DRS 意味解析における出現位置を利用した語彙数削減

黒澤 友哉　谷中 瞳
東京大学
{kurosawa-tomoya, hyanaka}@is.s.u-tokyo.ac.jp

概要

高度な意味解析タスクの一つとして、文を入力として談話表示構造 (Discourse Representation Structure, DRS) を出力する DRS 意味解析がある。DRS 意味解析器のうち、高精度を達成した van Noord et al. (2020) [1] のエンコーダ・デコーダモデルは 2 つの独立したエンコーダを用いている。一方は文中の単語とその意味タグを交互に並べて入力とし、他方は文を文字単位で分割し入力とすることで、文の意味情報を学習しようと試みている。しかしこの手法では、文中のトークンが DRS トークンに含まれるため、モデルの語彙数が非常に大きくなり、デコーダの計算量が増大し予測精度の低下につながる可能性がある。本研究では、DRS に含まれるトークンのうち解析対象の文にも含まれるトークンを、文における出現位置に置き換えるという手法を提案する。これにより、モデルの出力語彙数が 90%以上減少し、エンコーダ・デコーダモデルがより正確に DRS トークンを予測できる。実験により、既存手法と比較して精度の上昇を示したと同時に、学習時間の短縮とメモリ量の低下も達成した。

1　はじめに

意味解析は自然言語文を意味関係を構造化した意味表現に変換する、自然言語処理において最も基礎的なタスクの一つである。意味表現の形式は数多く存在するが、本研究では理論言語学の分析に基づく高度な意味表現形式 [2] である談話表示構造 (DRS) に着目する。図 1 は *I was born on the 31st of May in 1940* を表現する DRS である。談話指示子の集合（例: **t1**, **e1** など）と条件の集合（例: **t1** ≺”now” など）の組を「箱」と呼び、これらを用いて表現した DRS を **箱形式 DRS** と呼ぶ。

図 2 は図 1 と等価であるが、表現方法が異なる。図 2 のような DRS を **句形式 DRS** と呼ぶ。句形式 DRS は箱形式 DRS に対し、エンコーダ・デコーダモデルで扱う表現方法として適しており、[1, 3] などで用いられている。

t1 e1	**[b1]**
t1 ≺ “now”	time.n.08(**t1**)
Time(**e1**, **t1**)	DayOfMonth(**t1**, “31”)
Patient(**e1**, “speaker”)	MonthOfYear(**t1**, “05”)
bear.v.02(**e1**)	YearOfCentury(**t1**, “1940”)

図 1　*I was born on the 31st of May in 1940* を表す箱形式 DRS。

```
1: b1 TPR t1 "now"
2: b1 Time e1 t1
3: b1 REF t1
4: b1 REF e1
5: b1 Patient e1 "speaker"
6: b1 bear "v.02" e1
7: b1 DOM t1 "31"
8: b1 time "n.08" t1
9: b1 MOY t1 "05"
10: b1 YOC t1 "1940"
```

図 2　図 1 と等価な句形式 DRS。DOM, MOY, YOC はそれぞれ DayOfMonth, MonthOfYear, YearOfCentury の略である。

特に [1] が提案した意味解析器は、複数のエンコーダを用いたエンコーダ・デコーダモデルで、入力のトークン列のみならず、その文字の系列や様々な言語情報を与えて学習させている。彼らは様々な組み合わせで実験を行なっているが、そのうち、2 つのエンコーダを用い、一方には入力のトークン列とその意味タグを、他方には文字の系列を与える手法が最も良い精度を達成した。

しかし、句形式 DRS の一部のトークンは、モデルが予測すべき語彙の数を大きく増加させる要因となりうる。一つ目は図 2 の 6 行目のように、文に出現する単語が第 2 トークンに挿入されることがある。二つ目は図 2 の 7, 9, 10 行目のように、文に出現する単語の表す値が第 4 トークンに挿入されることがある。いずれについても任意の単語や値が挿入されうるため、エンコーダ・デコーダモデルはより膨大な語彙を有する必要があり、結果として予測精度が落ちている可能性がある。さらに、学習時に観測しなかった未知語も扱わなければならない。

表 1 PMB 3.0.0 と 4.0.0 の英語におけるドキュメント数の一覧。

	Gold			Silver	Bronze
	Train	Dev	Test	Train	Train
3.0.0	6,620	885	898	97,598	146,371
4.0.0	7,668	1,169	1,048	127,303	151,493

そこで本研究では、上述したトークンを可能な限り減少させることにより、エンコーダ・デコーダモデルによる予測の精度と頑健性の向上を狙う。それを可能にするために、文と DRS の両方に現れるトークンは、該当の DRS トークンを文中の出現位置を明示的に示すトークンに書き換えたデータで学習し、予測をする。出現位置に置き換えたデータで学習したエンコーダ・デコーダモデルは予測の際にも出現位置トークンとして出力し、出現位置トークンが示す文中の単語に書き換えることで最終的な予測とする。

2 関連研究

2.1 Parallel Meaning Bank

Parallel Meaning Bank (PMB) [4] は大規模な DRS コーパスの一種であり、英語、ドイツ語、イタリア語、オランダ語の 4 言語のデータがデータセットとして提供されている。データは文と DRS 以外にも、各トークンの品詞や意味タグなど多くの言語情報が付与されている。2023 年 1 月現在、4.0.0 までが利用可能である[1)]。各言語のデータは Gold, Silver, Bronze の 3 つに分割されており、Gold は完全に人手でアノテーションされたもの、Silver は一部が人手によるアノテーションが付与されたもの、Bronze は人手によるアノテーションを用いて学習したモデルが自動でアノテーションを付与したものがそれぞれ割り当てられている。表 1 に PMB 3.0.0 と 4.0.0 のうち、英語のデータセットの詳細を示す。

日本語 PMB データについてはデータセットして提供されていないものの、[5] の貢献により PMB Explorer[2)] では利用可能である。

2.2 DRS 意味解析

DRS は 80 年代に導入された談話表示理論 (Discourse Representation Theory) [2] に基づく意味表現形式であり、様々な DRS 意味解析手法が提案されている [6, 7]。近年は深層学習のモデルを用いた研究が多く、[3] は双方向 long short-term memory (bi-LSTM) [8] を用いている。[9] も深層学習のモデルを手法に組み入れているが、DRS は句形式ではなくグラフ形式を採用している。また [10] もグラフ形式を採用しており、いずれの研究も少量の学習データで比較的高い精度を達成している。

なお、本研究に最も近い研究は [11] である。この研究は我々が語彙数の増加の原因として着目しているトークンをダミー化し、これらをエンコーダ・デコーダモデルに予測させるもしくは後処理で埋める手法をとっている。語彙数の減少に注視する点は共通するが、本研究では完全にダミー化するのではなく、意味タグとトークンの出現位置との組み合わせによって精度上昇を狙う。

1) https://pmb.let.rug.nl/data.php
2) https://pmb.let.rug.nl/explorer/explore.php

```
1: $NEW TPR @1 "now"          6: $0 bear "v.02" @0
2: $0 Time @2 @1              7: $0 DOM @-1 "31"
3: $0 REF                     8: $0 time "n.08" @-1
4: $0 REF                     9: $0 MOY @-1 "05"
5: $0 Patient @0 "speaker"   10: $0 YOC @-1 "1940"
```

図 3 図 2 を、箱変数と個体変数をそれぞれ相対的に変換した相対句形式 DRS。見やすさのために開業している。なお、エンコーダへの入力では改行は *** と表される。

```
1: $NEW TPR @1 "now"          6: $0 #2 "v.02" @0
2: $0 Time @2 @1              7: $0 DOM @-1 "#5"
3: $0 REF                     8: $0 time "n.08" @-1
4: $0 REF                     9: $0 MOY @-1 "#7"
5: $0 Patient @0 "speaker"   10: $0 YOC @-1 "#9"
```

図 4 図 3 に、文中に現れるトークンをその出現位置で置き換えた相対句形式 DRS（提案手法）。

3 提案手法

出力する語彙数を抑制し、効果的に DRS トークンを予測する方法として以下の手法を提案する。学習の前処理として、句形式 DRS に含まれるトークンのうち、それらが文中にも出現する場合、文中における出現位置に書き換える。先行研究 [1] において、図 2 の句形式 DRS は、図 3 の**相対句形式 DRS** [3] に変換してエンコーダに入力する。相対句形式 DRS において、箱変数 $ と個体変数 @ はそれぞれの導入との相対位置で記述される。例えば、6 行目の第 4 トークン `@0` は最新の個体変数（5 行目の `REF = e1`）を指し、7 行目の第 3 トークン `@-1` は最新から一つ前の個体変数（3 行目の `REF = t1`）を指す。

提案手法は、図 3 の相対句形式 DRS をさらに図 4 に変換する。表 2 は句形式 DRS の行を分類したも

表 2 句形式 DRS の各行のフォーマット。ここで箱変数（b1 など）を $ で、個体変数（t1, e1 など）を @ で表している。

	タイプ	トークン数	第 1 トークン	第 2 トークン	第 3 トークン	第 4 トークン	例
A	変数導入	3	$	REF	@	–	b1 REF x1
B	箱間の関係	3 or 4	$	関係性	$	($)	b1 PRESUPPOSITION b2
C	意味	4	$	単語	語義	@	b2 child "n.02" x3
D	役割	4	$	役割	@	@ or ”値”	b2 Quantity x3 "4"

のであるが、書き換えられるトークンは大きく分けて 2 種類あり、タイプ C（6 行目）の第 2 トークンと、タイプ D（7, 9, 10 行目）の第 4 トークンである。

3.1 タイプ C の書き換え

タイプ C の第 3 トークンにおいて第 2 トークンの語義 (synset) が WordNet [12] 形式で与えられるため、その品詞情報を用いて文中の単語を原形に変換する。この原形が第 2 トークンと同等の場合、# に続いて、その単語の出現位置を付与する。例えば、図 3 の 6 行目の bear は、文中に（0 始まりで）2 番目に出現する動詞 *born* の原形であるので、 #2 に書き換えられる。

第 2 トークンが _（アンダースコア）繋がりで与えられる場合、そのトークンが複合名詞（mobile_phones など）のものと句動詞（get_up など）のものがある。複合名詞の場合、文中において ~（チルダ）繋がりで与えられているため、それらを空白として扱い、原形へ変換した上で一致するか否かで書き換えの有無を決定する。一方で句動詞の場合、構成する単語の間に別の単語が挟まる場合がある。例えば、*The boy has taken the toy away from his little~sister* には句動詞 *take away* が存在するが、それを構成する単語は隣接していない。そのため句動詞については、それを構成する 2 番目以降の単語の出現位置を、直前の単語からの相対位置として扱う。例えば、先述の文の相対句動詞 DRS に含まれる take_away は、文中で take は 3 番目（*taken* の原形）、 away は 6 番目に現れるので #3_3 に書き換えられる。

なお、以上の一連の判定で第 2 トークンが文中の単語と同等にならない場合、そのトークンは書き換えない。

3.2 タイプ D の書き換え

タイプ D で第 4 トークンが値の場合、そのトークンは図 2 の 6 行目の brown（名前）、17 行目の 4（数量）の他にも、okinawa（地名）や yen（単位）などが存在する。それらを完全に網羅することは困難であるため、以下の手法をとり文中に出現する単語と同じかを判定する。

まず、第 4 トークンの値と一致する単語が文中に存在すればその出現位置に書き換える。この規則により役割（第 2 トークンのことを表す）が Name（名称）や YearOfCentury （年）の値について対応可能になる。Unit（単位）についても、単数形に変換することにより一致すれば書き換えの対象になる。

次に、第 4 トークンの値に対応する単語が文中に存在すればその出現位置に書き換える。例えば、図 3 の 7 行目の 31 は文中では *31st* であり、これらが同等なので図 4 においてその出現位置に書き換えられている。この対応関係は人手で判定することは困難なため、学習データの対応関係から作成した辞書を用いて同等かを判定する。これにより Quantity（数量）、DayOfMonth（日にち）、ClockTime（時刻）など、数に関係するほとんどの役割について対応可能になる。

3.3 出現位置を基にした復元方法

タイプ C に関しては、出現位置の指す単語を品詞情報を用いて原形に変換し、出現位置トークンを書き換える。タイプ D に関しては、出現位置トークンへ変換する際に構成した辞書を用いて復元する。このときに、出現位置トークンの指す単語が辞書に含まれていないなどのエラーが発生した場合はダミーのトークンで書き換える。例えば $n DayOfMonth @m "#2" に対し文トークンの 2 番目に日にちを表す単語が存在しなかった場合、$n DayOfMonth @m "00" とする。

4 実験

4.1 実験設定

3.1 節と 3.2 節で説明した手法により PMB 3.0.0 の英語の Gold データと Silver データを書き換える。書き換えを行なった各セットを用い、van Noord et al. (2020) [1] で最も精度が良かった実験設定である、bi-LSTM エンコーダを 2 つ導入したエンコーダ・デコーダモデルを用いる。2 つのエンコーダのうち一

表 3　実験結果。[11] では ill-formed 数の報告がなかった。「– タイプ C」は「提案手法」において、タイプ C の復元をしなかった場合の値を表す。

	F-score (%)	ill-formed 数
van Noord et al. [1]	89.3	**2.0**
Shen and Evang [11]	88.4	N/A
提案手法	**89.9**	10.0
– タイプ C	74.9	10.0
– タイプ D	84.6	10.0

表 4　事前学習時間、消費 GPU メモリ、語彙数の比較。先行研究の各値は再現実験で得られた値である。

	事前学習時間（分）	消費 GPU メモリ (GB)	語彙数
van Noord et al. [1]	237	15.30	11,394
提案手法	**186**	**8.98**	**1,081**
差異の割合	21.5%	41.3%	90.5%

方は PMB で与えられたトークナイズされたトークン列と意味タグ [13] を交互に並べてエンコーダへ入力し、他方はトークン列を文字ごとに区切って入力する。

事前学習では Gold Train データと Silver Train データを合わせて用い、Gold Dev データでバリデーションを行なう。ファインチューニングでは Gold Train データと Gold Dev データを用いて学習・バリデーションを行ない、Gold Test データで評価する。一度の事前学習に対して 5 回のファインチューニングを施し、それらから得られた値の平均値を結果として採用する。

評価指標　各実験結果は Counter [14] を用いて得られる。この評価ツールは出力された句構造 DRS と正解とを比較して、行を単位とする F-score などの評価値を出力する。テストデータ全体の精度はミクロ F-score として求められる。例えば、句構造 DRS が 5 行からなる A と 20 行からなる B があるとする。ある DRS 意味解析器が A に対しては正解とまったく同じ出力をしたのに対し、B に対しては何らかのエラーにより一切の出力がなかった。このとき、Counter が出力する値は各々の F-score の平均値 $(1+0)/2 = 0.5$ ではなく、行数によって重み付けされて計算された F-score である $(5+0)/(5+20) = 0.2$ である。

4.2　結果と分析

表 3 に結果を示す。[1] が報告した値 89.3 と比較し、我々の手法を用いることでより高い精度である 89.9 を達成した。一方で、出力された相対句形式 DRS が ill-formed であったものの数は [1] と比較して増加した。ここで相対句形式 DRS が ill-formed であるとは、表 2 のいずれにも当てはまらない行が存在したり、変数の相対参照に失敗したりするなどで正しい句形式 DRS に変換できないことを表す。

アブレーション実験　表 3 の下 2 行はアブレーション実験の結果を示している。これはタイプ C（3.1 節）とタイプ D（3.2 節）の書き換えそれぞれがどの程度精度上昇に寄与したかを示すものである。表 3 によると、タイプ C の書き換えは約 15% 、タイプ D は約 5% の精度上昇をもたらしたことが観察できる。

事前学習時間、消費 GPU メモリ、語彙数の比較　表 4 に実験を行なった際の各指標を示す。我々の提案手法は精度上昇のみならず、事前学習時間を約 20% 以上減、GPU メモリの消費量を約 40% 以上減し、特に語彙数は 90% 以上減を達成した。

未知語に関する分析　[1] と提案手法の 5 回の予測のうちそれぞれ一つを抽出し比較したところ、[1] で未知語であることを示す `@@UNKNOWN@@` トークンは 250 個出力されていたが、そのうち提案手法において正しく予測されたトークンは 233 個であった。

ill-formed 数に関する分析　ill-formed になったケースは量化や否定など、学習データが不十分な言語現象を含むものが多くを占めた。特に、一度でも ill-formed になった 31 ケースのうち、接続や量化を示す意味タグ AND をもつものは 10 ケース存在していた。

5　おわりに

本研究では DRS の意味解析に着目し、句形式 DRS において文と共通する DRS トークンを出現位置トークンに置き換える手法を試みた。実験の結果、出力語彙数が大幅に減少したことにより、既存手法と比較し有意な精度上昇が見られ、さらに事前学習時間と GPU メモリの消費量も減少した。

今後は、PMB に含まれる英語以外の言語に対して、語彙数を抑制するアプローチを試みる予定である。また、DRS 意味解析以外の意味解析タスクへの応用なども考えられる。

謝辞

本研究は JST さきがけ JPMJPR21C8 の支援を受けたものである。

参考文献

[1] Rik van Noord, Antonio Toral, and Johan Bos. Character-level representations improve DRS-based semantic parsing even in the age of BERT. In **Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)**, pp. 4587–4603, Online, 2020. Association for Computational Linguistics.

[2] Hans Kamp and Uwe Reyle. **From Discourse to Logic**. Springer, Dordrecht, 1993.

[3] Rik van Noord, Lasha Abzianidze, Antonio Toral, and Johan Bos. Exploring neural methods for parsing discourse representation structures. **Transactions of the Association for Computational Linguistics**, Vol. 6, pp. 619–633, 2018.

[4] Lasha Abzianidze, Johannes Bjerva, Kilian Evang, Hessel Haagsma, Rik van Noord, Pierre Ludmann, Duc-Duy Nguyen, and Johan Bos. The Parallel Meaning Bank: Towards a multilingual corpus of translations annotated with compositional meaning representations. In **Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers**, pp. 242–247, Valencia, Spain, 2017. Association for Computational Linguistics.

[5] 谷中瞳, 峯島宏次, 山田彬尭, 山口悠, 窪田悠介, Lasha Abzianidze, Johan Bos. 多言語統語・意味情報コーパス Parallel Meaning Bank 日本語版の構築. 言語処理学会 第 26 回年次大会. 言語処理学会年次大会発表論文集, 2020.

[6] Hajime Wada and Nicholas Asher. BUILDRS: An implementation of DR theory and LFG. In **Proceedings of Coling 1986 Volume 1: The 11th International Conference on Computational Linguistics**, 1986.

[7] Johan Bos. Wide-coverage semantic analysis with Boxer. In **Proceedings of Semantics in Text Processing. STEP 2008 Conference**, pp. 277–286. College Publications, 2008.

[8] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. **Neural computation**, Vol. 9, pp. 1735–80, 1997.

[9] Federico Fancellu, Sorcha Gilroy, Adam Lopez, and Mirella Lapata. Semantic graph parsing with recurrent neural network DAG grammars. In **Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)**, pp. 2769–2778, Hong Kong, China, 2019. Association for Computational Linguistics.

[10] Wessel Poelman, Rik van Noord, and Johan Bos. Transparent semantic parsing with Universal Dependencies using graph transformations. In **Proceedings of the 29th International Conference on Computational Linguistics**, pp. 4186–4192, Gyeongju, Republic of Korea, 2022. International Committee on Computational Linguistics.

[11] Minxing Shen and Kilian Evang. DRS parsing as sequence labeling. In **Proceedings of the 11th Joint Conference on Lexical and Computational Semantics**, pp. 213–225, Seattle, Washington, 2022. Association for Computational Linguistics.

[12] George A. Miller. Wordnet: A lexical database for english. **Commun. ACM**, Vol. 38, No. 11, p. 39–41, 1995.

[13] Lasha Abzianidze and Johan Bos. Towards universal semantic tagging. In **Proceedings of the 12th International Conference on Computational Semantics**, 2017.

[14] Rik van Noord, Lasha Abzianidze, Hessel Haagsma, and Johan Bos. Evaluating scoped meaning representations. In **Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)**, Miyazaki, Japan, 2018. European Language Resources Association (ELRA).