Discourse Representation Structure Parsing

Citation for published version:

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published In:
56th Annual Meeting of the Association for Computational Linguistics

General rights
Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.
Discourse Representation Structure Parsing

Jiangming Liu    Shay B. Cohen    Mirella Lapata
Institute for Language, Cognition and Computation
School of Informatics, University of Edinburgh
10 Crichton Street, Edinburgh EH8 9AB
Jiangming.Liu@ed.ac.uk, scohen@inf.ed.ac.uk, mlap@inf.ed.ac.uk

Abstract
We introduce an open-domain neural semantic parser which generates formal meaning representations in the style of Discourse Representation Theory (DRT; Kamp and Reyle 1993). We propose a method which transforms Discourse Representation Structures (DRSs) to trees and develop a structure-aware model which decomposes the decoding process into three stages: basic DRS structure prediction, condition prediction (i.e., predicates and relations), and referent prediction (i.e., variables). Experimental results on the Groningen Meaning Bank (GMB) show that our model outperforms competitive baselines by a wide margin.

1 Introduction
Semantic parsing is the task of mapping natural language to machine interpretable meaning representations. A variety of meaning representations have been adopted over the years ranging from functional query language (FunQL; Kate et al. 2005) to dependency-based compositional semantics (λ-DCS; Liang et al. 2011), lambda calculus (Zettlemoyer and Collins, 2005), abstract meaning representations (Banarescu et al., 2013), and minimal recursion semantics (Copestake et al., 2005).

Existing semantic parsers are for the most part data-driven using annotated examples consisting of utterances and their meaning representations (Zelle and Mooney, 1996; Wong and Mooney, 2006; Zettlemoyer and Collins, 2005). The successful application of encoder-decoder models (Sutskever et al., 2014; Bahdanau et al., 2015) to a variety of NLP tasks has provided strong impetus to treat semantic parsing as a sequence transduction problem where an utterance is mapped to a target meaning representation in string format (Dong and Lapata, 2016; Jia and Liang, 2016; Kočiský et al., 2016). The fact that meaning representations do not naturally conform to a linear ordering has also prompted efforts to develop recurrent neural network architectures tailored to tree or graph-structured decoding (Dong and Lapata, 2016; Cheng et al., 2017; Yin and Neubig, 2017; Alvarez-Melis and Jaakkola, 2017; Rabinovich et al., 2017; Buys and Blunsom, 2017).

Most previous work focuses on building semantic parsers for question answering tasks, such as querying a database to retrieve an answer (Zelle and Mooney, 1996; Cheng et al., 2017), or conversing with a flight booking system (Dahl et al., 1994). As a result, parsers trained on query-based datasets work on restricted domains (e.g., restaurants, meetings; Wang et al. 2015), with limited vocabularies, exhibiting limited compositionality, and a small range of syntactic and semantic constructions. In this work, we focus on open-domain semantic parsing and develop a general-purpose system which generates formal meaning representations in the style of Discourse Representation Theory (DRT; Kamp and Reyle 1993).

DRT is a popular theory of meaning representation designed to account for a variety of linguistic phenomena, including the interpretation of pronouns and temporal expressions within and across sentences. Advantageously, it supports meaning representations for entire texts rather than isolated sentences which in turn can be translated into first-order logic. The Groningen Meaning Bank (GMB; Bos et al. 2017) provides a large collection of English texts annotated with Discourse Representation Structures (see Figure 1 for an example). GMB integrates various levels of semantic annotation (e.g., anaphora, named entities, thematic roles, rhetorical relations) into a unified formalism providing expressive meaning representations for open-domain texts.

We treat DRT parsing as a structure prediction problem. We develop a method to transform DRSs to tree-based representations which can be further linearized to bracketed string format. We examine a series of encoder-decoder models (Bahdanau et al., 2015) differing in the way tree-
The statement says each of the dead men wore magazine vests and carried two hand grenades.

Figure 1: DRT meaning representation for the sentence The statement says each of the dead men wore magazine vests and carried two hand grenades.

2 Discourse Representation Theory

In this section we provide a brief overview of the representational semantic formalism used in the GMB. We refer the reader to Bos et al. (2017) and Kamp and Reyle (1993) for more details.

Discourse Representation Theory (DRT; Kamp and Reyle 1993) is a general framework for representing the meaning of sentences and discourse which can handle multiple linguistic phenomena including anaphora, presuppositions, and temporal expressions. The basic meaning-carrying units in DRT are Discourse Representation Structures (DRSs), which are recursive formal meaning structures that have a model-theoretic interpretation and can be translated into first-order logic (Kamp and Reyle, 1993). Basic DRSs consist of discourse referents (e.g., \(x, y\)) representing entities in the discourse and discourse conditions (e.g., \(\text{man}(x), \text{magazine}(y)\)) representing information about discourse referents. Following conventions in the DRT literature, we visualize DRSs in a box-like format (see Figure 1).

GMB adopts a variant of DRT that uses a neo-Davidsonian analysis of events (Kipper et al., 2008), i.e., events are first-order entities characterized by one-place predicate symbols (e.g., \(\text{say}(e)\) in Figure 1). In addition, it follows Projective Discourse Representation Theory (PDRT; Venhuizen et al. 2013) an extension of DRT specifically developed to account for the interpretation of presuppositions and related projection phenomena.
Basic DRSs consist of a set of referents \( \langle \text{ref} \rangle \) (denoting individuals or discourse referents) and \( \langle \text{exp} \rangle \) (i.e., truth values):

\[
\langle \text{exp} \rangle := \langle \text{ref} \rangle, \quad \langle \text{exp} \rangle := \langle \text{drs} \rangle | \langle \text{sdrs} \rangle \tag{1}
\]

discourse referents \( \langle \text{ref} \rangle \) are in turn classified into six categories, namely common referents \( \langle x_n \rangle \), event referents \( \langle e_n \rangle \), state referents \( \langle s_n \rangle \), segment referents \( \langle k_n \rangle \), proposition referents \( \langle \pi_n \rangle \), and time referents \( \langle t_n \rangle \). \( \langle \text{drs} \rangle \) and \( \langle \text{sdrs} \rangle \) denote basic and segmented DRSs, respectively:

\[
\langle \text{drs} \rangle := \langle \text{pvar} \rangle : \frac{[\langle \text{pvar} \rangle, \langle \text{ref} \rangle]^*}{[\langle \text{pvar} \rangle, \langle \text{condition} \rangle]^*} \tag{2}
\]

\[
\langle \text{sdrs} \rangle := \frac{k_1 : \langle \text{exp} \rangle, k_2 : \langle \text{exp} \rangle}{\text{coo}(k_1, k_2)} | \frac{k_1 : \langle \text{exp} \rangle, k_2 : \langle \text{exp} \rangle}{\text{sub}(k_1, k_2)} \tag{3}
\]

Basic DRSs consist of a set of referents \( \langle \langle \text{ref} \rangle \rangle \) and conditions \( \langle \langle \text{condition} \rangle \rangle \), whereas segmented DRSs are recursive structures that combine two \( \langle \langle \text{exp} \rangle \rangle \) by means of coordinating \( \langle \text{coo} \rangle \) or subordinating \( \langle \text{sub} \rangle \) relations. DRS conditions can be basic or complex:

\[
\langle \langle \text{condition} \rangle \rangle := \langle \langle \text{basic} \rangle \rangle | \langle \langle \text{complex} \rangle \rangle \tag{4}
\]

Basic conditions express properties of discourse referents or relations between them:

\[
\langle \langle \text{basic} \rangle \rangle := \langle \text{sym}_1 \rangle (\langle \text{exp} \rangle) | \langle \text{sym}_2 \rangle (\langle \text{exp} \rangle, \langle \text{exp} \rangle) | \langle \langle \text{exp} \rangle \rangle = \langle \langle \text{exp} \rangle \rangle | \langle \text{time} \rangle (\langle \text{exp} \rangle, \langle \text{sym}_0 \rangle) | \langle \text{named} \rangle (\langle \text{exp} \rangle, \langle \text{sym}_0 \rangle, \langle \text{class} \rangle) \tag{5}
\]

where \( \langle \text{sym}_n \rangle \) denotes \( n \)-place predicates, \( \langle \text{num} \rangle \) denotes cardinal numbers, \( \langle \text{time} \rangle \) expresses temporal information (e.g., \( \langle \text{time} \rangle(x_7, 2005) \) denotes the year 2005), and \( \langle \text{class} \rangle \) refers to named entity classes (e.g., location).

Complex conditions are unary or binary. Unary conditions have one DRS as argument and represent negation \( \langle \neg \rangle \) and modal operators expressing necessity \( \langle \Box \rangle \) and possibility \( \langle \Diamond \rangle \). Condition

<table>
<thead>
<tr>
<th>sections</th>
<th># doc</th>
<th># sent</th>
<th># token</th>
<th>avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>00-99</td>
<td>10,000</td>
<td>62,010</td>
<td>1,354,149</td>
<td>21.84</td>
</tr>
<tr>
<td>20-99</td>
<td>7,970</td>
<td>49,411</td>
<td>1,078,953</td>
<td>21.83</td>
</tr>
<tr>
<td>10-19</td>
<td>1,038</td>
<td>6,483</td>
<td>142,344</td>
<td>21.95</td>
</tr>
<tr>
<td>00-09</td>
<td>992</td>
<td>6,116</td>
<td>132,852</td>
<td>21.72</td>
</tr>
</tbody>
</table>

Table 1: Statistics on the GMB (avg denotes the average number of tokens per sentence).

\( \langle \text{ref} \rangle : \langle \text{exp} \rangle \) represents verbs with propositional content (e.g., factive verbs). Binary conditions are conditional statements \( \langle \rightarrow \rangle \) and questions.

\[
\langle \langle \text{complex} \rangle \rangle := \langle \langle \text{unary} \rangle \rangle | \langle \langle \text{binary} \rangle \rangle, \tag{6}
\]

\[
\langle \langle \text{unary} \rangle \rangle := \neg \langle \langle \text{exp} \rangle \rangle | \Box \langle \langle \text{exp} \rangle \rangle | \Diamond \langle \langle \text{exp} \rangle \rangle, \tag{6}\langle \langle \text{binary} \rangle \rangle := \langle \langle \text{exp} \rangle \rangle \rightarrow \langle \langle \text{exp} \rangle \rangle | \langle \langle \text{exp} \rangle \rangle \vee \langle \langle \text{exp} \rangle \rangle | \langle \langle \text{exp} \rangle \rangle \wedge \langle \langle \text{exp} \rangle \rangle |
\]
a necessary first step for more global semantic representations. It is relatively straightforward to obtain sentence-level DRSs from document-level annotations since referents and conditions are indexed to tokens. We match each sentence in a document with the DRS whose content bears the same indices as the tokens occurring in the sentence. This matching process yields 52,268 sentences for training (sections 20–99), 5,172 sentences for development (sections 10–19), (development), and 5,440 sentences for testing (sections 00–09).

In order to simplify the representation, we omit referents in the top part of the DRS (e.g., $x_1, e_1$, and $\pi_1$ in Figure 1) but preserve them in conditions without any information loss. Also we ignore pointers to DRSs since this information is implicitly captured through the typing and co-indexing of referents. Definition (1) is simplified to:

$$\text{drs} ::= \text{DRS}(\text{condition}^*)$$

(7)

where DRS() denotes a basic DRS. We also modify discourse referents to SDRSs (e.g., $k_1, k_2$ in Figure 1) which we regard as elements bearing scope over expressions $\langle \text{exp}_i \rangle$ and add a 2-place predicate $\langle \text{sym}_2 \rangle$ to describe the discourse relation between them. So, definition (3) becomes:

$$\text{sdr} ::= \text{SDRS}(\langle \text{ref} \rangle (\langle \text{exp}_i \rangle)^* \langle \text{sym}_2 \rangle (\langle \text{ref} \rangle (\langle \text{ref} \rangle))^*)$$

(8)

where SDRS() denotes a segmented DRS, and $\langle \text{ref} \rangle$ are segment referents.

We treat cardinal numbers $\langle \text{num} \rangle$ and $\langle \text{sym}_0 \rangle$ in relation timex as constants. We introduce the binary predicate “card” to represent cardinality (e.g., $|x_8|=2$ is card($x_8,$NUM)). We also simplify $\langle \text{exp}_c \rangle = \langle \text{exp}_r \rangle$ to eq($\langle \text{exp}_r \rangle, \langle \text{exp}_l \rangle$) using the binary relation “eq” (e.g., $x_1=x_2$ becomes eq($x_1,x_2$)). Moreover, we ignore class in named and transform named($\langle \text{exp}_n \rangle$, $\langle \text{sym}_m \rangle$, $\langle \text{class} \rangle$) into $\langle \text{sym}_1 \rangle (\langle \text{sym}_2 \rangle)$ (e.g., named($x_2,$mongolia,geo) becomes mongolia($x_2$)). Consequently, basic conditions (see definition (5)) are simplified to:

$$\text{basic} ::= \langle \text{sym}_1 \rangle (\langle \text{exp}_n \rangle) (\langle \text{sym}_2 \rangle (\langle \text{exp}_c \rangle, \langle \text{exp}_l \rangle))$$

(9)

Analogously, we treat unary and binary conditions as scoped functions, and definition (6) becomes:

$$\text{unary} ::= \rightarrow | \odot | \odot | \langle \text{ref} \rangle (\langle \text{exp}_r \rangle)$$

$$\text{binary} ::= \rightarrow | \lor | ?(\langle \text{exp}_c \rangle, \langle \text{exp}_l \rangle)$$

(10)

Following the transformations described above, the DRS in Figure 1 is converted into the tree in Figure 2, which can be subsequently linearized into a PTB-style bracketed sequence. It is important to note that the conversion does not diminish the complexity of DRSs. The average tree width in the training set is 10.39 and tree depth is 4.64.

4 Semantic Parsing Models

We present below three encoder-decoder models which are increasingly aware of the structure of the DRT meaning representations. The models take as input a natural language sentence $X$ represented as $w_1,w_2,\ldots,w_m$, and generate a sequence $Y=(y_1,y_2,\ldots,y_n)$, which is a linearized tree (see Figure 2 bottom), where $n$ is the length of the sentence, and $m$ the length of the generated DRS sequence. We aim to estimate $p(Y|X)$, the conditional probability of the semantic parse tree $Y$ given natural language input $X$:

$$p(Y|X) = \prod_j p(y_j|Y_{j-1},X^t)$$

4.1 Encoder

An encoder is used to represent the natural language input $X$ into vector representations. Each token in a sentence is represented by a vector $x_k$ which is the concatenation of randomly initialized embeddings $e_{w_i}$, pre-trained word embeddings $\tilde{e}_{w_i}$, and lemma embeddings $e_l$: $x_k = \text{tanh}(e_{w_i};\tilde{e}_{w_i};e_l) * W_i + b_l$, where $W_i \in \mathbb{R}^{d_{w_i}}$ and $D$ is a shorthand for $(d_w + d_p + d_l) \times d_{\text{input}}$ (subscripts $w$, $p$, and $l$ denote the dimensions of word embeddings, pre-trained embeddings, and lemma embeddings, respectively); $b_l \in \mathbb{R}^{d_{\text{input}}}$ and the symbol $;$ denotes concatenation. Embeddings $e_{w_i}$,
and $e_{t}$ are randomly initialized and tuned during training, while $\bar{e}_{w_i}$ are fixed.

We use a bidirectional recurrent neural network with long short-term memory units (bi-LSTM; Hochreiter and Schmidhuber 1997) to encode natural language sentences:

$$[h_{c_i} ; h_{e_i}] = \text{bi-LSTM}(x_1 : x_n),$$

where $h_{c_i}$ denotes the hidden representation of the encoder, and $x_i$ refers to the input representation of the $i$th token in the sentence. Table 2 summarizes the notation used throughout this paper.

### 4.2 Sequence Decoder

We employ a sequential decoder (Bahdanau et al., 2015) as our baseline model with the architecture shown in Figure 3(a). Our decoder is a (forward) LSTM, which is conditionally initialized with the hidden state of the encoder, i.e., we set $h_{d_0} = h_{e_n}$ and $c_{d_0} = c_{e_n}$, where $c$ is a memory cell:

$$h_{d_i} = \text{LSTM}(e_{y_{i-1}}),$$

where $h_{d_i}$ denotes the hidden representation of $y_{i}$, $e_{y_{i}}$ are randomly initialized embeddings tuned during training, and $y_0$ denotes the start of sequence.

The decoder uses the contextual representation of the encoder together with the embedding of the previously predicted token to output the next token from the vocabulary $V$:

$$s_j = [h_{ct_j} ; e_{y_{j-1}}] \ast W_2 + b_2,$$

where $W_2 \in \mathbb{R}^{(d_{enc} + d_y) \times |V|}$, $b_2 \in \mathbb{R}^{|V|}$, $d_{enc}$ and $d_y$ are the dimensions of the encoder hidden unit and output representation, respectively, and $h_{ct_j}$ is obtained using an attention mechanism:

$$h_{ct_j} = \sum_{i=1}^{n} \beta_{ji} h_{e_i},$$

where the weight $\beta_{ji}$ is computed by:

$$\beta_{ji} = \frac{e^{f(h_{d_i} ; h_{e_i})}}{\sum_{k} e^{f(h_{d_k} ; h_{e_k})}},$$

and $f$ is the dot-product function. We obtain the probability distribution over the output tokens as:

$$p_j = p(y_j | Y_{1}^{j-1}, X_{1}^{n}) = \text{SOFTMAX}(s_j)$$

### 4.3 Shallow Structure Decoder

The baseline decoder treats all conditions in a DRS uniformly and has no means of distinguishing between conditions corresponding to tokens in a sentence (e.g., the predicate $\text{say}(e_1)$ refers to the verb $\text{said}$) and semantic relations (e.g., $\text{Cause}(e_1, x_i)$). Our second decoder attempts to take this into account by distinguishing conditions which are local and correspond to words in a sentence from items which are more global and express semantic content (see Figure 3(b)). Specifically, we model sentence specific conditions using a copying mechanism, and all other conditions $G$ which do not correspond to sentential tokens (e.g., thematic roles, rhetorical relations) with an insertion mechanism.

Each token in a sentence is assigned a copying score $o_{ji}$:

$$o_{ji} = h_{d_i}^{\top} W_3 h_{e_i},$$

where subscript $ji$ denotes the $i$th token at $j$th time step, and $W_3 \in \mathbb{R}^{d_{enc} \times d_{enc}}$. All other conditions $G$ are assigned an insertion score:

$$s_j = [h_{ct_j} ; e_{y_{j-1}}] \ast W_4 + b_4,$$

where $W_4 \in \mathbb{R}^{(d_{enc} + d_y) \times |G|}$, $b_4 \in \mathbb{R}^{|G|}$, and $h_{ct_j}$ are the same with the baseline decoder. We obtain the probability distribution over output tokens as:

$$p_j = p(y_j | Y_{1}^{j-1}, X_{1}^{n}) = \text{SOFTMAX}([o_j ; s_j])$$
The state represents copy and insertion scores, respectively. Blue boxes are encoder hidden units, red boxes are decoder LSTM hidden units, green and yellow boxes are not displayed: (c.1) predicts DRS structure, (c.2) predicts conditions, and (c.3) predicts referents. DRS(π) rewritten as: 

\[ \bar{Y} = \prod_{j=1}^{k} \bar{Y}_j \]

Our third decoder (see Figure 3(c)) first predicts the structural make-up of the DRS, then the conditions, and finally their referents in an end-to-end framework. The probability distribution of structured output \( Y \) given natural language input \( X \) is rewritten as:

\[
p(Y|X) = p(\hat{Y}, \bar{Y}, y_j|X) = \prod_{j=1}^{k} p(\hat{y}_j|\hat{Y}^{j-1}, X) \times \prod_{j=1}^{k} p(\bar{y}_j|\bar{Y}^{j-1}, \hat{Y}^j, X)
\]

\[ \hat{Y}^j \] denotes the structure predicted before conditions \( \hat{y}_j \); \( \bar{Y}^j \) and \( \hat{Y}^j \) are the structures and conditions predicted before referents \( \hat{y}_j \). We next discuss how each decoder is modeled.

**Structure Prediction** To model basic DRS structure we apply the shallow decoder discussed in Section 4.3 and also shown in Figure 3(c.1). Tokens in such structures correspond to parent nodes in a tree; in other words, they are all inserted from \( G \), and subsequently predicted tokens are only scored with the insert score, i.e., \( \hat{s}_t = s_t \). The hidden units of the decoder are:

\[ \hat{h}_{dt} = \text{LSTM}(e_{\hat{y}_{j-1}}), \]

And the probabilistic distribution over structure denoting tokens is:

\[ p(y_j|Y_{j-1}, X) = \text{SOFTMAX}(\hat{s}_j) \]

**Condition Prediction** DRS conditions are generated by taking previously predicted structures into account, e.g., when “DRS(” or “SDRS(” are predicted, their conditions will be generated next. By mapping \( j \) to \( (k, m_k) \), the sequence of conditions can be rewritten as \( \hat{y}_1, \ldots, \hat{y}_j, \ldots, \hat{y}_r = \bar{y}_{1(1)}, \bar{y}_{1(2)}, \ldots, \bar{y}_{(k,m_k)} \), where \( \bar{y}_{(k,m_k)} \) is \( m_k \)th
condition of structure token $\hat{y}_k$. The corresponding hidden units $\tilde{h}_{d_j}$ act as conditional input to the decoder. Structure denoting tokens (e.g., “DRS(“) or “SDRS(“) are fed into the decoder one by one to generate the corresponding conditions as:

$$e_{\hat{y}(k,0)} = \tilde{h}_{d_j} \ast W_S + b_S,$$

where $W_S \in \mathbb{R}^{d_{dec} \times d_j}$ and $b_S \in \mathbb{R}^{d_j}$. The hidden unit of the conditions decoder is computed as:

$$\tilde{h}_{d_j} = \tilde{h}_{d_j,k_{mk}} = \text{LSTM}(e_{\hat{y}(k,m_{-1})}),$$

Given hidden unit $\tilde{h}_{d_j}$, we obtain the copy score $\tilde{c}_{j}$ and insert score $\tilde{d}_{j}$. The probabilistic distribution over conditions is:

$$p(\hat{y}_j | \tilde{y}_j^{1-1}, \tilde{y}_j^{'}, X) = \text{SOFTMAX}([\tilde{j}_j; \tilde{c}_j]).$$

Referent Prediction Referents are generated based on the structure and conditions of the DRS. Each condition has at least one referent. Similar to condition prediction, the sequence of referents can be rewritten as $\hat{y}_1, \ldots, \hat{y}_j, \ldots, \hat{y}_e = \hat{y}(1.1); \hat{y}(1.2); \ldots; \hat{y}(k,m_k) \ldots$. The hidden units of the conditions decoder are fed into the referent decoder $e_{\hat{y}(k,0)} = \tilde{h}_{d_{j}} \ast W_S + b_6$, where $W_6 \in \mathbb{R}^{d_{dec} \times d_j}$, $b_6 \in \mathbb{R}^{d_j}$. The hidden unit of the referent decoder is computed as:

$$\tilde{h}_{d_j} = \tilde{h}_{d_{j},k_{mk}} = \text{LSTM}(e_{\hat{y}(k,m_{-1})}),$$

All referents are inserted from $\mathcal{G}$, given hidden unit $\tilde{h}_{d_j}$ (we only obtain the insert score $\tilde{d}_j$). The probabilistic distribution over predicates is:

$$p(\hat{y}_j | \tilde{y}_j^{1-1}, \tilde{y}_j^{'}, \tilde{y}_j^{''}, X) = \text{SOFTMAX}(\tilde{d}_j).$$

Note that a single LSTM is adopted for structure, condition and referent prediction. The mathematical symbols are summarized in Table 2.

4.5 Training

The models are trained to minimize a cross-entropy loss objective with $\ell_2$ regularization:

$$L(\theta) = -\sum_j \log p_j + \frac{\lambda}{2} ||\theta||^2,$$

where $\theta$ is the set of parameters, and $\lambda$ is a regularization hyper-parameter ($\lambda = 10^{-6}$). We used stochastic gradient descent with Adam (Kingma and Ba, 2014) to adjust the learning rate.

5 Experimental Setup

Settings Our experiments were carried out on the GMB following the tree conversion process discussed in Section 3. We adopted the training, development, and testing partitions recommended in Bos et al. (2017). We compared the three models introduced in Section 4, namely the baseline sequence decoder, the shallow structured decoder and the deep structure decoder. We used the same empirical hyper-parameters for all three models. The dimensions of word and lemma embeddings were 64 and 32, respectively. The dimensions of hidden vectors were 256 for the encoder and 128 for the decoder. The encoder used two hidden layers, whereas the decoder only one. The dropout rate was 0.1. Pre-trained word embeddings (100 dimensions) were generated with Word2Vec trained on the AFP portion of the English Gigaword corpus.\(^3\)

Evaluation Due to the complex nature of our structured prediction task, we cannot expect model output to exactly match the gold standard. For instance, the numbering of the referents may be different, but nevertheless valid, or the order of the children of a tree node (e.g., “DRS(india(x_1) say(e_1))” and “DRS(say(e_1) india(x_1))” are the same). We thus use $F_1$ instead of exact match accuracy. Specifically, we report D-match\(^4\) a metric designed to evaluate scoped meaning representations and released as part of the distribution of the Parallel Meaning Bank corpus (Abzianidze et al., 2017). D-match is based on Smatch\(^5\), a metric used to evaluate AMR graphs (Cai and Knight, 2013); it calculates $F_1$ on discourse representation graphs (DRGs), i.e., triples of nodes, arcs, and their referents, applying multiple restarts to obtain a good referent (node) mapping between graphs.

We converted DRSs (predicted and goldstandard) into DRGs following the top-down procedure described in Algorithm 1. IsCONDITION returns true if the child is a condition (e.g., “india(x_1)”), where three arcs are created, one is connected to a parent node and the other two are connected to arg1 and arg2, respectively (lines 7–12). IsQUANTIFIER returns true if the child is a quantifier (e.g., “π” and “□”) and three arcs are created; one is connected to the parent node, one to the referent that is created if and only

\(^3\)The models are trained on a single GPU without batches.

\(^4\)https://github.com/RikVM/D-match

\(^5\)https://github.com/snowblink14/smatch

\(^6\)We refer the interested reader to the supplementary material for more details.
Algorithm 1 DRS to DRG Conversion

\textbf{Input:} \( T \), tree-like DRS  
\textbf{Output:} \( G \), a set of edges

1: \( n_b \leftarrow 0; \ n_c \leftarrow 0; \ G \leftarrow \emptyset \)  
2: \( \text{stack} \leftarrow []; \ R \leftarrow \emptyset \)  
3: \textbf{procedure TRAVELDRS}(parent)  
4: \( \text{stack}.\text{append}(b_n); \ n_b \leftarrow n_b + 1 \)  
5: \( \text{node}_p \leftarrow \text{stack}\text{.top} \)  
6: \textbf{for} child in parent \textbf{do}  
7: \textbf{if} \( \text{ISCONDITION}(\text{child}) \) \textbf{then}  
8: \( G \leftarrow G \cup \{ \text{node}_p \xrightarrow{\text{child}.\text{ref}} c_n \} \)  
9: \( G \leftarrow G \cup \{ c_n \xrightarrow{\text{arg1}} \text{child.arg1} \} \)  
10: \( G \leftarrow G \cup \{ c_n \xrightarrow{\text{arg2}} \text{child.arg2} \} \)  
11: \( n_c \leftarrow n_c + 1 \)  
12: \( \text{ADDREFERENT}(\text{node}_p, \text{child}) \)  
13: \textbf{else} \( \text{ISQUANTIFIER}(\text{child}) \) \textbf{then}  
14: \( G \leftarrow G \cup \{ \text{node}_p \xrightarrow{\text{child.class}} c_n \} \)  
15: \( G \leftarrow G \cup \{ c_n \xrightarrow{\text{arg1}} \text{child.arg1} \} \)  
16: \( G \leftarrow G \cup \{ c_n \xrightarrow{\text{arg2}} b_{n+1} \} \)  
17: \( n_c \leftarrow n_c + 1 \)  
18: \textbf{if} \( \text{ISPROPSEG}(\text{child}) \) \textbf{then}  
19: \( \text{ADDREFERENT}(\text{node}_p, \text{child}) \)  
20: \textbf{end if}  
21: \( \text{TRAVELDRS}(\text{child.nextDRS}) \)  
22: \textbf{end if}  
23: \textbf{end for}  
24: \( \text{stack}.\text{pop}() \)  
25: \textbf{end procedure}  
26: \textbf{procedure ADDREFERENT}(\text{node}_p, \text{child})  
27: \textbf{if} \( \text{child.arg1} \) \text{not in} \( R \) \text{then}  
28: \( G \leftarrow G \cup \{ \text{node}_p \xrightarrow{\text{ref}} \text{child.arg1} \} \)  
29: \( R \leftarrow R \cup \text{child.arg1} \)  
30: \textbf{end if}  
31: \textbf{if} \( \text{child.arg2} \) \text{not in} \( R \) \text{then}  
32: \( G \leftarrow G \cup \{ \text{node}_p \xrightarrow{\text{ref}} \text{child.arg2} \} \)  
33: \( R \leftarrow R \cup \text{child.arg2} \)  
34: \textbf{end if}  
35: \textbf{end procedure}  
36: \( \text{TRAVELDRS}(T) \)  
37: return \( G \)
Table 3: GMB development set.

<table>
<thead>
<tr>
<th>Model</th>
<th>P (%)</th>
<th>R (%)</th>
<th>F (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseline</td>
<td>51.35</td>
<td>63.85</td>
<td>56.92</td>
</tr>
<tr>
<td>shallow</td>
<td>67.88</td>
<td>63.53</td>
<td>65.63</td>
</tr>
<tr>
<td>deep</td>
<td>79.01</td>
<td>75.65</td>
<td>77.29</td>
</tr>
<tr>
<td>deep (--pre)</td>
<td>78.47</td>
<td>73.43</td>
<td>75.87</td>
</tr>
<tr>
<td>deep (--pre &amp; lem)</td>
<td>78.21</td>
<td>72.82</td>
<td>75.42</td>
</tr>
</tbody>
</table>

Table 4: GMB test set.

<table>
<thead>
<tr>
<th>Model</th>
<th>DRG</th>
<th>DRG w/o refs</th>
<th>DRG w/o refs &amp; conds</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>R</td>
<td>F</td>
<td>P</td>
</tr>
<tr>
<td>baseline</td>
<td>52.21</td>
<td>64.46</td>
<td>57.69</td>
</tr>
<tr>
<td>shallow</td>
<td>66.61</td>
<td>63.92</td>
<td>65.24</td>
</tr>
<tr>
<td>deep</td>
<td>79.27</td>
<td>75.88</td>
<td>77.54</td>
</tr>
</tbody>
</table>

Figure 5: F1 score as a function of sentence length.

Figure 5 shows F1 performance for the three parsers on sentences of different length. We observe a similar trend for all models: as sentence length increases, model performance decreases. The baseline and shallow models do not perform well on short sentences which despite containing fewer words, can still represent complex meaning which is challenging to capture sequentially. On the other hand, the performance of the deep model is relatively stable. LSTMs in this case function relatively well, as they are faced with the easier task of predicting meaning in different stages (starting with a tree skeleton which is progressively refined). We provide examples of model output in the supplementary material.

7 Related Work

Tree-structured Decoding A few recent approaches develop structured decoders which make use of the syntax of meaning representations. Dong and Lapata (2016) and Alvarez-Melis and Jaakkola (2017) generate trees in a top-down fashion, while in other work (Xiao et al., 2016; Krishnamurthy et al., 2017) the decoder generates from a grammar that guarantees that predicted logical forms are well-typed. In a similar vein, Yin and Neubig (2017) generate abstract syntax trees (ASTs) based on the application of production rules defined by the grammar. Rabinovich et al. (2017) introduce a modular decoder whose various components are dynamically composed according to the generated tree structure. In comparison, our model does not use grammar information explicitly. We first decode the structure of the DRS, and then fill in details pertaining to its semantic content. Our model is not strictly speaking top-down, we generate partial trees sequentially, and then expand non-terminal nodes, ensuring that when we generate the children of a node, we have already obtained the structure of the entire tree.

Wide-coverage Semantic Parsing Our model is trained on the GMB (Bos et al., 2017), a richly annotated resource in the style of DRT which provides a unique opportunity for bootstrapping wide-coverage semantic parsers. Boxer (Bos, 2008) was a precursor to the GMB, the first semantic parser of this kind, which deterministically maps CCG derivations onto formal meaning representations. Le and Zuidema (2012) were the first to train a semantic parser on an early release of the GMB (2,000 documents; Basile et al. 2012), however, they abandon lambda calculus in favor of a graph based representation. The latter is closely related to AMR, a general-purpose meaning representation language for broad-coverage text. In AMR the meaning of a sentence is represented as a rooted, directed, edge-labeled and leaf-labeled graph. AMRs do not resemble classical meaning representations and do not have a model-theoretic interpretation. However, see Bos (2016) and Artzi et al. (2015) for translations to first-order logic.

8 Conclusions

We introduced a new end-to-end model for open-domain semantic parsing. Experimental results on the GMB show that our decoder is able to recover discourse representation structures to a good degree (77.54 F1), albeit with some simplifications. In the future, we plan to model document-level representations which are more in line with DRT and the GMB annotations.

Acknowledgments We thank the anonymous reviewers for their feedback and Johan Bos for answering several questions relating to the GMB. We gratefully acknowledge the support of the European Research Council (Lapata, Liu; award number 681760) and the EU H2020 project SUMMA (Cohen, Liu; grant agreement 688139).
References


Hans Kamp and Uwe Reyle. 1993. From discourse to logic: an introduction to modaltheoretic semantics of natural language, formal logic and DRT.


