Addressing Long-Distance Dependencies in AMR Parsing with Hierarchical Clause Annotation

Yunlong Fan , Bin Li , Yikemaiti Sataer , Miao Gao , Chuanqi Shi , Zhiqiang Gao *

Posted Date: 4 August 2023

doi: 10.20944/preprints202308.0405.v1

Keywords: hierarchical clause annotation; long-distance dependencies; AMR parsing; self-attention; curriculum learning

Article

Addressing Long-Distance Dependencies in AMR Parsing with Hierarchical Clause Annotation

Yunlong Fan [1,2], Bin Li [1,2], Yikemaiti Sataer [1,2], Miao Gao [1,2], Chuanqi Shi [1,2] and Zhiqiang Gao [1,2,*]

[1] School of Computer Science and Engineering, Southeast University, Nanjing 211189, China; fanyunlong@seu.edu.cn (Y.F.); lib@seu.edu.cn (B.L.); yikmat@seu.edu.cn (Y.S.); miaogao@seu.edu.cn (M.G.); chuanqi_shi@seu.edu.cn (C.S.)

[2] Key Laboratory of Computer Network and Information Integration, Southeast University, Ministry of Education, Nanjing 211189, China

* Correspondence: zqgao.seu.edu.cn (Z.G.)

Abstract: Most natural language processing (NLP) tasks operate an input sentence as a sequence with token-level embeddings and features, despite its clausal structures. Taking Abstract Meaning Representation (AMR) parsing as an example, recent parsers are empowered by Transformers and pre-trained language models, but long-distance dependencies (LDDs) introduced by long sequences are still open problems. We argue that LDDs are not superficially blamed on the sequence length but are essentially related to the internal clause hierarchy. Typically, non-verb words in a clause cannot depend on words outside, and verbs from different but related clauses have much longer dependencies than those in the same clause. With this intuition, we introduce a type of clausal feature, hierarchical clause annotation (HCA), into AMR parsing and propose two HCA-based approaches, HCA-based self-attention (HCA-SA) and HCA-based curriculum learning (HCA-CL), to integrate HCA trees of complex sentences for addressing LDDs. We conduct extensive experiments on two in-distribution (ID) AMR datasets (AMR 2.0 and AMR 3.0) and three out-of-distribution (OOD) ones (TLP, New3, and Bio). Experimental results show that our HCA-based approaches achieve significant and explainable improvements against the baseline model and outperform the state-of-the-art (SOTA) model when encountering sentences with complex clausal structures that introduce most LDD cases.

Keywords: hierarchical clause annotation; long-distance dependencies; AMR parsing; self-attention; curriculum learning

1. Introduction

Most natural language processing (NLP) tasks operate an input sentence as a word sequence with token-level embeddings and features, suffering long-distance dependencies (LDDs) when encountering long complex sentences such as dependency parsing [1], constituency parsing [2], semantic role labeling (SRL) [3], machine translation [4], discourse parsing [5] and text summarization [6]. In previous works, the length of a sentence is blamed for LDDs superficially, and several universal methods are proposed to cure this issue, e.g., hierarchical recurrent neural networks [7], long short-term memory (LSTM) [8], attention mechanism [9], Transformer [10], implicit graph neural networks [11], and such.

For example, abstract meaning representation (AMR) parsing [12], translating a sentence to a directed acyclic semantic graph with relations among abstract concepts, has witnessed some efforts for LDDs in different approaches. In transition-based strategies, Peng et al. [13] propose a cache system to predict arcs between long-distant words. In graph-based methods, Cai and Lam [14] present a graph↔sequence iterative inference to overcome inherent defects of the one-pass prediction process in parsing long sentences. In seq2seq-based approaches, Bevilacqua et al. [15] employ the Transformers-based pre-trained language model, BART [16], to address LDDs in long sentences. Among these categories, seq2seq-based approaches become mainstream and recent parsers [17–20] employ the seq2seq architecture with the popular codebase, *SPRING* [15], achieving

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0405.v1

better performances. Notably, *HGAN* [20] integrates token-level features, syntactic dependencies (SDP), and SRL with heterogeneous graph neural networks and becomes the state-of-the-art (SOTA) in the settings of removing extra silver training data, graph-categorization and ensemble methods.

However, these AMR parsers still suffer performance degradation when encountering long sentences with deeper AMR graphs [18,20] that introduces the most LDD cases. We argue that the complexity of the clausal structure inside a sentence is the essence of LDDs, where clauses are core units of grammar and center on a verb that determines the occurrences of other constituents [21]. Our intuition is that non-verb words in a clause typically can not depend on words outside, while dependencies between verbs correspond to the inter-clause relations, resulting in LDDs across clauses [22].

To prove our claim, we demonstrate the AMR graph of a sentence from the AMR 2.0 dataset [1] and distinguish the AMR relation distances in terms of different segment levels (clause/phrase/token) in Figure 1. Every AMR relation is represented as a dependent edge between two abstract AMR nodes that align to one or more input tokens. The dependency distances of inter-token relations are subtractive results from the indices of tokens aligned to the source and target nodes, while those of inter-phrase and inter-clause relations are calculated by indices of the headwords in phrases and the verbs [2] in clauses, respectively. As can be observed:

- Dependency distances of inter-clause relations are typically much longer than those of inter-phrase and inter-token relations, leading to the most LDD cases. E.g., the AMR relation, $anxious \xrightarrow{:ARG0-of} go\text{-}01$, occurring in the clause "*I get very anxious*" and its relative clause "*which does sort of go away ...*", has a dependency distance of 6 (subtracting the 9^{th} token "*anxious*" from the 15^{th} token "*go*").
- Reentrant AMR nodes abstracted from pronouns also bring the long distant AMR relations. E.g., the AMR relation, $wait\text{-}01 \xrightarrow{:ARG0} I$, has a dependency distance of 33 (subtracting the 1^{st} token "*I*" from the 34^{th} token "*wait*").

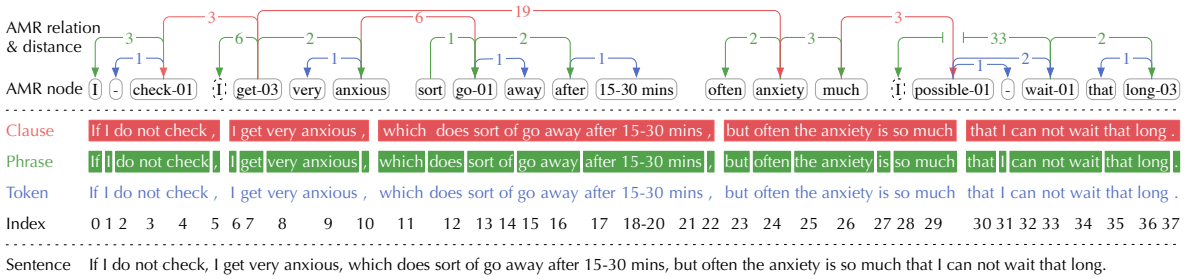


Figure 1. AMR relation dependency distances in different segment levels of an AMR 2.0 sentence. The input sentence is placed at the bottom, and the sentence's clause/phrase/token-level segments are positioned in the middle along with the token indices. The corresponding AMR graph is displayed at the top, where AMR relations are represented as directed edges with a dependency distance, i.e., the indices subtraction of two tokens mapping to the source/target AMR nodes. Inter-clause/phrase/token relations are distinguished in separate colors, corresponding to the segment levels' colors. Note that two virtual AMR nodes in dashed boxes of the reentrant node "*I*" are added for simplicity.

Based on the findings above, we are inspired to utilize the clausal features of a sentence to cure LDDs. Rhetorical structure theory (RST) [23] provides a general way to describe the coherence relations among clauses and some phrases, i.e., elementary discourse units, and postulates a hierarchical

[1] https://catalog.ldc.upenn.edu/LDC2017T10

[2] The AMR relation distances between a main clause and a relative/appositive clause are decided by the modified noun phrase in the former and the verb in the latter.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0405.v1

discourse structure called discourse tree. Except for RST, a novel clausal feature, hierarchical clause annotation (HCA) [24], also captures a tree structure of a complex sentence, where the leaves are segmented clauses, and the edges are the inter-clause relations.

Due to the better parsing performances of the clausal structure [24], we select and integrate the HCA trees of complex sentences to cure LDDs in AMR parsing. Specifically, we propose two HCA-based approaches, HCA-based self-attention (HCA-SA) and HCA-based curriculum learning (HCA-CL), to integrate the HCA trees as clausal features in the popular AMR parsing codebase, SPRING [15]. In HCA-SA, we convert an HCA tree into a clause adjacency matrix and a token visibility matrix to restrict the attention scores between tokens from unrelated clauses and increase those from related clauses in masked-self-attention encoder layers. In HCA-CL, we employ curriculum learning with two training curricula, Clause-Number and Tree-Depth, with the intuition that "the more number of clauses or the deeper clausal tree in a sentence, the more difficult it is to learn".

We conduct extensive experiments on two in-distribution (ID) AMR datasets (i.e., AMR 2.0 and AMR 3.0 [3]) and three out-of-distribution (OOD) ones (i.e., TLP, New3, and Bio) to evaluate our two HCA-based approaches. In ID datasets, our parser achieves 0.7 Smatch F1 score improvements against the baseline model, *SPRING*, on both AMR 2.0 and AMR 3.0., and outperforms the SOTA parser, *HGAN*, by 0.5 and 0.6 F1 scores of the fine-grained metric *SRL* in the two datasets. Notably, as the clause number of the sentence increases, our parser outperforms *SPRING* by a large margin and achieves better Smatch F1 scores than *HGAN*, indicating the ability to cure LDDs. In OOD datasets, the performance boosts achieved by our HCA-based approaches are more evident on complicated corpora like New3 and Bio, where sentences consist of more clauses and longer clauses. Our code is publicly available at https://github.com/MetroVancloud/HCA-AMRparsing (accessed on 3 August 2023).

The rest of this paper is organized as follows: The related works are summarized in Section 2, and the proposed approaches are detailed in Section 3. Then, the experiments of AMR parsing are presented in Section 4. Next, the discussion of the experimental results is presented in Section 5. Finally, our work is concluded in Section 6.

2. Related Work

In this section, we first introduce the open problem LDDs and some universal methods. Then, we summarize the four main categories of parsers and the LDD cases in AMR parsing. Finally, we introduce the novel clausal feature, HCA.

2.1. Long-Distance Dependencies

LDDs, first proposed by Hockett [25], describe an interaction between two (or more) elements in a sequence separated by an arbitrary number of positions. LDDs are related to the rate of decay of statistical dependence of two points with increasing time intervals or spatial distances between them [26]. In recent linguistic research of LDDs, Liu et al. [22] propose the verb-frame frequency account to robustly predict acceptability ratings in sentences with LDDs, indicating the affinities between the number of verbs and LDDs in sentences.

In recent advances in NLP tasks, hierarchical recurrent neural networks [7], LSTM [8], attention mechanism [9], Transformer [10], implicit graph neural networks [11] are proposed to cure LDDs. These universal neural models all represent the input sequence with token-level embeddings from pretrained language models in most NLP tasks.

---

[3] https://catalog.ldc.upenn.edu/LDC2020T02

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0405.v1

2.2. AMR Parsing

AMR parsing is a challenging semantic parsing task since AMR is a deep semantic representation consisting of many special annotations (e.g., abstract concept nodes, named entities, co-reference, and such.) [12]. The aim of AMR parsing is translating a sentence to a directed acyclic semantic graph with relations among abstract conceptsPrevious methods, where two main characteristics are:

1. *Abstraction*: assigns the same AMR to sentences with the same basic meaning and also brings a challenge of alignments between input tokens and output AMR nodes [12], e.g., the token "*can*" and its corresponding AMR node *possible*-01 in Figure 1.
2. *Reentrancy*: introduces the presence of nodes with multiple parents and represent sentences as graphs rather than trees [27], causing some LDD cases, e.g., the AMR node "*I*" in Figure 1.

Existing AMR parsers can be summarized into four categories:

1. *Graph-based:* Directly predict nodes and edges in either two-stage procedures [28–30] or incremental one-stage [14,31] procedures. The SOTA graph-based model, *AMR-gs* [14], enhances the incremental graph construction with an AMR graph↔sequence (AMR-gs) iterative inference mechanism in one-stage procedures.
2. *Seq2seq-based:* Model the task as transduction of the sentence into a linearization of the AMR graph [15,17–20,32–35]. *SPRING* [15] is a popular seq2seq-based codebase that employs transfer learning by exploiting pretrained encoder-decoder model, BART [16], to generate a linearized graph incrementally with a single auto-regressive pass of a seq2seq decoder. The subsequent models, *ANCES* [17], *HCL* [18], *ATP* [19], and *HGAN* [20] all follow the architecture of *SPRING*, where *HGAN* integrates SDP and SRL features with heterogeneous graph neural networks and achieves the best performances in the settings of removing extra silver training data, graph re-categorization, and ensemble methods.
3. *Transition-based:* Predict a sequence of actions that generate the graph while processing tokens left-to-right through the sentence [36–42]. The SOTA transition-based model, *StructBART* [42], explores the integration of general pre-trained sequence-to-sequence language models and a structure-aware transition-based approach.
4. *Grammar-based:* Peng et al. [43] introduce a synchronous hyperedge replacement grammar solution. Pust et al. [44] regard the task as a machine translation problem, while Artzi et al. [45] adapt combinatory categorical grammar. Groschwitz et al. [46] and Lindemann et al. [47] view AMR graphs as the structural AM algebra.

Despite different architectures, most AMR parsers employ word embeddings from pretrained language models and utilize token-level features like part-of-speech (POS), SDP, and SRL. However, these parsers still suffer performance degradation [18,20] when parsing complex sentences with LDDs due to the difficulties of aligning input tokens with output AMR nodes in such a long sequence.

2.3. Hierarchical Clause Annotation

RST [23] provides a general way to describe the coherence relations among parts in a text and postulates a hierarchical discourse structure called a discourse tree. The leaves of a discourse tree can be a clause or a phrase without strict definitions, known as elementary discourse units. However, the performances of RST parsing tasks are unsatisfactory due to the loose definitions of elementary discourse units and abundant types of discourse relations[48,49].

Except for RST, Fan et al. [24] also propose a novel clausal feature, HCA, which represents a complex sentence as a tree consisting of clause nodes and inter-clause relation edges. The HCA framework is based on the English grammar [21], where clauses are elementary grammar units that center around a verb, and inter-clause relations can be classified into two categories:

1. Coordination: is an equal relation shared by clauses with the same syntactic status, including *And*, *Or*, and *But* relations

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0405.v1

2. Subordination: occur in a matrix and a subordinate clause, including *Subjective*, *Objective*, *Predicative*, *Appositive*, *Relative*, and nine sublevel *Adverbial* relations.

Inter-clause relations have appropriate alignments with AMR relations, where nominal clause relations correspond to the frame arguments (e.g., *Subjective* vs. `:ARG0`) and adverbial clause relations are mapped to general semantic relations (e.g., *Adverbial_of_Condition* vs. `:conditon`). Figure 2 demonstrates the segmented clauses and the HCA tree of the same exemplified sentence in Figure 1.

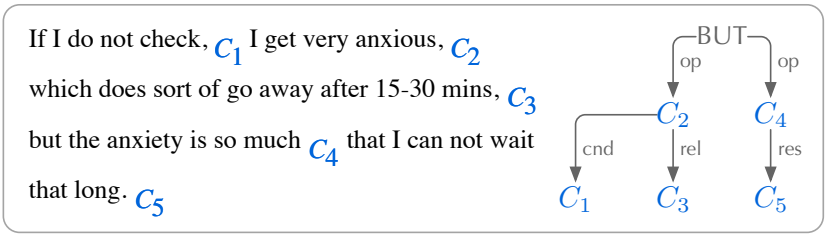


Figure 2. Segmented clauses and the HCA tree of a sentence in AMR 2.0. Clause C_2 and C_4 are contrasted and coordinated, dominated by the node *BUT*. Clause C_1, C_3, and C_5 are subordinated to their matrix clauses, where *cnd*, *rel*, and *res* represent the inter-clause relations of *Adverbial_of_Condition*, *Relative*, and *Adverbial_of_Result*, respectively.

Based on well-defined clauses and inter-clause relations, Fan et al. provide a manual-annotated HCA corpus for AMR 2.0 and high-performance neural models to generate silver HCA trees for more complex sentences. Therefore, we select and utilize the HCA trees as clausal features to address LDDs in AMR parsing.

3. Model

In this paper, we propose two HCA-based approaches, HCA-SA (Section 3.1) and HCA-CL (Section 3.2), to integrate HCA trees in the popular AMR parser codebase, *SPRING* for addressing LDDs in AMR parsing.

3.1. HCA-based Self-Attention

Existing AMR parsers (e.g., *SPRING*) employ Transformer [10] as an encoder to obtain the input sentence representation. However, in the standard self-attention mechanism adopted by Transformer, every token needs to attend to all other tokens, and the learned attention matrix A is often very sparse across most data points [50]. Inspired by the work of Liu et al. [51], we propose the HCA-SA approach, which utilizes hierarchical clause annotations as a structural bias to restrict the attention scope and attention scores.

3.1.1. Token Visibility Matrix

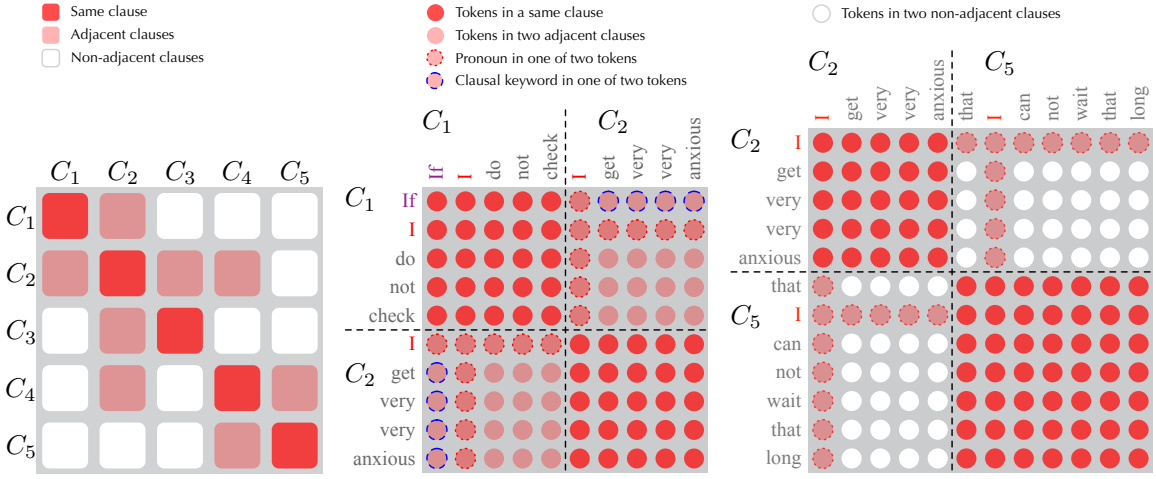
For the exemplified sentence in Figure 2, its HCA tree can be transferred into a clause adjacency matrix in Figure 3a by checking if two clauses have an inter-clause edge (not meaning that they are adjacent in the source sentence), where adjacent clauses have specific correlation (pink) and non-adjacent ones share no semantics (white). Especially each clause has the strongest correlation with itself (red).

Furthermore, we transform the clause adjacency matrix into token visibility matrices by splitting every clause into tokens. As shown in Figure 3b,c, the visibilities between tokens can be summarized into the following cases:

- **Full Visibility**: tokens in the same clause are fully visible mutually
- **Patial Visibility**: tokens from two adjacent clauses C_1 and C_2 are sharing partial visibility
- **None Visibility**: tokens from non-adjacent clauses C_2 and C_5 are invisible to each other

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0405.v1

- **Global Visibility**: tokens with a pronoun POS (e.g., "I" in C_5) are globally visible for linguistic phenomena of co-reference
- **Additional Visibility**: tokens that are clausal keywords (i.e., coordinators, subordinators, and antecedents) share additional visibilities with the tokens in adjacent clauses (e.g., "if" in C_1 to tokens in C_2)



(a) Clause Adjacency Matrx (b) Token Adjacency Matrx - Adjacent (c) Token Adjacency Matrx - Non-adjacent

Figure 3. Overview of our hierarchical clause annotation (HCA) -based Self Attention approach that integrates the clausal structure of input sentences. In (**a**), red blocks mean that clauses have the strongest correlation with themself; the pink/white ones mean that the corresponding two are adjacent/non-adjacent in the HCA tree. In (**b**,**c**), the adjacency between two clauses is concretized in a token visibility matrix. Pink circles with a red dotted border mean one of the two corresponding tokens is a pronoun, while those with a blue dotted border indicate the existence of a clausal keyword (i.e., coordinator, subordinator, or antecedent).

Therefore, we introduce two token visibility matrices M^{vis} and M^{deg}, where the former signals if two tokens are *visible* mutually, and the latter measures the visibility *degree* between them:

$$M^{\text{vis}}_{i,j} = \begin{cases} -\infty & w_i \otimes w_j \\ 0, & otherwise \end{cases} \tag{1}$$

$$M^{\text{deg}}_{i,j} = \begin{cases} 0 & w_i \otimes w_j \\ 1, & w_i \odot w_j \\ \lambda_0, & w_i \ominus w_j \\ \lambda_1, & w_i \ominus w_j \wedge Key(w_i/w_j) \\ \mu, & PRN(w_i/w_j) \end{cases} \tag{2}$$

where $\otimes$, $\odot$, and $\ominus$ mean that the corresponding two tokens, w_i and w_j, are positioned in two nonajacent (*None Visibility*), same (*Full Visibility*), and adjacent (*Partial Visibility*) clauses, respectively. $Key(w_i/w_j)$ indicates that at least one of w_i and w_j is a clausal keyword token, while $PRN(w_i/w_j)$ denotes the existences of at least one pronoun in w_i and w_j. Values of the hyperparameters λ_0, λ_1 and μ are in $(0, 1)$, and $\lambda_1 > \lambda_0$.

3.1.2. Masked-Self-Attention

To some degree, the token visibility matrices M^{vis} and M^{deg} contain the structural information of the HCA tree. For vanilla Transformers employed in existing AMR parsers, stacked self-attention

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0405.v1

layers inside can not receive M^{vis} and M^{deg} as inputs, so we modify it into *Masked-Self-Attention*, which can restrict the attention scope and attention scores according to M^{vis} and M^{deg}. Formally, the masked attention scores S^{mask} and the masked attention matrix A^{mask} are defined as:

$$S^{\text{mask}} = \text{Softmax}(\frac{QK^\top}{\sqrt{d}} \cdot M^{\text{deg}} + M^{\text{vis}}) \tag{3}$$

$$A^{\text{mask}} = S^{\text{mask}} V \tag{4}$$

where self-attention inputs $Q, K, V \in \mathbb{R}^{N \times d}$, N is the length of input sentences, and scaling factor d is the dimension of the model. Intuitively, if token w_i is invisible to w_j, the attention scores $S^{\text{mask}}_{i,j}$ will be masked to 0 due to the value $-\infty$ of $M^{\text{vis}}_{i,j}$ and the value 0 of $M^{\text{deg}}_{i,j}$. Otherwise, $S^{\text{mask}}_{i,j}$ will be scaled according to $M^{\text{deg}}_{i,j}$ in different cases.

3.1.3. Clause-Relation-Binded Attention Head

In every stacked self-attention layer of Transformer, multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions [10]. It also provides us the possibility of integrating different inter-clause relations in different attention heads. Instead of masking the attention matrix with non-labeled HCA trees, we propose a clause-relation-binded attention head setting, where every head attends more to a specific inter-clause relation.

In this setting, we increase the visibilities between tokens in two adjacent clauses with interrelation rel_i in the attention matrix $\text{A}^{\text{mask}}_{\text{rel}_i}$ of the binded head, i.e., increasing λ_0 to 1 in Equation (2). Therefore, the final attention matrix $A^{\text{MultiHead}}$ of each stacked self-attention layer is defined as:

$$A^{\text{MultiHead}} = \text{Concat}(\text{A}^{\text{mask}}_{\text{rel}_1}, \ldots, \text{A}^{\text{mask}}_{\text{rel}_i}, \ldots, \text{A}^{\text{mask}}_{\text{rel}_h}) W^O \tag{5}$$

where the parameter matrix $W^O \in \mathbb{R}^{hN \times d}$ and h is the attention head number mapping to 16 inter-clause relations in HCA [24].

3.2. HCA-Based Curriculum Learning

Inspired by the idea of curriculum learning [52] that humans handle difficult tasks from easy examples to hard ones, we propose an HCA-CL approach for training an AMR parser, where the clause number and the tree depth of a sentence's HCA are the measurements of learning difficulty. Referring to the previous work of Wang et al. [18], we set two learning curricula, *Clause-Number* and *Tree-Depth*, in our HCA-CL approach demonstrated in Figure 4.

3.2.1. Clause-Number Curriculum

In *Clause-Number* (CN) curriculum, sentences with more clauses that involve more inter-clause relations and longer dependency distances (demonstrated in Figure 1) are considered to be harder to learn. With this intuition, all training sentences are divided into N buckets according to their clause number $\{C_i : i = 1, \ldots, N\}$, where C_i contains sentences with the clause number i. In the training epoch of the *Clause-Number* curriculum, there are N training episodes with T_{cn} steps individually. In each step of the i-th episode, the training scheduler samples a batch of examples from buckets $\{I_j : j \leq i\}$ to train the model.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0405.v1

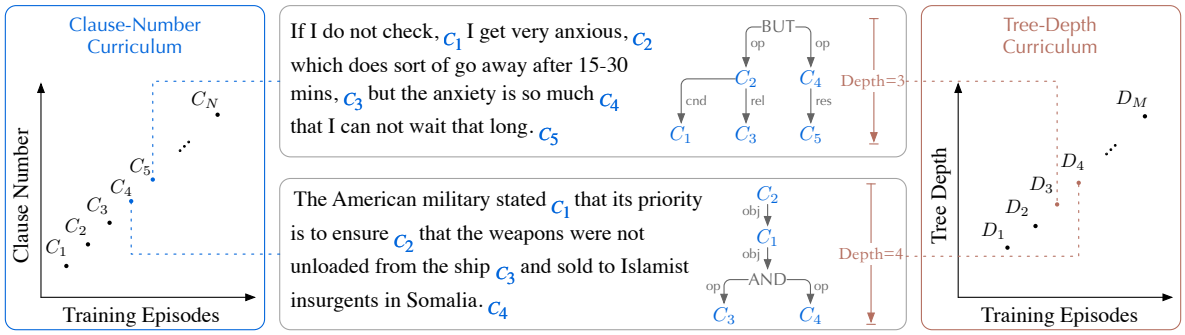


Figure 4. Overview of our hierarchical clause annotation (HCA) -based curriculum learning approach with two curricula, *Clause-Number* and *Tree-Depth*. The learning difficulties of the two curricula are set by the clause number and the tree depth of a sentence's HCA in the left and right charts. Two exemplified sentences from AMR 2.0 and their HCAs are demonstrated in the middle.

3.2.2. Tree-Depth Curriculum

In *Tree-Depth* (TD) curriculum, sentences with deeper HCA trees that correspond to deeper hierarchical AMR graphs are considered to be harder to learn. With this intuition, all training sentences are divided into M buckets according to their clause number $\{D_i : i = 1, \ldots, M\}$, where D_i contains sentences with the clause number i. In the training epoch of the *Clause-Number* curriculum, there are M training episodes with T_{td} steps individually. In each step of the i-th episode, the training scheduler samples a batch of examples from buckets $\{I_j : j \leq i\}$ to train the model.

4. Experiments

In this section, we describe the details of datasets, environments, model hyperparameters, evaluation metrics, compared models, and parsing results in the experiments.

4.1. Datasets

For the benchmark datasets, we choose two standard AMR datasets, AMR 2.0 and AMR 3.0, as the ID settings and three test sets, TLP, New3, and Bio, as the OOD settings.

For the HCA tree of each sentence, we use the manual-annotated HCA trees for AMR 2.0 provided by [24], and auto-annotated HCA trees for the rest datasets, which are all generated by the HCA Segmenter and the HCA Parser proposed by [24].

4.1.1. In-Distribution Datasets

We first train and evaluate our HCA-based parser on two standard AMR parsing evaluation benchmarks:

- *AMR 2.0*: includes 39,260 sentence-AMR pairs in which source sentences are collected for the DARPA BOLT AND DEFT programs, transcripts and English translations of Mandarin Chinese broadcast news programming from China Central TV, Wall Street Journal text, translated Xinhua news texts, various newswire data from NIST OpenMT evaluations and weblog data used in the DARPA GALE program.
- *AMR 3.0*: is a superset of AMR 2.0 and enriches the data instances to 59,255. New source data added to AMR 3.0 includes sentences from Aesop's Fables, parallel text and the situation frame data set developed by LDC for the DARPA LORELEI program, and lead sentences from Wikipedia articles about named entities.

The training, development, and test sets in both datasets are a random split, and therefore we take them as ID datasets like previous works [15,17,18,20,42].

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0405.v1

4.1.2. Out-of-Distribution Datasets

To further estimate the effects of our HCA-based approaches on open-world data that comes from a different distribution, we follow the OOD settings introduced by [15], and predict on three OOD test sets with the parser trained on the AMR 2.0 training set:

- *New3* [4]: a set of 527 instances from AMR 3.0, whose source was the LORELEI DARPA project – not included in the AMR 2.0 training set – consisting of excerpts from newswire and online forums
- *TLP* [5]: the full AMR-tagged children's novel, The Little Prince (version 3.0), consisting of 1,562 pairs
- *Bio* [6]: the test set of the Bio-AMR corpus, consisting of 500 instances, featuring biomedical texts [53]

4.1.3. Hierarchical Clause Annotations

For the hierarchical clausal features utilized in our HCA-based approaches, we use the manual-annotated HCA corpus for AMR 2.0 provided in [24]. Moreover, we employ the HCA-segmenter and the HCA-parser proposed in [24] to generate silver HCA trees for AMR 3.0 and three OOD test sets. Detailed statistics of the evaluation datasets in this paper are listed in Table 1.

Table 1. Main statistics of five AMR parsing benchmarks. "ID" and "OOD" denote in-distribution and out-of-distribution settings, respectively. "#Snt." and "#HCA" represent the total number of sentences and complex sentences with hierarchical clause annotations in each split set.

	Dataset	Training		Development		Test	
		#Snt.	#HCA	#Snt.	#HCA	#Snt.	#HCA
ID	AMR 2.0	36,521	17,886	1,368	741	1,371	753
	AMR 3.0	55,635	36,921	1,722	1,243	1,898	1,258
OOD	New3	-		-		527	286
	TLP	-		-		1,562	825
	Bio	-		-		500	367

4.2. Baseline and Compared Models

We compare our HCA-based AMR parser with several recent parsers:

- *AMR-gs (2020)* [14], a graph-based parser that enhances incremental graph construction with an AMR graph↔sequence (AMR-gs) iterative inference mechanism in one-stage procedures.
- *APT (2021)* [41], a transition-based parser that employs an action-pointer Transformer (APT) to decouple source tokens from node representations and address alignments.
- *StructBART (2021)* [42], a transition-based parser that integrates the pre-trained language model, BART, for structured fine-tuning.
- *SPRING (2021)* [15], a fine-tuned BART model that predicts a linearized AMR graph.
- *HCL (2022)* [18], a hierarchical curriculum learning (HCL) framework that helps the seq2seq model adapts to the AMR hierarchy.
- *ANCES (2022)* [17], a seq2seq-based parser that adds the important ancestor (ANCES) information into the Transformer decoder.

---

[4] Located at AMR 3.0/data/amrs/split/test/amr-release-3.0-amrs-test-lorelei.txt
[5] https://amr.isi.edu/download/amr-bank-struct-v3.0.txt
[6] https://amr.isi.edu/download/2016-03-14/amr-release-test-bio.txt

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0405.v1

- *HGAN (2022)* [20], a seq2seq-based parser that applies a heterogeneous graph attention network (HGAN) to argument word representations with syntactic dependencies and semantic role labelings of input sentences. It is also the current SOTA parser in the settings of removing graph re-categorization, extra silver training data, and ensemble methods.

Since *SPRING* provides a clear and efficient seq2seq-based architecture based on a vanilla BART, recent seq2seq-based models, *HCA*, *ANCES*, and *HGAN*, all select it as the codebase. Therefore, we also choose *SRPING* as the baseline model to apply our HCA-based approaches. Besides, we do not take the competitive AMR parser, *ATP* [19], into our compared models since it employs syntactic dependency parsing and semantic role labeling as intermediate tasks to introduce extra silver training data.

4.3. Hyper-Parameters

For the hyper-parameters of our HCA-based approaches, we list their layer, name, and value in Table 2. All models are trained until reaching the maximum epochs, and then select the best model checkpoint on the development set.

Table 2. Final hyper-parameters configuration of clause segmentation model. "HCA-SA Encoder" indicates the HCA-based Self Attention approach user in the encoder, and "HCA-CL Strategy" represents the HCA-based Curriculum Learning approach used before the normal training epochs.

Layer	Hyper-Parameter	Value
Word Embedding	Bart-large	1,024
HCA-SA Encoder	layer	12
	head	16
	λ_0	0.5
	λ_1	0.8
	μ	1
Decoder	layer	12
	head	16
HCA-CL Strategy	T_{cn}	500
	T_{td}	1,500
Trainer	optimizer	RAdam
	weight decay	4e-3
	loss function	Cross-entropy
	learning rate	5e-5
	batch size	500
	dropout	0.25
	maximum epochs	30
Prediction	beam size	5

4.4. Evaluation Metrics

Following previous AMR parsing works, we use the Smatch scores [54] and the fine-grained metrics [55] to evaluate the performances. Specifically, the fine-grained AMR metrics are:

1. Unlabeled (*Unlab.*): Smatch computed on the predicted graphs after removing all edge labels.
2. No WSD. (*NoWSD*): Smatch while ignoring Propbank senses (e.g., "go-01" vs "go-02").
3. Named entity (*NER*): F-score on the named entity recognition (`:name` roles).
4. Wikification (*Wiki.*): F-score on the wikification (`:wiki` roles).
5. Negation (*Neg.*): F-score on the negation detection (`:polarity` roles).
6. Concepts (*Conc.*): F-score on the concept identification task.
7. Reentrancy (*Reent.*): Smatch computed on reentrant edges only, e.g., the edges of node "I" in Figure 1.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0405.v1

8. Semantic role labelings (*SRL*): Smatch computed on `:ARGi` roles only.

As suggested in [18], *Unlab.*, *Reent*, and *SRL* are as structure-dependent metrics, since

- *Unlab.* does not consider any edge labels and only considers the graph structure.
- *Reent.* is a typical structure feature for the AMR graph. Without reentrant edges, the AMR graph is reduced to a tree.
- *SRL* denotes the core-semantic relation of the AMR, which determines the core structure of the AMR.

Conversely, all other metrics are classified as structure-independent metrics.

4.5. Experimental Environments

Table 3 lists the information on the main hardware and software used in our experimental environments. Note that the model on AMR 2.0 is trained in a total of 30 epochs for 16 hours, while the model trained on AMR 3.0 is finished in a total of 30 epochs for 28 hours with the experimental environments.

Table 3. Hardware and software used in our experiments.

Environment	Value
Hardware	
CPU	Intel(R) Xeon(R) Silver 4216 CPU @ 2.10GHz
GPU	NVIDIA RTX 2080Ti (11G)
Memory	64 GB
Software	
Python	3.8.16
Pytorch	1.13.0
Anaconda	4.10.1
CUDA	11.0
IDE	PyCharm 2022.2.3

4.6. Experimental Results

We now report the AMR parsing performances of our HCA-based parser and other compared parsers on ID datasets and OOD datasets, respectively.

4.6.1. Results in ID datasets

As demonstrated in Table 4, we report AMR parsing performances of the baseline model (*SPRING*), other compared parsers, and the modified *SRPING* that applies our HCA-based Self-Attention (*HCA-SA*) and Curriculum Learning (*HCA-CL*) approaches on ID datasets, AMR 2.0 and AMR 3.0. All the results of our HCA-based model have averaged scores of five experimental trials, and we compute the significance of performance differences using the non-parametric approximate randomization test [56]. From the results, we have the following observations:

- Equipped with our *HCA-SA* and *HCA-CL* approaches, the baseline model *SPRING* achieves 0.7 Smatch F1 score improvements on both AMR 2.0 and AMR 3.0. The improvements are significant with $p < 0.005$ and $p < 0.001$, respectively.
- In AMR 2.0, our HCA-based model outperforms all compared models except *ANCES* and the *HGAN* version that introduces both DP and SRL features.
- In AMR 3.0, consisting of more sentences with HCA trees, the performance gap between our HCA-based parser and the SOTA (*HGAN* with DP and SRL) is only 0.2 Smatch F1 scores.

To better analyze how the performance improvements of the baseline model are achieved when applying our HCA-based approaches, we also report structure-dependent fine-grained results in

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0405.v1

Table 4. As claimed in Section 1, inter-clause relations in the HCA can bring LDD issues, which are typically related to AMR concept nodes aligned with verb phrases, and reflected in structure-dependent metrics. As can be observed:

- Our HCA-based model outperforms the baseline model in nearly all fine-grained metrics, especially in structure-dependent metrics with 1.1, 1.8, and 3.9 F1 scores improvements in *Unlab.*, *Reent.*, and *SRL*, respectively.
- In the *SRL*, *Conc.*, and *Neg* metrics, our HCA-based model achieves the best performances against all compared models.

Table 4. Smatch and fine-grained F1 scores (%) of our AMR parser and compared ones on two in-distribution (ID) evaluation test sets. The column "Feat." means the extra features that an AMR parser requires, where "DP", "SRL", and "HCA" indicate syntactic dependencies, semantic role labelings, and hierarchical clause annotations, respectively. For comparison fairness, the performances of compared parsers are the versions without graph re-categorization, extra silver training data, and ensemble methods. The best result per measure across each test set is shown in bold, while that in baseline model (*SRPING*) and ours is underlined. "w/o" denotes "without".

	Model	Feat.	Smatch	Structure-Dependent			Structure-Independent				
				Unlab.	Reent.	SRL	NoWSD	Conc.	Wiki.	NER	Neg.
AMR 2.0	AMR-gs (2020) [14]	-	78.7	81.5	63.8	74.5	79.2	88.1	81.3	87.1	66.1
	APT (2021) [41]	-	81.7	85.5	71.1	80.8	82.3	88.7	78.8	88.5	69.7
	StructBART (2021) [42]	-	84.3	87.9	74.3	-	-	-	-	-	-
	HCL (2022) [18]	-	84.3	87.7	74.5	83.2	85.0	90.2	84.0	91.6	**75.9**
	ANCES (2022) [17]	-	84.8	**88.1**	**75.1**	83.4	85.3	90.5	84.1	91.8	74.0
	HGAN (2022) [20]	DP	84.4	-	-	-	-	-	-	-	-
	HGAN (2022) [20]	DP,SRL	**84.9**	87.8	73.9	83.0	**85.5**	**90.8**	**84.6**	**91.9**	74.7
	SPRING (2021) [15]	-	83.8	86.1	70.8	79.6	84.4	90.2	84.3	90.6	74.4
	Ours	HCA	84.5	87.0	72.5	**83.5**	84.5	90.7	84.4	91.2	75.2
AMR 3.0	AMR-gs (2020) [14]	-	78.0	81.9	63.7	73.2	78.5	88.5	75.7	83.7	68.9
	APT (2021) [41]	-	80.3	-	-	-	-	-	-	-	-
	StructBART (2021) [42]	-	83.2	-	-	-	-	-	-	-	-
	HCL (2022) [18]	-	83.7	**86.9**	**73.9**	82.4	84.2	89.5	82.6	89.0	73.0
	ANCES (2022) [17]	-	83.5	86.6	74.2	82.2	84.0	89.5	81.5	88.9	72.6
	HGAN (2022) [20]	DP	83.5	-	-	-	-	-	-	-	-
	HGAN (2022) [20]	DP,SRL	**83.9**	86.5	73.0	82.2	**84.3**	90.2	**83.0**	**89.2**	73.2
	SPRING (2021) [15]	-	83.0	85.4	70.4	78.9	83.5	89.8	82.7	87.2	73.0
	Ours	HCA	83.7	86.6	72.2	**82.8**	83.4	**90.5**	82.6	88.0	**73.8**

4.6.2. Results in OOD Datasets

As demonstrated in Table 5, we report the parsing performances of our HCA-based model and compared models on three OOD datasets. As can be seen,

- Our HCA-based model outperforms the baseline model *SPRING* with 2.5, 0.7, and 3.1 Smatch F1 score improvements in New3, TLP, and Bio test sets, respectively.
- In the New3 and Bio datasets that contain long sentences of newswire and biomedical texts and have more HCA trees, our HCA-based model outperforms all compared models.
- In the TLP dataset that contains many simple sentences of a children's story and fewer HCA trees, our HCA-based does not perform as well as *HCL* and *HGAN*.

Table 5. Smatch F1 scores (%) of our HCA-based model and compared models on out-of-distribution (OOD) datasets. The best result on each test set is shown in bold.

	New3	TLP	Bio
SPRING (2022) [15]	73.7	77.3	59.7
HCL (2022) [18]	75.3	78.2	61.1
HGAN (2022) [20]	**76.0**	**79.2**	61.6
Ours	**76.0**	78.0	**62.3**

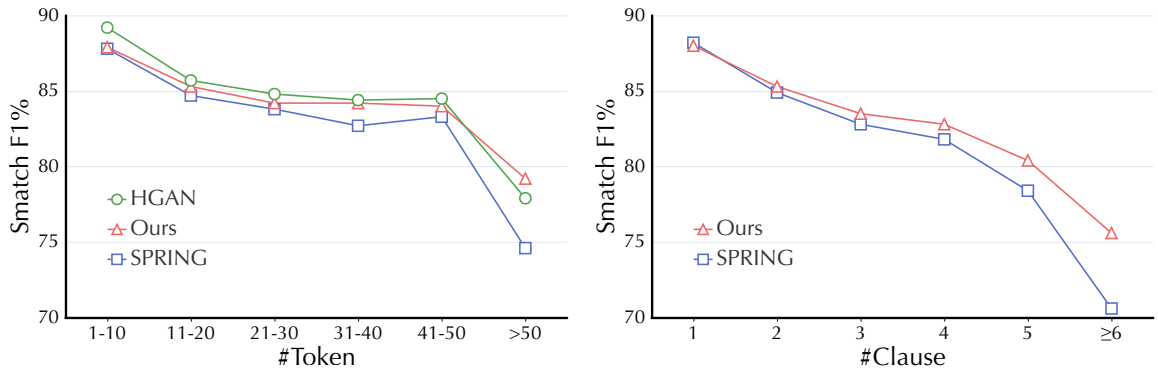
Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0405.v1

5. Discussion

As shown in the previous section, our HCA-based model achieves prominent improvements against the baseline model, *SPRING*, and outperforms other compared models, including the SOTA model, *HGAN* in some fine-grained metrics in ID and ODD datasets. In this section, we further discuss the paper's main issue of whether our HCA-based approaches have effects in curing LDDs. Additionally, the ablation studies and the case studies are also provided.

5.1. Effects on Long-Distance Dependencies in ID datasets

As claimed in Section 1, most LDD cases occur in sentences with complex hierarchical clause structures. In Figure 5, we demonstrate the parsing performances trends of the baseline model *SPRING*, the SOTA parser *HGAN*[7], and our HCA-based model over the number of tokens and clauses in sentences from AMR 2.0. As can be observed:



(a) Performance trends over the number of tokens

(b) Performance trends over the number of clauses

Figure 5. Performances trends of *SPRING*, *HGAN*, and *Ours* in the AMR 2.0 dataset over the number of tokens (denoted as "#Token") and clauses (denoted as "#Clause") inside a sentence.

- When the number of tokens (denoted as #*Token* for simplicity) >20 in a sentence, the performance boosts of our HCA-based model against the baseline *SPRING* gradually become significant.
- For the case #*Token*>50 that indicates sentences with many clauses and inter-clause relations, our HCA-based model outperforms both *SPRING* and *HGAN*.
- When compared on performances trends over #*Clause*, the performance lead of our HCA-based model against *SPRING* becomes much more evident as #*Clause* increases.

To sum up, our HCA-based approaches show significant effectiveness on long sentences with complex clausal structures that introduce most LDD cases.

5.2. Effects on Long-Distance Dependencies in OOD datasets

As the performance improvements achieved by our HCA-based approaches are much more prominent on OOD datasets than on ID datasets, we have further exploration of the OOD datasets with different characteristics.

Figure 6 demonstrates the two main statistics of three OOD datasets, i.e., the average number of clauses per sentence (denoted as $\overline{\#C/S}$) and the average number of tokens per clause ($\overline{\#T/C}$). These two statistics of datasets both characterize the complexity of the clausal structure inside a sentence, where

[7] We only use the original data published in their paper to draw performance trends over the number tokens, without performances in terms of the number of clauses.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0405.v1

- $\overline{\#C/S}$ shows the number of complex sentences with more than one clauses
- $\overline{\#T/C}$ depicts the latent dependency distance between two tokens from different clauses

We also present the performance boosts of our HCA-based parser against *SPRING* in Figure 6. As can be observed, the higher values of $\overline{\#C/S}$ and $\overline{\#C/S}$ in an OOD dataset, the higher Smatch improvements are achieved by our HCA-based approaches. Specifically, New3 and Bio seem to cover more complex texts from newswire and biomedical articles, while TLP contains simpler sentences that are easy for children to read. Therefore, our AMR parser performs much better on complex sentences from Bio and New3, indicating the effectiveness of our HCA-based approaches on LDDs.

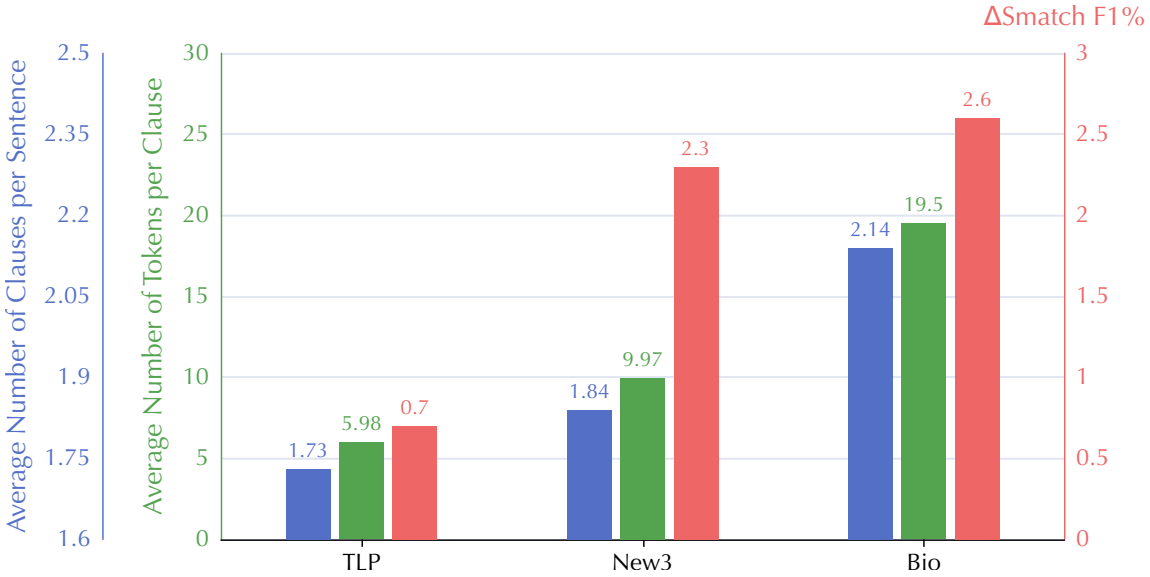


Figure 6. Two important characteristics of three different out-of-distribution (OOD) test sets (i.e., TLP, New3, and Bio) and performance boosts of our HCA-based parser on each test sets. The blue and green statistics of each dataset represent the average number of clauses per sentence and that of tokens per clause, respectively. The red statistics show the improvements of our HCA-based model against the baseline model, *SPRING*, on each OOD dataset.

5.3. Ablation Study

In the HCA-SA approach, two token visibility matrices derived from HCA trees are introduced to mask certain attention heads. Additionally, we propose a clause-relation-binded attention head setting to integrate inter-clause relations in the encoder. Therefore, we conduct ablation studies by introducing random token visibility matrices (denoted as "w/o VisMask") and removing the clause-relation-binded attention setting (denoted as "w/o ClauRel"). Note that "w/o VisMask" contains the case of "w/o ClauRel" because the clause-relation-binded attention setting is based on the masked-self-attention mechanism.

In the HCA-CL approach, extra training epochs for *Clause-Number* and *Tree-Depth* curricula serve as a warm-up stage for the subsequent training process. To eliminate the effect of the extra epochs, we also add the same number of training epochs in the ablation study of our HCA-CL approach.

As shown in Table 6:

- In HCA-SA, the clause-relation-binded attention setting (demoted as "ClauRel") contributes most in the *SRL* metric due to the mappings between inter-clause relations (e.g., *Subjective* and *Objective*) and SRL-type AMR relations (e.g., `:ARG0` and `:ARG1`).
- In HCA-SA, the masked-self-attention mechanism (demoted as "VisMask") achieves significant improvements in the *Reent.* metric by increasing the visibility of pronoun tokens to all tokens.
- In HCA-CL, the *Tree-Depth* curriculum (demoted as "TD") has no effects on the parsing performances. We conjecture that sentences with much deeper clausal structures are rare, and

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 4 August 2023 doi:10.20944/preprints202308.0405.v1

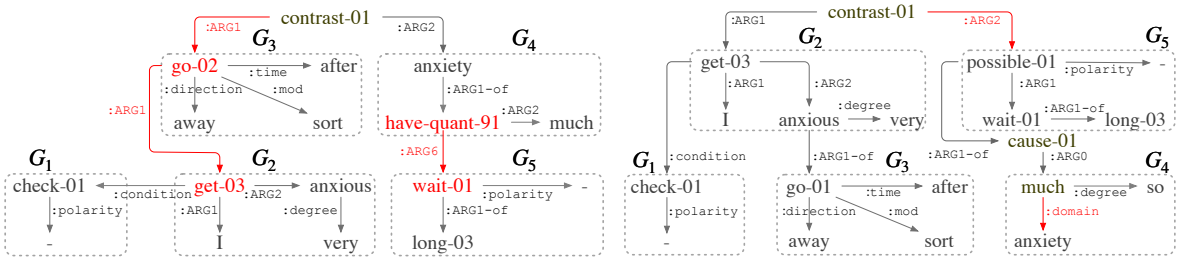
the number of split buckets for the depth of clausal trees is not big enough to distinguish the training sentences.

Table 6. F1 scores (%) of Smatch and three structure-dependent metrics achieved by our HCA-based models in ablation studies on AMR 2.0. "w/o" denotes "without". "VisMask" and "ClauRel" indicate "token visibilities matrices" and the "clause-relation-binded attention head setting" in the HCA-based self-attention (HCA-SA) approach. "CN" and "TD" represent the *Clause-Number* and *Tree-Depth* curricula in the HCA-based curriculum learning (HCA-CL) approach.

Model		Smatch	Unlab.	Reent.	SRL
SPRING (2021) [15]		83.8	86.1	70.8	79.6
Ours	Full	84.5	87.0	72.5	83.5
	w/o VisMask	84.1	86.5	70.9	81.2
	w/o ClauRel	84.4	86.8	72.4	81.5
	w/o CN	84.2	86.7	72.4	83.4
	w/o TD	84.5	87.0	72.5	83.4
	w/o CN,TD	84.2	86.7	72.4	83.4

5.4. Case Study

To further demonstrate the effectiveness of our HCA-based approaches on LDDs in AMR parsing, we compare the output AMR graphs of the same sentence exemplified in Figure 1, parsed by the baseline model *SPRING* and the modified *SPRING* that applies our HCA-SA and HCA-CL approaches (denoted as *Ours*), respectively in Figure 7.



(a) SPRING (b) Ours

Figure 7. Parsing results of the baseline model *SPRING* and the modified *SPRING* that applies our HCA-based approaches (denoted as *Ours*) when encountering the same AMR 2.0 sentence in Section 1. AMR nodes and edges in red are parsing errors compared to the gold AMR graph. Extra nodes and edges, which are correctly parsed by both, are omitted.

For *SPRING*, it mislabels node "go-02" in subgraph G_3 as the `:ARG1` role of node "contrast-01". Then it fails to realize that it is "anxious" in G_2 that takes the `:ARG1` role of "go-02" in G_3. Additionally, the causality between G_4 and G_5 is not interpreted correctly due to the absence of node "cause-01" and its arguments.

In contrast, when integrating the HCA, *Ours* seems to understand the inter-clause relations better. Although "possible-01" in subgraph G_5 is mislabeled as the `:ARG2` role of node "contrast-01", it succeeds in avoiding errors made by *SPRING*. Another mistake in *Ours* is that the relation `:quant` between "much" and "anxiety" is reversed and replaced by `:domain`, which impacts little on Smatch F1 scores. The vast performance gap between *SPRING* and our HCA-based *SPRING* in Smatch F1 scores% (66.8 vs. 88.7) also proves the effectiveness of the HCA on LDDs in AMR parsing.

6. Conclusion

We propose two HCA-based approaches, HCA-SA and HCA-CL, to integrate HCA trees of complex sentences for addressing LDDs in AMR parsing. Taking AMR parsing as an example, we apply our HCA-based framework to a popular AMR parser *SPRING* to integrate HCA features in the encoder. In the evaluations on ID datasets, our parser achieves prominent and explainable improvements against the baseline model, *SPRING*, and outperforms the SOTA parser, *HGAN*, in some fine-grained metrics. Notably, as the clause number of the sentence increases, our parser outperforms *SPRING* by a large margin and achieves better Smatch F1 scores than *HGAN*, indicating the ability to cure LDDs. In the evaluations on OOD datasets, the performance boosts achieved by our HCA-based approaches are more evident on complicated corpora like New3 and Bio, where sentences consist of more clauses and longer clauses.

Author Contributions: Conceptualization, Y.F. and Z.G.; Methodology, Y.F.; Software, Y.F. and Y.S.; Writing—original draft, Y.F.; Writing—review and editing, B.L., M.G., Y.S., Q.C., and Z.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets used in our experiments are publicly available: AMR 2.0 at https://catalog.ldc.upenn.edu/LDC2017T10 (accessed on 11 June 2022), AMR 3.0 and its subset New3 at https://catalog.ldc.upenn.edu/LDC2020T02 (accessed on 11 August 2022), TLP at https://amr.isi.edu/download/amr-bank-struct-v3.0.txt (accessed on 12 October 2022), Bio at https://amr.isi.edu/download/2016-03-14/amr-release-test-bio.txt (accessed on 15 October 2022), and the HCA features of all datasets at https://github.com/MetroVancloud/HCA-AMRparsing (accessed on 3 August 2023).

Acknowledgments: Our code extends the GitHub repositoriy *SPRING* at https://github.com/SapienzaNLP/spring (accessed on 1 March 2022), thanks to them very much.

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper

Abbreviations

The following abbreviations are used in this manuscript:

AMR	Abstract Meaning Representation
CL	Curriculum Learning
HCA	Hierarchical Clause Annotation
ID	In-Distribution
IGNN	Implicit Graph Neural Network
LDD	Long-Distance Dependency
LSTM	Long Short-Term Memory
NLP	Natural Language Processing
OOD	Out-of-Distribution
POS	Part-of-Speech
RST	Rhetorical Structure Theory
SA	Self Attention
SDP	Semantic Dependency
SRL	Semantic Role Labeling
SOTA	State-of-the-Art

References

1. Li, Z.; Cai, J.; He, S.; Zhao, H. Seq2seq Dependency Parsing. In Proceedings of the Proceedings of the 27th International Conference on Computational Linguistics; Association for Computational Linguistics: Santa Fe, New Mexico, USA, 2018; pp. 3203–3214.
2. Tian, Y.; Song, Y.; Xia, F.; Zhang, T. Improving Constituency Parsing with Span Attention. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2020; Association for Computational Linguistics: Online, 2020; pp. 1691–1703. https://doi.org/10.18653/v1/2020.findings-emnlp.153.
3. He, L.; Lee, K.; Lewis, M.; Zettlemoyer, L. Deep Semantic Role Labeling: What Works and What's Next. In Proceedings of the Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers); Association for Computational Linguistics: Vancouver, Canada, 2017; pp. 473–483. https://doi.org/10.18653/v1/P17-1044.
4. Tang, G.; Müller, M.; Rios, A.; Sennrich, R. Why Self-Attention? A Targeted Evaluation of Neural Machine Translation Architectures. In Proceedings of the Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing; Association for Computational Linguistics: Brussels, Belgium, 2018; pp. 4263–4272. https://doi.org/10.18653/v1/D18-1458.
5. Jia, Y.; Ye, Y.; Feng, Y.; Lai, Y.; Yan, R.; Zhao, D. Modeling discourse cohesion for discourse parsing via memory network. In Proceedings of the Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers); Association for Computational Linguistics: Melbourne, Australia, 2018; pp. 438–443. https://doi.org/10.18653/v1/P18-2070.
6. Xu, J.; Gan, Z.; Cheng, Y.; Liu, J. Discourse-Aware Neural Extractive Text Summarization. In Proceedings of the Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics; Association for Computational Linguistics: Online, 2020; pp. 5021–5031. https://doi.org/10.18653/v1/2020.acl-main.451.
7. Hihi, S.; Bengio, Y. Hierarchical Recurrent Neural Networks for Long-Term Dependencies. In Proceedings of the Advances in Neural Information Processing Systems. MIT Press, 1995, Vol. 8.
8. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural computation* **1997**, *9*, 1735–1780.
9. Salton, G.; Ross, R.; Kelleher, J. Attentive Language Models. In Proceedings of the Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers); Asian Federation of Natural Language Processing: Taipei, Taiwan, 2017; pp. 441–450.
10. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.u.; Polosukhin, I. Attention is All you Need. In Proceedings of the Advances in Neural Information Processing Systems. Curran Associates, Inc., 2017, Vol. 30.
11. Gu, F.; Chang, H.; Zhu, W.; Sojoudi, S.; El Ghaoui, L. Implicit Graph Neural Networks. In Proceedings of the Advances in Neural Information Processing Systems. Curran Associates, Inc., 2020, Vol. 33, pp. 11984–11995.
12. Banarescu, L.; Bonial, C.; Cai, S.; Georgescu, M.; Griffitt, K.; Hermjakob, U.; Knight, K.; Koehn, P.; Palmer, M.; Schneider, N. Abstract Meaning Representation for Sembanking. In Proceedings of the Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse; Association for Computational Linguistics: Sofia, Bulgaria, 2013; pp. 178–186.
13. Peng, X.; Gildea, D.; Satta, G. AMR Parsing With Cache Transition Systems. *Proceedings of the AAAI Conference on Artificial Intelligence* **2018**, *32*. https://doi.org/10.1609/aaai.v32i1.11922.
14. Cai, D.; Lam, W. AMR Parsing via Graph-Sequence Iterative Inference. In Proceedings of the Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics; Association for Computational Linguistics: Online, 2020; pp. 1290–1301. https://doi.org/10.18653/v1/2020.acl-main.119.
15. Bevilacqua, M.; Blloshmi, R.; Navigli, R. One SPRING to rule them both: Symmetric AMR semantic parsing and generation without a complex pipeline. In Proceedings of the Proceedings of the AAAI Conference on Artificial Intelligence, 2021, Vol. 35, pp. 12564–12573.
16. Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; Zettlemoyer, L. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In Proceedings of the Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics; Association for Computational Linguistics: Online, 2020; pp. 7871–7880. https://doi.org/10.18653/v1/2020.acl-main.703.
17. Yu, C.; Gildea, D. Sequence-to-sequence AMR Parsing with Ancestor Information. In Proceedings of the Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume

2: Short Papers); Association for Computational Linguistics: Dublin, Ireland, 2022; pp. 571–577. https://doi.org/10.18653/v1/2022.acl-short.63.

18. Wang, P.; Chen, L.; Liu, T.; Dai, D.; Cao, Y.; Chang, B.; Sui, Z. Hierarchical Curriculum Learning for AMR Parsing. In Proceedings of the Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers); Association for Computational Linguistics: Dublin, Ireland, 2022; pp. 333–339. https://doi.org/10.18653/v1/2022.acl-short.37.
19. Chen, L.; Wang, P.; Xu, R.; Liu, T.; Sui, Z.; Chang, B. ATP: AMRize Then Parse! Enhancing AMR Parsing with PseudoAMRs. In Proceedings of the Findings of the Association for Computational Linguistics: NAACL 2022; Association for Computational Linguistics: Seattle, United States, 2022; pp. 2482–2496. https://doi.org/10.18653/v1/2022.findings-naacl.190.
20. Sataer, Y.; Shi, C.; Gao, M.; Fan, Y.; Li, B.; Gao, Z. Integrating Syntactic and Semantic Knowledge in AMR Parsing with Heterogeneous Graph Attention Network. In Proceedings of the ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2023, pp. 1–5. https://doi.org/10.1109/ICASSP49357.2023.10097098.
21. Carter, R.; McCarthy, M. *Cambridge grammar of English: a comprehensive guide; spoken and written English grammar and usage*; Cambridge University Press, 2006.
22. Liu, Y.; Ryskin, R.; Futrell, R.; Gibson, E. A verb-frame frequency account of constraints on long-distance dependencies in English. *Cognition* **2022**, *222*, 104902. https://doi.org/https://doi.org/10.1016/j.cognition.2021.104902.
23. Mann, W.C.; Thompson, S.A. Rhetorical structure theory: Toward a functional theory of text organization. *Text-interdisciplinary Journal for the Study of Discourse* **1988**, *8*, 243–281.
24. Fan, Y.; Li, B.; Sataer, Y.; Gao, M.; Shi, C.; Cao, S.; Gao, Z. Hierarchical Clause Annotation: Building a Clause-Level Corpus for Semantic Parsing with Complex Sentences. *Preprints.org* **2023**. https://doi.org/10.20944/preprints202306.0530.v1.
25. Hockett, C.F. A formal statement of morphemic analysis. *Studies in Linguistics* **1952**, *10*, J39.
26. Mahalunkar, A.; Kelleher, J.D. Understanding Recurrent Neural Architectures by Analyzing and Synthesizing Long Distance Dependencies in Benchmark Sequential Datasets, 2020, [arXiv:cs.LG/1810.02966].
27. Szubert, I.; Damonte, M.; Cohen, S.B.; Steedman, M. The Role of Reentrancies in Abstract Meaning Representation Parsing. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2020; Association for Computational Linguistics: Online, 2020; pp. 2198–2207. https://doi.org/10.18653/v1/2020.findings-emnlp.199.
28. Flanigan, J.; Thomson, S.; Carbonell, J.; Dyer, C.; Smith, N.A. A Discriminative Graph-Based Parser for the Abstract Meaning Representation. In Proceedings of the Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers); Association for Computational Linguistics: Baltimore, Maryland, 2014; pp. 1426–1436. https://doi.org/10.3115/v1/P14-1134.
29. Lyu, C.; Titov, I. AMR Parsing as Graph Prediction with Latent Alignment. In Proceedings of the Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers); Association for Computational Linguistics: Melbourne, Australia, 2018; pp. 397–407. https://doi.org/10.18653/v1/P18-1037.
30. Zhang, S.; Ma, X.; Duh, K.; Van Durme, B. Broad-Coverage Semantic Parsing as Transduction. In Proceedings of the Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP); Association for Computational Linguistics: Hong Kong, China, 2019; pp. 3786–3798. https://doi.org/10.18653/v1/D19-1392.
31. Zhang, S.; Ma, X.; Duh, K.; Van Durme, B. AMR Parsing as Sequence-to-Graph Transduction. In Proceedings of the Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics; Association for Computational Linguistics: Florence, Italy, 2019; pp. 80–94. https://doi.org/10.18653/v1/P19-1009.
32. Konstas, I.; Iyer, S.; Yatskar, M.; Choi, Y.; Zettlemoyer, L. Neural AMR: Sequence-to-Sequence Models for Parsing and Generation. In Proceedings of the Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers); Association for Computational Linguistics: Vancouver, Canada, 2017; pp. 146–157. https://doi.org/10.18653/v1/P17-1014.
33. Peng, X.; Song, L.; Gildea, D.; Satta, G. Sequence-to-sequence Models for Cache Transition Systems. In Proceedings of the Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics

(Volume 1: Long Papers); Association for Computational Linguistics: Melbourne, Australia, 2018; pp. 1842–1852. https://doi.org/10.18653/v1/P18-1171.

34. Ge, D.; Li, J.; Zhu, M.; Li, S. Modeling Source Syntax and Semantics for Neural AMR Parsing. In Proceedings of the Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19. International Joint Conferences on Artificial Intelligence Organization, 2019, pp. 4975–4981. https://doi.org/10.24963/ijcai.2019/691.
35. Xu, D.; Li, J.; Zhu, M.; Zhang, M.; Zhou, G. Improving AMR Parsing with Sequence-to-Sequence Pre-training. In Proceedings of the Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP); Association for Computational Linguistics: Online, 2020; pp. 2501–2511. https://doi.org/10.18653/v1/2020.emnlp-main.196.
36. Wang, C.; Xue, N.; Pradhan, S. A Transition-based Algorithm for AMR Parsing. In Proceedings of the Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies; Association for Computational Linguistics: Denver, Colorado, 2015; pp. 366–375. https://doi.org/10.3115/v1/N15-1040.
37. Ballesteros, M.; Al-Onaizan, Y. AMR Parsing using Stack-LSTMs. In Proceedings of the Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing; Association for Computational Linguistics: Copenhagen, Denmark, 2017; pp. 1269–1275. https://doi.org/10.18653/v1/D17-1130.
38. Vilares, D.; Gómez-Rodríguez, C. Transition-based Parsing with Lighter Feed-Forward Networks. In Proceedings of the Proceedings of the Second Workshop on Universal Dependencies (UDW 2018); Association for Computational Linguistics: Brussels, Belgium, 2018; pp. 162–172. https://doi.org/10.18653/v1/W18-6019.
39. Naseem, T.; Shah, A.; Wan, H.; Florian, R.; Roukos, S.; Ballesteros, M. Rewarding Smatch: Transition-Based AMR Parsing with Reinforcement Learning. In Proceedings of the Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics; Association for Computational Linguistics: Florence, Italy, 2019; pp. 4586–4592. https://doi.org/10.18653/v1/P19-1451.
40. Fernandez Astudillo, R.; Ballesteros, M.; Naseem, T.; Blodgett, A.; Florian, R. Transition-based Parsing with Stack-Transformers. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2020; Association for Computational Linguistics: Online, 2020; pp. 1001–1007. https://doi.org/10.18653/v1/2020.findings-emnlp.89.
41. Zhou, J.; Naseem, T.; Fernandez Astudillo, R.; Florian, R. AMR Parsing with Action-Pointer Transformer. In Proceedings of the Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies; Association for Computational Linguistics: Online, 2021; pp. 5585–5598. https://doi.org/10.18653/v1/2021.naacl-main.443.
42. Zhou, J.; Naseem, T.; Fernandez Astudillo, R.; Lee, Y.S.; Florian, R.; Roukos, S. Structure-aware Fine-tuning of Sequence-to-sequence Transformers for Transition-based AMR Parsing. In Proceedings of the Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing; Association for Computational Linguistics: Online and Punta Cana, Dominican Republic, 2021; pp. 6279–6290. https://doi.org/10.18653/v1/2021.emnlp-main.507.
43. Peng, X.; Song, L.; Gildea, D. A Synchronous Hyperedge Replacement Grammar based approach for AMR parsing. In Proceedings of the Proceedings of the Nineteenth Conference on Computational Natural Language Learning; Association for Computational Linguistics: Beijing, China, 2015; pp. 32–41. https://doi.org/10.18653/v1/K15-1004.
44. Pust, M.; Hermjakob, U.; Knight, K.; Marcu, D.; May, J. Parsing English into Abstract Meaning Representation Using Syntax-Based Machine Translation. In Proceedings of the Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing; Association for Computational Linguistics: Lisbon, Portugal, 2015; pp. 1143–1154. https://doi.org/10.18653/v1/D15-1136.
45. Artzi, Y.; Lee, K.; Zettlemoyer, L. Broad-coverage CCG Semantic Parsing with AMR. In Proceedings of the Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing; Association for Computational Linguistics: Lisbon, Portugal, 2015; pp. 1699–1710. https://doi.org/10.18653/v1/D15-1198.
46. Groschwitz, J.; Lindemann, M.; Fowlie, M.; Johnson, M.; Koller, A. AMR dependency parsing with a typed semantic algebra. In Proceedings of the Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers); Association for Computational Linguistics: Melbourne, Australia, 2018; pp. 1831–1841. https://doi.org/10.18653/v1/P18-1170.

47. Lindemann, M.; Groschwitz, J.; Koller, A. Compositional Semantic Parsing across Graphbanks. In Proceedings of the Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics; Association for Computational Linguistics: Florence, Italy, 2019; pp. 4576–4585. https://doi.org/10.18653/v1/P19-1450.
48. Gessler, L.; Behzad, S.; Liu, Y.J.; Peng, S.; Zhu, Y.; Zeldes, A. DisCoDisCo at the DISRPT2021 Shared Task: A System for Discourse Segmentation, Classification, and Connective Detection. In Proceedings of the Proceedings of the 2nd Shared Task on Discourse Relation Parsing and Treebanking (DISRPT 2021); Association for Computational Linguistics: Punta Cana, Dominican Republic, 2021; pp. 51–62. https://doi.org/10.18653/v1/2021.disrpt-1.6.
49. Kobayashi, N.; Hirao, T.; Kamigaito, H.; Okumura, M.; Nagata, M. A Simple and Strong Baseline for End-to-End Neural RST-style Discourse Parsing. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2022; Association for Computational Linguistics: Abu Dhabi, United Arab Emirates, 2022; pp. 6725–6737.
50. Child, R.; Gray, S.; Radford, A.; Sutskever, I. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509* **2019**.
51. Liu, W.; Zhou, P.; Zhao, Z.; Wang, Z.; Ju, Q.; Deng, H.; Wang, P. K-bert: Enabling language representation with knowledge graph. In Proceedings of the Proceedings of the AAAI Conference on Artificial Intelligence, 2020, Vol. 34, pp. 2901–2908.
52. Bengio, Y.; Louradour, J.; Collobert, R.; Weston, J. Curriculum Learning. In Proceedings of the Proceedings of the 26th Annual International Conference on Machine Learning; Association for Computing Machinery: New York, NY, USA, 2009; ICML '09, p. 41–48. https://doi.org/10.1145/1553374.1553380.
53. May, J.; Priyadarshi, J. SemEval-2017 Task 9: Abstract Meaning Representation Parsing and Generation. In Proceedings of the Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017); Association for Computational Linguistics: Vancouver, Canada, 2017; pp. 536–545. https://doi.org/10.18653/v1/S17-2090.
54. Cai, S.; Knight, K. Smatch: an Evaluation Metric for Semantic Feature Structures. In Proceedings of the Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers); Association for Computational Linguistics: Sofia, Bulgaria, 2013; pp. 748–752.
55. Damonte, M.; Cohen, S.B.; Satta, G. An Incremental Parser for Abstract Meaning Representation. In Proceedings of the Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers; Association for Computational Linguistics: Valencia, Spain, 2017; pp. 536–546.
56. Riezler, S.; Maxwell, J.T. On Some Pitfalls in Automatic Evaluation and Significance Testing for MT. In Proceedings of the Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization; Association for Computational Linguistics: Ann Arbor, Michigan, 2005; pp. 57–64.