Recurrent Entity Networks における文脈表現への係り受け関係の活用

鈴木 諒　　鶴岡 慶雅
東京大学 工学部電子情報工学科
{r-suzu, tsuruoka}@logos.t.u-tokyo.ac.jp

1 はじめに

ニューラルネットワークの発展に伴い，近年 Memory Networks [1] のように自然言語理解のタスクにおいて構文情報を用いずに高い精度を達成するモデルが登場している．そのようなモデルでは，文や句のベクトル表現として Bag of Words (BoW) が用いられることも多い．

Henaff らが提案する Recurrent Entity Networks [2] では，ある単語を中心とする幅 5 のウィンドウの BoW を，その単語に関する文脈表現として利用している．つまり，

“Early next morning the King , Queen , ladies-in-waiting , and officers came out to see where the Princess had been .”

という文の “Queen” に関する文脈を
{“King”, “,”, “Queen”, “,”, “ladies-in-waiting”}
という BoW で表現するということである．しかしこれは，“Queen” に関する文脈として有用な単語を捉えられているとは言い難い．一方，文全体を含むような BoW を用いた場合，長い文や複雑な文では個々の単語の意味が薄くなってしまい，やはり文脈を上手く表現することはできない．

このように，BoW にはその単純さや計算コストの低さといった利点も存在する反面，長い文や複雑な文に対する表現力は乏しいという問題がある．そのため，計算量を抑えつつも単純な BoW よりも表現力の高い手法を考案することができれば，応用範囲も広く非常に有意義であると考えられる．

そこで本研究では，ある単語に関する文脈を表現する上で有用な単語は，単純な単語間距離よりもむしろ係り受け木における距離が近くなるという仮定に基づき，Reccurent Entity Networks における文脈表現に係り受け関係を用いる方法を提案する．その結果，係り受け関係を利用する際の課題を発見することができたのと同時に，並列語の捕捉能力など，従来手法であるウィンドウの BoW には無い有用性を見出すこともできた．

2 対象とするタスク

本稿で扱うタスクは，複数の文からなる文書とそれに続くクエリが与えられた時に，そのクエリに対して正しく応答するタスクである．bAbI tasks [3] などの質問応答タスクでは文書の内容に関する質問文がクエリとなり，Children’s Book Test [4] などの空所補充問題では一部の単語を伏せた文がクエリとなる．どちらにおいても，正しい解答を導くためにはそれまでの文脈を正しく捉えた上で推論を行う必要がある．

3 Recurrent Entity Networks

Recurrent Entity Networks (EntNet) は，長期メモリを利用して文脈に応じた推論を行うニューラルネットワークのモデルである．同様の推論モデルである Memory Networks の各メモリは各入力文にそれぞれ対応しているが，EntNet の各メモリは文中に登場する各エンティティに対応付けられているという特徴がある．これにより，bAbI tasks において最高水準の性能を示したほか，Children’s Book Test においてもシングルパスのモデルとしては非常に高い性能を示した．

図 1 は EntNet の概略図である．以下ではこのモデルの基本的な仕組みを説明する．

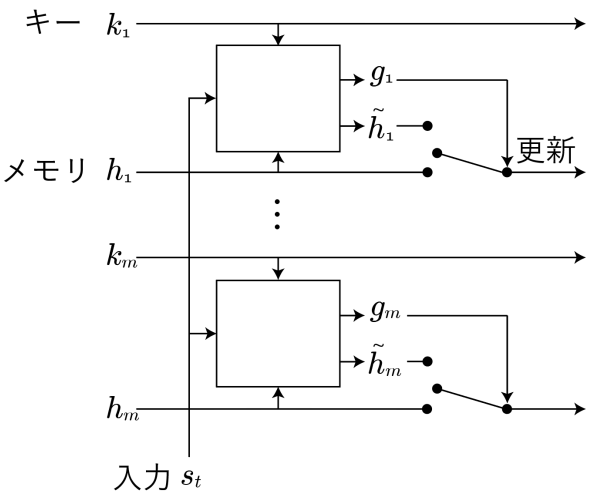

図 1: Recurrent Entity Networks の概略図

3.1 入力

EntNet において，入力文は単語の位置を考慮した Bag of Words (BoW) で表現される．つまり，t 番目の入力文のベクトル表現 $s_t \in \mathbb{R}^{d_e \times 1}$ は，

$$\boldsymbol{s_t} = \sum_i f_i \odot \boldsymbol{e_i}, \tag{1}$$

のように各単語の埋め込み $\boldsymbol{e_1}, ..., \boldsymbol{e_k} \in \mathbb{R}^{d_e \times 1}$ の重み付き和として計算する．ここで f_i は単語の位置による重みであり，他のパラメータと同様に学習する．$\odot$ は要素積である．

また，Children's Book Test においては文ごとの BoW ではなく，解答の候補となる語（あるいは空所）の周りの Window Memory [4] を入力として使用する．例えば入力文の i 番目の単語 w_i が解答候補として与えられた語であった場合，その単語を中心とする幅 b のウィンドウ $\{w_{(i-(b-1)/2)}, ..., w_i, ..., w_{(i+(b-1)/2)}\}$ の BoW となる．実際には $b = 5$ の Window Memory が使用された．また単一の文中に複数の候補語が含まれる場合もあるため，一般に入力の数は入力文の数よりも多くなる．

3.2 動的メモリ

EntNet の隠れ層はメモリと呼ばれ，入力文を受け取る度に全てのメモリを更新する構造になっている．j 番目のメモリ $\boldsymbol{h_j} \in \mathbb{R}^{d_e \times 1}$ の更新式は以下の通りである．

$$\boldsymbol{h_j} \leftarrow \boldsymbol{h_j} + g_j \odot \tilde{\boldsymbol{h_j}}. \tag{2}$$

ここで g_j は更新の度合いを表すスカラー値であり，次のように計算する．

$$g_j \leftarrow \sigma(\boldsymbol{s_t}^T \boldsymbol{h_j} + \boldsymbol{s_t}^T \boldsymbol{k_j}). \tag{3}$$

シグモイド中の第一項はメモリの内容と入力文との関連度を意味しており，既に記憶されている事項と関連する入力文を受け取った場合にそのメモリを更新できるようにする役割を持つ．また，第二項の $\boldsymbol{k_j} \in \mathbb{R}^{d_e \times 1}$ は各メモリに一対一に対応するキーベクトルである．ここでは文書中に登場するエンティティ（単語）の埋め込みベクトルをそのままキーベクトルとし，メモリを各エンティティに関連付けている．その結果，関連付けられた単語が入力文に含まれる場合にそのメモリを更新できるようになる．

一方，$\tilde{\boldsymbol{h_j}} \in \mathbb{R}^{d_e \times 1}$ は実際の更新量の候補値であり，

$$\tilde{\boldsymbol{h_j}} \leftarrow \phi(\boldsymbol{U h_j} + \boldsymbol{V k_j} + \boldsymbol{W s_t}), \tag{4}$$

と算出する．$\boldsymbol{U}$，$\boldsymbol{V}$，$\boldsymbol{W} \in \mathbb{R}^{d_e \times d_e}$ はそれぞれ学習するパラメータである．ϕ としては identity や parametric ReLU [5] を使用する．

最後に正規化を行い，古い情報の重みを相対的に小さくしていくことで，新しい情報をより重視するようにする．

$$\boldsymbol{h_j} \leftarrow \frac{\boldsymbol{h_j}}{\|\boldsymbol{h_j}\|}. \tag{5}$$

以上の操作をまとめたものが図 1 である．全ての入力文に対して同じ操作を再帰的に行うことで文書の内容をメモリに記憶していく．

3.3 出力

全ての入力文についてメモリの更新を行った後，

$$p_j = \mathrm{Softmax}(\boldsymbol{q}^{\mathrm{T}} \boldsymbol{h_j}), \tag{6}$$

$$\boldsymbol{u} = \sum_j p_j \boldsymbol{h_j}, \tag{7}$$

$$\boldsymbol{y} = \boldsymbol{R}\phi(\boldsymbol{q} + \boldsymbol{H u}), \tag{8}$$

のように最終的なメモリの状態とクエリから回答と算出する．$\boldsymbol{q} \in \mathbb{R}^{d_e \times 1}$ はクエリを表すベクトルであり，入力文と同様の方法で算出する．$\boldsymbol{R} \in \mathbb{R}^{d_v \times d_e}$ と $\boldsymbol{H} \in \mathbb{R}^{d_e \times 1}$ は学習するパラメータである．$\boldsymbol{y} \in \mathbb{R}^{d_v \times 1}$ の要素うち最も大きいものに対応する単語を回答として出力する．

ただし各メモリに関連付けられている単語の中から正解を選ぶ場合には，式 (7) と式 (8) については省略が可能である．その場合，$\boldsymbol{p}$ は各単語の正解となる確率の分布と見ることができる．

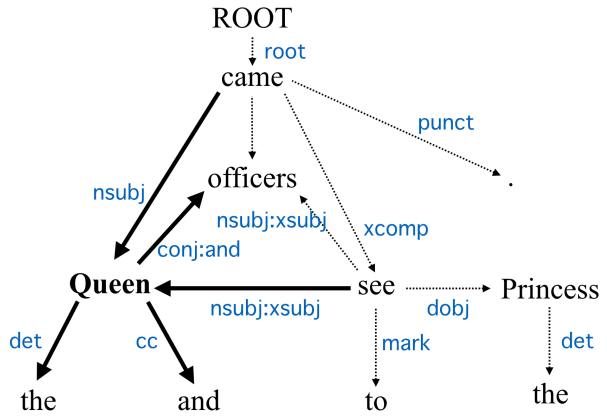

図 2: 係り受け木の例（Standard Stanford Dependencies 形式）

4 提案手法

EntNet の入力の計算式 (1) を次式に置き換える．

$$\boldsymbol{s_t} = \sum_i \boldsymbol{g}^{\mathrm{T}} \boldsymbol{l_i} \odot \boldsymbol{e_i}. \tag{9}$$

$\boldsymbol{e_1}, ..., \boldsymbol{e_N} \in \mathbb{R}^{d_e \times 1}$ には中心語および係り受け木において中心語と隣接する単語の埋め込みが含まれる．$\boldsymbol{l_1}, ..., \boldsymbol{l_N} \in \mathbb{R}^{d_l \times 1}$ はそれぞれの単語と中心語との間の関係を表すラベルを表す one-hot ベクトルであり，$\boldsymbol{g} \in \mathbb{R}^{d_l \times 1}$ はラベルに依る重みを得るためのパラメータである．なお，ラベルは中心語に対する係り受け関係の向きによって区別している．また，中心語自体のラベルについては別途定義した．

例えば “The Queen and officers came to see the Princess.” という文の係り受け木は図 2 のようになる．この場合，“Queen” に隣接する単語 {“the”, “Queen”, “and”, “officers”, “came”, “see”} とそれに対応するラベル {“O:det”, “_TARGET”, “O:cc”, “O:conj:and”, “I:nsubj”, “I:nsubj:xsubj”} を用いて入力を計算することになる．なお，ラベルの接頭の “O:” や “I:” は中心語から出る向きの関係か入る向きの関係かを区別するために付与したものである．また，“_TARGET” は中心語そのものを表すラベルとして定義したものである．

5 実験

5.1 実験設定

学習と評価には Children’s Book Test[1] の NE セット（10 個の Named Entity から空所を埋める語を選ぶタスク）を用いた．20 個の文からなる文書とそれに続く穴開き文を 1 ストーリーとして，学習用データセットには 108,719 個，評価用データセットには 2,500 個のストーリーが含まれる．係り受け解析には Stanford CoreNLP [6] 3.7.0 [2] を用い，出力は Standard Stanford Dependencies [7] の形式とした．また，比較として幅 5 の Window Memory を用いた手法についても実験を行った．実装は github で公開されているソースコード[3]を元に行った．

単語の埋め込みは 100 次元とし，ドロップアウト（率 0.5）を適用した．メモリの数は 10 個とし，各メモリとキーは対応する解答候補語の埋め込みで初期化した．また，入力時のラベルに依る重み $\boldsymbol{g}$ は式 (3)(4)(6) の入力でそれぞれ別に用意し，それぞれ 1 で初期化した．それ以外のパラメータについては平均 0 標準偏差 0.1 のガウス分布で初期化した．パラメータの最適化は標準的な確率的勾配降下法によって行い，学習率は初期値 0.1 から 20 エポック毎に半減させた．

表 1: 実験結果

	モデル	正解率
	Kneser-Ney LM [Heafield ら]	0.439
	Contextual LSTMs [Hill ら]	0.436
	Window Memory [Henaff ら]	0.616
EntNet	Window Memory（実装）	0.567
	提案手法	0.502

5.2 結果と考察

実験の結果を表 1 に示す．なお，本実験で用いたプログラムでは Henaff らの実験の値を再現することができなかったため，参考値として掲載している．また，Heafield らによる Kneser-Ney Language Model [8] や Hill らによる Contextual LSTMs [4] は，EntNet と同様に文書を一度しか読まないシングルパスのモデルである．

提案手法の係り受け関係を用いた方法は，EntNet 以外のシングルパスのモデルよりは良い精度を達成できたが，従来の Window Memory を用いた EntNet には及ばない結果となった．なお，提案手法において 1 回の入力あたりに含まれる平均単語数は 5.18 であ

[1] http://www.thespermwhale.com/jaseweston/babi/CBTest.tgz

[2] http://stanfordnlp.github.io/CoreNLP/

[3] https://github.com/jimfleming/recurrent-entity-networks

表 2: 学習語のラベルの重み

ラベル	重み
O:compound	13.569
O:amod	5.288
O:nsubj	5.024
⋮	⋮
I:ccomp	0.720
O:conj:and	-0.526
I:conj:and	-1.098

り，幅 5 の Window Memory と比較することは妥当であると言える．

本実験では係り受け関係を表すラベルは 648 種類存在したが，そのうち学習後の重みが初期値 1.0 ± 0.05 に収まるものは，式 (3)82.6% (4)91.3% (6)95.5% にも及んだ．その多くは “nmod:close_to” のような細かい分類が付与されたラベルである．これは Standard Stanford Dependencies 形式の特徴であるが，細かく分類されているが故に個々の出現率が低く，学習が進まなかったものと考えられる．また，評価用データセットにしか存在しない未知ラベルが性能に悪影響を及ぼしている可能性も考えられる．

一方，式 (6) の入力で使用したラベルの重みのうち，学習後の上位と下位それぞれ 3 個を抜き出したものが表 2 である．

“compound” は複合語を成す 2 語を結ぶラベルであり，空所と複合語の関係にある単語を重視するように学習できていることが分かる．ただし複合語の単語同士は一般に隣接しており，Window Memory を用いた場合でも重視されることに変わりはない．

一方，“conj:and” は and により並列関係にある単語同士を結ぶラベルである．この重みが負値になっているのは，空所と並列関係にある単語がその空所に入ることは無いという事実によるものと思われる．Window Memory を用いた場合には並列関係にある単語がウィンドウの外に存在する場合もあるが，提案手法であればこの関係を確実に捉えることができ，解答候補から排除できるという点では有用性が認められる．

6 おわりに

本研究では，係り受け木において隣接する単語とその間の関係を用いて，ある単語に関する文脈を表現する方法を提案したが，精度は従来手法に及ばなかった．主語と目的語の関係など，距離 2 以上の位置にある単語が重要である場合も多い．これらを考慮するためのモデルの拡張を今後の課題とする．また，Standard Stanford Dependencies の細かいラベルは有用であるものもあるが，出現率の低さにより学習が進んでいないラベルが多く，これに対しては基本のラベルも併用するなどして対処していく必要がある．

参考文献

[1] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory Networks. In *Proceedings of the ICLR*, 2015.

[2] Mikael Henaff, Jason Weston, Arthur Szlam, Antoine Bordes, and Yann LeCun. Tracking the world state with recurrent entity networks. In *Proceedings of the ICLR*, 2017.

[3] Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. Towards ai-complete question answering: A set of prerequisite toy tasks. In *Proceedings of the ICLR*, 2016.

[4] Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. The goldilocks principle: Reading children’s books with explicit memory representations. In *Proceedings of the ICLR*, 2016.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *arXiv:1502.01852*, 2015.

[6] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *ACL System Demonstrations*, 2014.

[7] Marie-Catherine De Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D. Manning. Universal stanford dependencies: a cross-linguistic typology. In *Proceedings of the ICLR*, 2014.

[8] Kenneth Heafield, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn. Scalable modified kneser-ney language model estimation. In *Proceedings of the ACL*, 2013.