによるベクトルから加法性が失われていることが確認された。どのようなネットワークを構成すれば加法性が保たれるのかは今後の課題である。

なお本論文では n-gram のみを取り上げたが、ネットワークの構成の仕方からわかるように、単語のパターンであれば特に連続している必要はない。そのため単語が連続していないイディオムなどもモデル化できる可能性がある。それらについても検討を進める。

参考文献

[1] Qingyao Ai, Liu Yang, Jiafeng Guo, and W. Bruce Croft. Analysis of the Paragraph Vector Model for Information Retrieval. In *ICTIR*, 2016.

[2] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching Word Vectors with Subword Information. *TACL*, 2017.

[3] Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-Thought Vectors. In *NIPS*, 2015.

[4] Quoc V. Le and Tomas Mikolov. Distributed Representations of Sentences and Documents. In *ICML*, 2014.

[5] Oren Melamud, Jacob Goldberger, and Ido Dagan. context2vec: Learning Generic Context Embedding with Bidirectional LSTM. In *CoNLL*, 2016.

[6] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. *CoRR*, 2013.

[7] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic Regularities in Continuous Space Word Representations. In *HLT-NAACL*, 2013.

[8] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014.

[9] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *NAACL*, 2018.

[10] Adam Poliak, Pushpendre Rastogi, M. Patrick Martin, and Benjamin Van Durme. Efficient, compositional, order-sensitive n-gram embeddings. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pp. 503–508. Association for Computational Linguistics, 2017.

[11] Minh thang Luong, Richard Socher, and Christopher D. Manning. Better Word Representations with Recursive Neural Networks for Morphology. In *CoNLL*, 2013.

[12] Zhe Zhao, Tao Liu, Shen Li, Bofang Li, and Xiaoyong Du. Ngram2vec: Learning Improved Word Representations from Ngram Co-occurrence Statistics. In *EMNLP*, 2017.

[13] 総務省. 平成 30 年版 情報通信白書.