MRP 2019: Cross-Framework Meaning Representation Parsing

Stephan Oepen[♣], Omri Abend[♠], Jan Hajič[♡], Daniel Hershcovich[◇], Marco Kuhlmann[∘], Tim O'Gorman[⋆], Nianwen Xue[•], Jayeol Chun[•], Milan Straka[♡], and Zdeňka Urešová[♡]

[♣] University of Oslo, Department of Informatics

[♠] The Hebrew University of Jerusalem, School of Computer Science and Engineering

[♡] Charles University in Prague, Faculty of Mathematics and Physics, Institute of Formal and Applied Linguistics

[◇] University of Copenhagen, Department of Computer Science

[∘] Linköping University, Department of Computer and Information Science

[⋆] University of Colorado at Boulder, Department of Linguistics

[•] Brandeis University, Department of Computer Science

mrp-organizers@nlpl.eu,
jchun@brandeis.edu, {straka|uresova}@ufal.mff.cuni.cz

Abstract

The 2019 Shared Task at the Conference for Computational Language Learning (CoNLL) was devoted to Meaning Representation Parsing (MRP) across frameworks. Five distinct approaches to the representation of sentence meaning in the form of directed graphs were represented in the training and evaluation data for the task, packaged in a uniform graph abstraction and serialization. The task received submissions from eighteen teams, of which five do not participate in the official ranking because they arrived after the closing deadline, made use of extra training data, or involved one of the task co-organizers. All technical information regarding the task, including system submissions, official results, and links to supporting resources and software are available from the task web site at:

http://mrp.nlpl.eu

1 Background and Motivation

All things semantic are receiving heightened attention in recent years, and despite remarkable advances in vector-based (continuous and distributed) encodings of meaning, 'classic' (discrete and hierarchically structured) semantic representations will continue to play an important role in 'making sense' of natural language. While parsing has long been dominated by tree-structured target representations, there is now growing interest in general graphs as more expressive and arguably more adequate target structures for sentence-level analysis beyond surface syntax, and in particular for the representation of semantic structure.

The 2019 Conference on Computational Language Learning (CoNLL) hosts a shared task (or 'system bake-off') on Cross-Framework Meaning Representation Parsing (MRP 2019). The goal of the task is to advance data-driven parsing into *graph-structured* representations of *sentence meaning*. For the first time, this task combines *formally* and *linguistically* different approaches to meaning representation in graph form in a uniform training and evaluation setup. Participants were invited to develop parsing systems that support five distinct semantic graph frameworks (see § 3 below)—which all encode core predicate–argument structure, among other things—in the same implementation. Ideally, these parsers predict sentence-level meaning representations in all frameworks in parallel. Architectures utilizing complementary knowledge sources (e.g. via parameter sharing) were encouraged, though not required. Learning from multiple flavors of meaning representation in tandem has hardly been explored (with notable exceptions, e.g. the parsers of Peng et al., 2017; Hershcovich et al., 2018; or Stanovsky and Dagan, 2018).

Training and evaluation data were provided for all five frameworks. The task design aims to reduce framework-specific 'balkanization' in the field of meaning representation parsing. Its contributions include (a) a unifying formal model over different semantic graph banks (§ 2), (b) uniform representations and scoring (§ 4 and § 6), (c) contrastive evaluation across frameworks (§ 5), and (d) increased cross-fertilization via transfer and multi-task learning (§ 7). Thus, the task engages the combined community of parser developers for graph-structured output representations, including from prior framework-specific tasks at the Semantic Evaluation (SemEval) exercises between 2014 and 2019 (Oepen et al., 2014, 2015; May, 2016; May and Priyadarshi, 2017; Hershcovich et al., 2019). Owing to the scarcity of semantic anno-

Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 CoNLL, pages 1–27
Hong Kong, November 3, 2019. ©2019 Association for Computational Linguistics
https://doi.org/10.18653/v1/K19-2001

tations across frameworks, the MRP 2019 shared task is regrettably limited to parsing English for the time being.

2 Definitions: Graphs and Flavors

Reflecting different traditions and communities, there is wide variation in how individual meaning representation frameworks think (and talk) about semantic graphs, down to the level of visual conventions used in rendering graph structures. The following paragraphs provide semi-formal definitions of core graph-theoretic concepts that can be meaningfully applied across the range of frameworks represented in the shared task.

Basic Terminology Semantic graphs (across different frameworks) can be viewed as directed graphs or *digraphs*. A semantic digraph is a triple (T, N, E) where N is a set of *nodes* and $E \subseteq N \times N$ is a set of *edges*. The *in-* and *out-degree* of a node count the number of edges arriving at or leaving from the node, respectively. In contrast to the unique *root* node in trees, graphs can have multiple (structural) roots, which we define as nodes with in-degree zero. The majority of semantic graphs are structurally multi-rooted. Thus, we distinguish one or several nodes in each graph as *top* nodes, $T \subset N$; the top(s) correspond(s) to the most central semantic entities in the graph, usually the main predication(s).

In a tree, every node except the root has in-degree one. In semantic graphs, nodes can have in-degree two or higher (indicating shared arguments), which constitutes a *reentrancy* in the graph. In contrast to trees, general digraphs may contain *cycles*, i.e. a directed path leading from a node to itself. Another central property of trees is that they are *connected*, meaning that there exists an undirected path between any pair of nodes. In contrast, semantic graphs need not generally be connected.

Finally, in some semantic graph frameworks there is a (total) linear order on the nodes, typically induced by the surface order of corresponding tokens. Such graphs are conventionally called *bi-lexical dependencies* and formally constitute ordered graphs. A natural way to visualize a bi-lexical dependency graph is to draw its edges as semicircles in the halfplane above the sentence. An ordered graph is called *noncrossing* if in such a drawing, the semicircles intersect only at their endpoints (this property is a natural generalization of projectivity as it is known from dependency trees).

A natural generalization of the noncrossing property, where one is allowed to also use the halfplane below the sentence for drawing edges is a property called *pagenumber two*. Kuhlmann and Oepen (2016) provide additional definitions and a quantitative summary of various formal graph properties across frameworks.

Hierarchy of Formal Flavors In the context of the shared task, we distinguish different *flavors* of semantic graphs based on the nature of the relationship they assume between the linguistic surface signal (typically a written sentence, i.e. a string) and the nodes of the graph. We refer to this relation as *anchoring* (of nodes onto sub-strings); other commonly used terms include alignment, correspondence, or lexicalization.

Flavor (0) is the strongest form of anchoring, obtained in bi-lexical dependency graphs, where graph nodes injectively correspond to surface lexical units (i.e. tokens or 'words'). In such graphs, each node is directly linked to one specific token (conversely, there may be semantically empty tokens), and the nodes inherit the linear order of their corresponding tokens.

Flavor (1) includes a more general form of anchored semantic graphs, characterized by relaxing the correspondence between nodes and tokens, allowing arbitrary parts of the sentence (e.g. sub-token or multi-token sequences) as node anchors, as well as multiple nodes anchored to overlapping sub-strings. These graphs afford greater flexibility in the representation of meaning contributed by, for example, (derivational) affixes or phrasal constructions and facilitate lexical decomposition (e.g. of causatives or comparatives).

Finally, Flavor (2) semantic graphs do not consider the correspondence between nodes and the surface string as part of the representation of meaning (thus backgrounding notions of derivation and compositionality). Such semantic graphs are simply unanchored.

While different flavors refer to formally defined sub-classes of semantic graphs, we reserve the term *framework* for specific linguistic approaches to graph-based meaning representation (typically encoded in a particular graph flavor, of course).

3 Meaning Representation Frameworks

The shared task combines five frameworks for graph-based meaning representation, each with its specific formal and linguistic assumptions. This

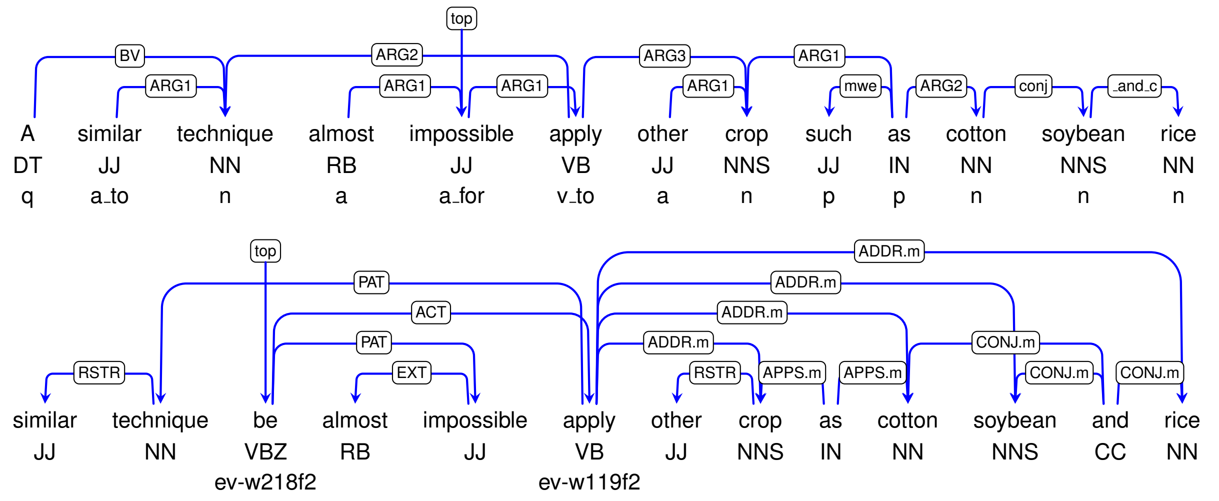

Figure 1: Bi-lexical semantic dependencies for the running example *A similar technique is almost impossible to apply to other crops, such as cotton, soybeans and rice*: DELPH-IN MRS Bi-Lexical Dependencies (DM; top) and Prague Semantic Dependencies (PSD; bottom).

section reviews the frameworks and presents example graphs for sentence #20209013 from the venerable Wall Street Journal (WSJ) Corpus from the Penn Treebank (PTB; Marcus et al., 1993):

(1) *A similar technique is almost impossible to apply to other crops, such as cotton, soybeans and rice.*

The example exhibits some interesting linguistic complexity, including what is called a tough adjective (*impossible*), a scopal adverb (*almost*), a tripartite coordinate structure, and apposition. The example graphs in Figures 1 through 3 are presented in order of (arguably) increasing 'abstraction' from the surface string, i.e. ranging from ordered Flavor (0) to unanchored Flavor (2).

Two of the frameworks in the shared task present simplifications into bi-lexical semantic dependencies (i.e. lossy reductions) of independently developed syntactico-semantic annotations. These representations were first prepared for the Semantic Dependency Parsing (SDP) tasks at the 2014 and 2015 SemEval campaigns (Oepen et al., 2014, 2015). The SDP graph banks were originally released through the Linguistic Data Consortium (as catalogue entry LDC 2016T10); they comprise four distinct bi-lexical semantic dependency frameworks, from which the MRP 2019 shared task selects two (a) DELPH-IN MRS Bi-Lexical Dependencies (DM) and (b) Prague Semantic Dependencies (PSD).[1]

DELPH-IN MRS Bi-Lexical Dependencies The DM bi-lexical dependencies (Ivanova et al., 2012) originally derive from the underspecified logical forms computed by the English Resource Grammar (Flickinger et al., 2017; Copestake et al., 2005). These logical forms are not in and of themselves semantic graphs (in the sense of § 2 above) and are often refered to as English Resource Semantics (ERS; Bender et al., 2015). The underlying grammar is rooted in the general linguistic theory of Head-Driven Phrase Structure Grammar (HPSG; Pollard and Sag, 1994).

Ivanova et al. (2012) propose a two-stage conversion from ERS into bi-lexical semantic dependency graphs, where ERS logical forms are first recast as Elementary Dependency Structures (EDS; Oepen and Lønning, 2006; see below) and then further simplified into pure bi-lexical semantic dependencies, dubbed DELPH-IN MRS Bi-Lexical Dependencies (or DM). As a Flavor (0) framework, graph nodes in DM are restricted to surface tokens. But DM graphs are neither lexically fully covering nor rooted trees, i.e. some tokens do not contribute to the graph, and for some nodes there are multiple incoming edges. In the example DM graph in Figure 1, *technique* semantically depends on the determiner (the quantificational locus), the modifier *similar*, and the predicate *apply*. Conversely, the predicative copula, infinitival *to*, and the vacu-

[1]Note, however, that the parsing problem for these frameworks is harder in the current shared task than in the ealier SDP 2014 and 2015 tasks, because gold-standard tokenization, lemmas, and parts of speech are not available as part of the parser input data. Also, some minor lemmatization errors have been corrected for both the DM and PSD graphs, in comparison to the original SDP releases.

ous preposition marking the deep object of *apply* (in the top of Figure 1) are analyzed as not having a semantic contribution of their own. The top node in the DM graph is the degree adverb *almost*, reflecting the underlying logical form, where *almost* has operator-like status scoping over the full proposition.

In DM, edge labels predominantly indicate semantic argument positions (ARG1, ARG2, ...) into the relation corresponding to their source node, but there are some more specialized edge labels too, like BV (bound variable) as a reflection of quantification in the underlying logic, conj and others for coordinate structures, and mwe to structurally tie together multi-token predicates. Node labels are tripartite, combining the lemmatized surface form with a part of speech (`pos`) and a framework-specific `frame` identifier. Together, these encode grammaticalized word sense distinctions, such as those between the nominal vs. verbal usages of *crop* or the distinct valency frames for three-place *apply ... to* (e.g. paint, to the wall) vs. binary *apply for* (e.g. promotion).

Prague Semantic Dependencies Another instance of simplification from richer syntactico-semantic representations into Flavor (0) bi-lexical semantic dependencies is the reduction of *tectogrammatical trees* (or t-trees) from the linguistic school of Functional Generative Description (FGD; Sgall et al., 1986; Hajič et al., 2012) into what are called Prague Semantic Dependencies (or PSD). Miyao et al. (2014) sketch the nature of this conversion, which essentially collapses empty (or *generated*, in FGD terminology) t-tree nodes with corresponding surface nodes and forward-projects incoming dependencies onto all members of paratactic constructions, e.g. the appositive and coordinate structures in the bottom of Figure 1.

The PSD graph for our running example has many of the same dependency edges as the DM one (albeit using a different labeling scheme and inverse directionality in a few cases), but it analyzes the predicative copula as semantically contentful and does not treat *almost* as 'scoping' over the entire graph. The ADDR.m(ember) argument relation to the *apply* predicate has been recursively propagated to both elements of the apposition and to all members of the coordinate structure. Accordingly, edge labels in PSD are not in general functional, in the sense of allowing multiple outgoing edges from one node with the same label.

In FGD, role labels (called *functors*) ACT(or), PAT(ient), ADDR(essee), ORIG(in), and EFF(ect) indicate 'participant' positions in an underlying valency frame and, thus, correspond more closely to the numbered argument positions in other frameworks than their names might suggest.[2] The PSD annotations are grounded in a machine-readable valency lexicon (Urešová et al., 2016), and the `frame` values on verbal nodes in Figure 1 indicate specific verbal senses in the lexicon.

Elementary Dependency Structures Elementary Dependency Structures (EDS; Oepen and Lønning, 2006) encode English Resource Semantics in a variable-free semantic dependency graph—not limited to bi-lexical dependencies—where graph nodes correspond to logical predications and edges to labeled argument positions. The EDS conversion from underspecified logical forms to directed graphs discards partial information on semantic scope from the full ERS, which makes these graphs abstractly—if not linguistically—similar to Abstract Meaning Representation (see below).

Nodes in EDS are in principle independent of surface lexical units, but for each node there is an explicit, many-to-many anchoring onto sub-strings of the underlying sentence. Thus, EDS instantiates Flavor (1) in our hierarchy of different formal types of semantic graphs. Breaking free of the Flavor (0) one-to-one correspondence between graph nodes and surface lexical units enables EDS to more adequately represent, among other things, lexical decomposition (e.g. of comparatives), sub-lexical or construction semantics, and covert (e.g. elided) meaning contributions. All nodes in the example EDS in the top of Figure 2 make explicit their anchoring onto sub-strings of the underlying input, for example span $\langle 2:9 \rangle$ for *similar*.

In the EDS analysis for the running example, nodes representing covert quantifiers (e.g. on bare nominals, labeled udef_q[3]), the two-place such+as_p relation, as well as the implicit_conj(unction) relation (which reflects recursive decomposition of the coordinate structure

[2]Accordingly, multiple instances of the same core participant role—as ADDR.m in Figure 1—will only occur with propagation of dependencies into paratactic constructions.

[3]In the EDS example in the top of Figure 2, all nodes corresponding to instances of bare 'nominal' meanings are bound by a covert quantificational predicate, including the group-forming implicit_conj and _and_c nodes that represent the nested, binary-branching coordinate structure. This practice of uniform quantifier introduction in ERS is acknowledged as "particularly exuberant" by Steedman (2011, p. 21).

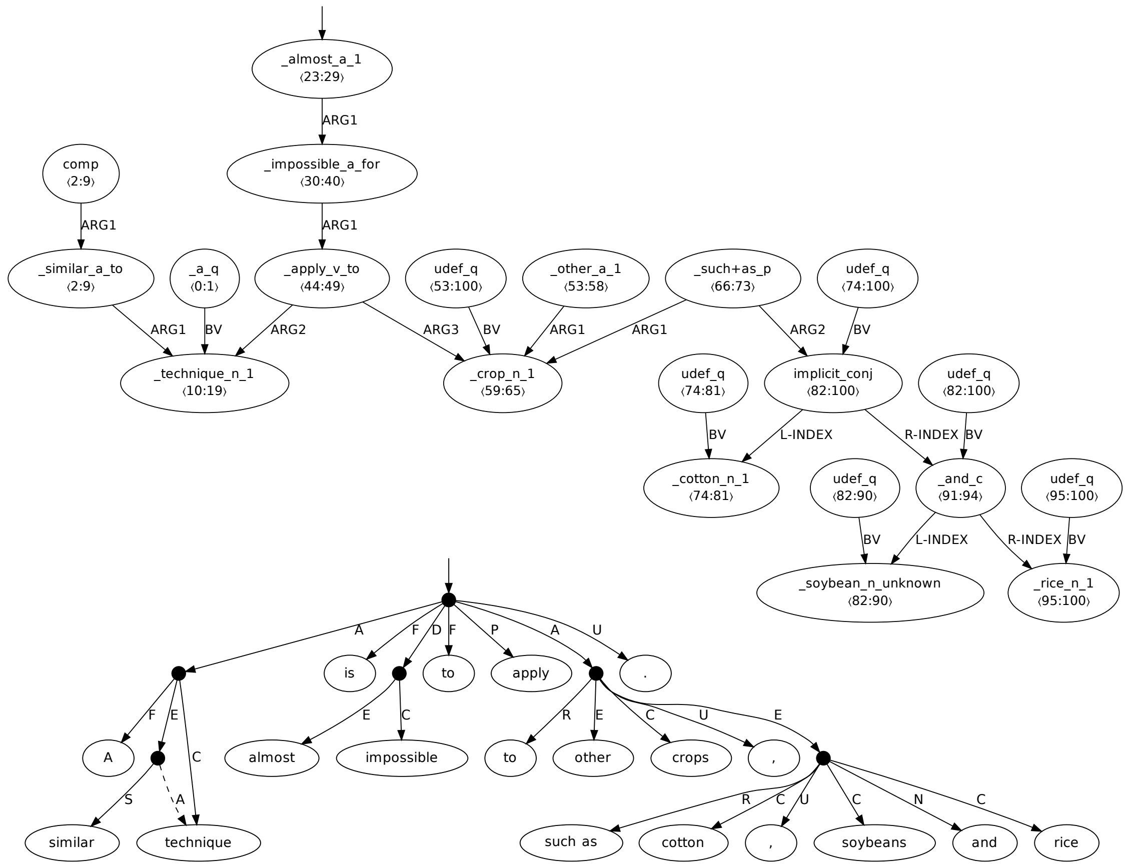

Figure 2: Semantic dependency graphs for the running example *A similar technique is almost impossible to apply to other crops, such as cotton, soybeans and rice*: Elementary Dependency Structures (EDS; top) and Universal Conceptual Cognitive Annotation (UCCA; bottom).

into binary predications) do not correspond to individual surface tokens (but are anchored on larger spans, overlapping with anchors from other nodes). Conversely, the two nodes associated with *similar* indicate lexical decomposition as a comparative predicate, where the second argument of the comp relation (the 'point of reference') remains unexpressed in Example (1).

Universal Conceptual Cognitive Annotation Universal Cognitive Conceptual Annotation (UCCA; Abend and Rappoport, 2013) is based on cognitive linguistic and typological theories, primarily Basic Linguistic Theory (Dixon, 2010/2012). The shared task targets the UCCA foundational layer, which focuses on argument structure phenomena (where predicates may be verbal, nominal, adjectival, or otherwise). This coarse-grained level of semantics has been shown to be preserved well across translations (Sulem et al., 2015). It has also been successfully used for improving text simplification (Sulem et al., 2018b), as well as to the evaluation of a number of text-to-text generation tasks (Birch et al., 2016; Sulem et al., 2018a; Choshen and Abend, 2018).

The basic unit of annotation is the *scene*, denoting a situation mentioned in the sentence, typically involving a predicate, participants, and potentially modifiers. Linguistically, UCCA adopts a notion of semantic constituency that transcends pure dependency graphs, in the sense of introducing separate, unlabeled nodes, called *units*. One or more labels are assigned to each edge. Formally, UCCA has a Type (1) flavor, where leaf (or terminal) nodes of the graph are anchored to possibly discontinuous sequences of surface sub-strings, while interior (or 'phrasal') graph nodes are formally unanchored.

The UCCA graph for the running example (see the bottom of Figure 2) includes a single scene, whose main relation is the Process (P) evoked by *apply*. It also contains a secondary relation labeled Adverbial (D), *almost impossible*, which is broken

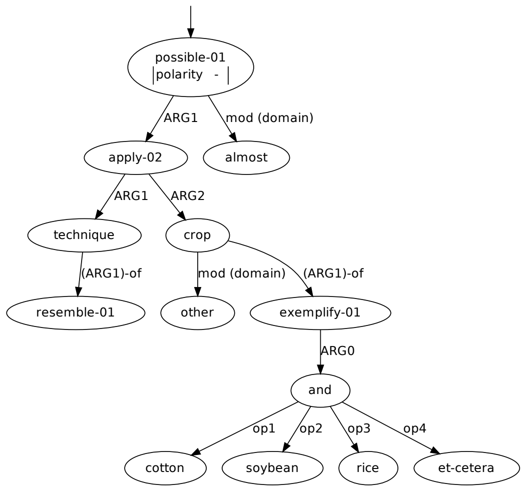

Figure 3: Abstract Meaning Representation (AMR) for the running example *A similar technique is almost impossible to apply to other crops, such as cotton, soybeans and rice*.

down into its Center (C) and Elaborator (E); as well as two complex arguments, labeled as Participants (A). Unlike the other frameworks in the task, the UCCA foundational layer integrates all surface tokens into the graph, possibly as the targets of semantically bleached Function (F) and Punctuation (U) edges. UCCA graphs need not be rooted trees: Argument sharing across units will give rise to reentrant nodes much like in the other frameworks. For example, *technique* in Figure 2 is both a Participant in the scene evoked by *similar* and a Center in the parent unit. UCCA in principle also supports implicit (unexpressed) units which do not correspond to any tokens, but these are currently excluded from parsing evaluation and, thus, suppressed in the UCCA graphs distributed in the context of the shared task.

Abstract Meaning Representation Finally, the shared task includes Abstract Meaning Representation (AMR; Banarescu et al., 2013), which in the MRP hierarchy of different formal types of semantic graphs (see § 2 above) is simply unanchored, i.e. represents Flavor (2). The AMR framework is independent of particular approaches to derivation and compositionality and, accordingly, does not make explicit how elements of the graph correspond to the surface utterance. Although most AMR parsing research presupposes a pre-processing step that 'aligns' graph nodes with (possibly discontinuous) sets of tokens in the underlying input, this anchoring is not part of the meaning representation proper.

At the same time, AMR frequently invokes lexical decomposition and normalization towards verbal senses, such that AMR graphs often appear to 'abstract' furthest from the surface signal. Since the first general release of an AMR graph bank in 2014, the framework has provided a popular target for data-driven meaning representation parsing and has been the subject of two consecutive tasks at SemEval 2016 and 2017 (May, 2016; May and Priyadarshi, 2017).

The AMR example graph in Figure 3 has a topology broadly comparable to EDS, with some notable differences. Similar to the UCCA example graph (and unlike EDS), the AMR representation of the coordinate structure is flat. Although most lemmas are linked to derivationally related forms in the sense lexicon, this is not universal, as seen by the nodes corresponding to *similar* and *such as*, which are labeled as resemble-01 and exemplify-01, respectively. These sense distinctions (primarily for verbal predicates) are grounded in the inventory of predicates from the PropBank lexicon (Kingsbury and Palmer, 2002; Hovy et al., 2006).

Role labels in AMR encode semantic argument positions, with the particular roles defined according to each PropBank sense, though the counting in AMR is zero-based such that the ARG1 and ARG2 roles in Figure 3 often correspond to ARG2 and ARG3, respectively, in the EDS of Figure 2. PropBank distinguishes such numbered arguments from non-core roles labeled from a general semantic inventory, such as frequency, duration, or domain.

Figure 3 also shows the use of inverted edges in AMR, for example ARG1-of and mod. These serve to allow annotators (and in principle also parsing systems) to view the graph as a tree-like structure (with occasional reentrancies) but are formally merely considered notational variants. Therefore, the MRP rendering of the AMR example graph also provides an unambiguous indication of the underlying, normalized graph: Edges with a label component shown in parentheses are to be reversed in normalization, e.g. representing an actual ARG0 edge from resemble-01 to technique or a domain edge from other to crop.

Given the non-compositionality of AMR annotation, AMR allows the introduction of semantic concepts which have no explicit lexicalization in the text, for example the et-cetera element in the coordinate structure in Figure 3. Conversely, like

		DM	PSD	EDS	UCCA	AMR
	Flavor	0	0	1	1	2
TRAIN	Text Type	newspaper	newspaper	newspaper	mixed	mixed
	Sentences	35,656	35,656	35,656	6,572	56,240
	Tokens	802,717	802,717	802,717	138,268	1,000,217
TEST	Text Type	mixed	mixed	mixed	mixed	mixed
	Sentences	3,359	3,359	3,359	1,131	1,998
	Tokens	64,853	64,853	64,853	21,647	39,520

Table 1: Quantitative summary of gold-standard training and evaluation data for the five frameworks.

in the other frameworks (except UCCA), some surface tokens are analyzed as semantically vacuous. For example, parallel to the PSD graph in Figure 1, there is no meaning contribution annotated for the determiner *a* (let alone for covert determiners in bare nominals, as are made explicit as quantificational nodes in EDS).

4 Task Setup

The following paragraphs summarize the 'logistics' of the MRP 2019 shared task, including data and software provided to participants, the schedule, and rules of participation.

Training and Evaluation Data Table 1 summarizes the primary training and evaluation data provided to task participants. The DM and PSD data sets are annotations over the exact same selection of texts, which for the eariler SemEval tasks have been aligned at the sentence and token levels. As DM was originally derived from EDS, the EDS graphs also cover the same texts. The training data for these frameworks draws from a homogeneous source, WSJ Sections 00–20 from the PTB. As a common point of reference, a sample of 100 WSJ sentences annotated in all five frameworks is available for public download from the task web site (see § 9 below).

UCCA training annotations are over web reviews from the English Web Treebank (LDC 2012T13), and from English Wikipedia articles on celebrities. While in principle UCCA structures are not confined to a single sentence (about 0.18 percent of edges cross sentence boundaries), in the MRP context passages are split to individual sentences, discarding inter-relations between them, to create a standard setting across the frameworks.

AMR annotations are drawn from a wide variety of texts, with the majority of sentences coming from on-line discussion forums. The training corpus also contains newswire, folktales, fiction, and Wikipedia articles.

Table 2 provides a quantitative side-by-side comparison of the training data, using some of the graph-theoretic properties discussed by Kuhlmann and Oepen (2016); see § 2 for semi-formal definitions (the row indices in Table 2 correspond to the numbering used by Kuhlmann and Oepen, 2016). The table indicates clear differences among the frameworks. The underlying input strings for AMR (where text selection is more varied), for example, are shorter; and EDS and UCCA have many more nodes per token, on average, than the other frameworks—reflecting lexical decomposition and 'phrasal' grouping, respectively, as evident in Figure 2. In some respects, the PSD and UCCA graphs are more tree-like than graphs in the other frameworks, for example in their proportions of actual rooted trees, the frequencies of reentrant nodes, and the lower percentages of multi-rooted structures. At the same time, PSD exhibits comparatively high average and maximal treewidth. Finally, the properties applicable to the ordered bi-lexical frameworks only are largely comparable, though PSD edges on average span over larger distances; propagation of dependencies into paratactic structures observed in Figure 1 may well contribute substantially to this quantitative difference.

Evaluation data for the five frameworks (also summarized in Table 1) draws on many of the same domains and genres, with two major additions: For DM, PSD, and EDS (where the training data is homogeneously comprised of newspaper texts), a little more than half of the evaluation data are taken from 'out-of-domain' texts, viz. a balanced sample of documents from the Brown Corpus (Francis and Kučera, 1982). Additionally, a fresh random selection of 100 sentences from the novel *The Little Prince* (by Antoine de Saint-Exupéry) was manually annotated with gold-standard semantic graphs

			DM	PSD	EDS	UCCA	AMR^{-1}
COUNTS	(02)	Average Tokens per Graph	22.51	22.51	22.51	21.03	17.78
	(03)	Average Nodes per Token	0.77	0.64	1.29	1.37	0.65
	(04)	Number of Edge Labels	59	90	10	15	101
TREENESS	(05)	$\%_g$ Rooted Trees	2.31	42.26	0.09	34.83	22.24
	(06)	$\%_g$ Treewidth One	69.82	43.08	68.99	41.57	50.00
	(07)	Average Treewidth	1.30	1.61	1.31	1.61	1.56
	(08)	Maximal Treewidth	3	7	3	4	5
	(09)	Average Edge Density	1.019	1.073	1.015	1.053	1.092
	(10)	$\%_n$ Reentrant	27.43	11.41	32.78	4.98	19.89
	(11)	$\%_g$ Cyclic	0.00	0.00	0.12	0.00	0.38
	(12)	$\%_g$ Not Connected	6.57	0.70	1.74	0.00	0.00
	(13)	$\%_g$ Multi-Rooted	97.47	40.60	99.93	0.00	71.37
ORDER	(15)	Average Edge Length	2.684	3.320	–	–	–
	(16)	$\%_g$ Noncrossing	69.21	64.61	–	–	–
	(17)	$\%_g$ Pagenumber Two	99.59	98.08	–	–	–

Table 2: Contrastive graph statistics for the MRP 2019 training data using a subset of the properties defined by Kuhlmann and Oepen (2016). Here, $\%_g$ and $\%_n$ indicate percentages of all graphs and nodes, respectively, in each framework; AMR^{-1} refers to the *normalized* form of the graphs, with inverted edges reversed, as discussed in § 3.

in all five frameworks.[4] This subset of the evaluation data is available for download from the task site.

Because some of the semantic graph banks involved in the shared task had originally been released by the Linguistic Data Consortium (LDC), the training data was made available to task participants by the LDC under no-cost evaluation licenses. Upon completion of the competition, all task data (including system submissions and evaluation results) are being prepared for general release through the LDC, while those subsets that are copyright-free will also become available for direct, open-source download.

Additional Resources For reasons of comparability and fairness, the shared task constrained which additional data or pre-trained models (e.g. corpora, word embeddings, lexica, or other annotations) can be legitimately used besides the resources distributed by the task organizers. The overall goal was that all participants should in principle be able to use the same range of data. However, to keep such constraints to the minimum required, a 'white-list' of legitimate resources was compiled from nominations by participants (with a cut-off date six weeks before the end of the evaluation period).[5] Thus, the task design reflects what is at times called a *closed track*, where participants are constrained in which additional data and pre-trained models can be used in system development.

At a technical level, training (and evaluation) data were distributed in two formats, (a) as sequences of 'raw' sentence strings and (b) in pre-tokenized, part-of-speech–tagged, lemmatized, and syntactically parsed form. For the latter, premium-quality English morpho-syntactic analyses were provided to participants, described in more detail below. These parser outputs are referred to as the MRP 2019 morpho-syntactic companion trees. Additional companion data available to participants includes automatically generated reference anchorings (commonly called 'alignments' in AMR parsing) for the AMR graphs in the training data, obtained from the JAMR and ISI tools of Flanigan et al. (2016) and Pourdamghani et al. (2014), respectively.

Companion Dependency Trees The optional morpho-syntactic trees were generated from the combination of a rule-based PTB-style tokenizer and a high-accuracy dependency parser trained on the union of (the majority of) available English syntactic treebanks. Notably, we applied an updated version of the converter by Schuster and Manning (2016) to the PTB annotations of the Brown Corpus (Francis and Kučera, 1982) and of the WSJ

[4]Annotations of the full novel have long served as a common reference point for AMR, and gold-standard DM and EDS graphs could be converted from the ERS inter-annotator agreement study by Bender et al. (2015). For PSD and UCCA, the 100-sentence subset used for MRP evaluation has been annotated specifically for the shared task.

[5]See `http://svn.nlpl.eu/mrp/2019/public/resources.txt` for the full list of seventeen generally available third-party resources, including a broad range of large English corpora and distributed word representations.

Corpus, as well as to the PTB-style annotations of the GENIA Corpus (Tateisi et al., 2005). This conversion targets Universal Dependencies (UD; McDonald et al., 2013; Nivre, 2015) version 2.x, so that the resulting gold-standard annotations could be concatenated with the UD English Web Treebank (Silveira et al., 2014), for a total of 2.2 million tokens annotated with lemmas, Universal and PTB-style parts of speech, and UD labeled dependency trees.

We then trained the currently best-performing UDPipe architecture (Straka, 2018; Straka et al., 2019), which implements a joint part-of-speech tagger, lemmatizer, and dependency parser employing contextualized BERT embeddings. To avoid overlap of morpho-syntactic training data with the texts underlying the semantic graphs of the shared task, we performed five-fold jack-knifing on the WSJ and EWT corpora. For compatibility with the majority of the training data, the 'raw' input strings for the MRP semantic graphs were tokenized using the PTB-style REPP rules of Dridan and Oepen (2012) and input to UDPipe in pre-tokenized form. Whether as merely a source of state-of-the-art PTB-style tokenization, or as a vantage point for approaches to meaning representation parsing that start from explicit syntactic structure, the optional morpho-syntactic companion data offers community value in its own right.

Graph Interchange Format Besides differences in anchoring, the frameworks also vary in how they label nodes and edges, and to what degree they allow multiple edges between two nodes, multiple outgoing edges of the same label, or multiple instances of the same property on a node. Node labels for Flavor (0) graphs typically are lemmas, optionally combined with a (morpho-syntactic) part of speech and a (syntactico-semantic) frame (or sense) identifier. Node labels for the other graph flavors tend to be more abstract, i.e. are interpreted as concept or relation identifiers (where for the vast majority, of course, there also is a systematic relationship to lemmas, lexical categories, and (sub-)senses). Graph nodes in UCCA are formally unlabeled, and anchoring is used to relate leaf nodes of these graphs to input sub-strings. Conversely, edge labels in all cases come from a fixed and relatively small inventory of (semantic) argument names, though there is stark variation in label granularity, ranging between about a dozen in UCCA and around 90 or 100 in PSD and AMR, respectively; see Table 2. The shared task has, for the first time, repackaged the five graph banks into a uniform and normalized abstract representation with a common serialization format.

The common interchange format for semantic graphs implements the abstract model of Kuhlmann and Oepen (2016) as a JSON-based serialization for graphs across frameworks. This format describes general directed graphs, with structured node and edge labels, and optional anchoring and ordering of nodes. JSON is easily manipulated in all programming languages and offers parser developers the option of 'in situ' augmentation of the graph representations from the task with system-specific additional information, e.g. by adding private properties to the JSON objects. The MRP interchange format is based on the JSON Lines format, where a stream of objects is serialized with line breaks as the separator character.

Each MRP graph is represented as a JSON object with top-level properties `tops`, `nodes`, and `edges`, reflecting the definitions in §2 above. Additionally, an `input` property on all graphs presents the 'raw' surface string corresponding to this graph; thus, parser inputs for the task are effectively assumed to be sentence-segmented but not pre-tokenized. Additional information about each graph is provided as properties `id` (a string), `flavor` (an integer in the range 0–2), `framework` (a string), `version` (a decimal number), and `time` (a string, encoding when the graph was serialized).

The `nodes` and `edges` values on graphs each are list-valued, but the order among list elements is only meaningful for the nodes of Flavor (0) graphs. Node objects have an obligatory `id` property (an integer) and optional properties called `label`, `properties` and `values`, as well as `anchors`. The `label` (a string) has a distinguished status in evaluation; the `properties` and `values` are both list-valued, such that elements between the lists correspond by position. Together, the two lists present a framework-specific, non-recursive attribute–value matrix (where duplicate properties are in principle allowed). The `anchors` list, if present, contains pairs of `from`–`to` sub-string indices into the `input` string of the graph. Finally, the edge objects in the top-level `edges` list all have two integer-valued properties: `source` and `target`, which encode the start and end nodes, respectively, to which the edge is incident. All edges in the MRP collection further have a (string-valued) `label` property, although formally this is considered optional. Parallel to graph nodes, edges can carry framework-specific `attributes` and `values` lists; in MRP 2019, only the UCCA framework makes use of edge attributes, viz. a boolean `remote` flag (corresponding to dashed edges in the bottom of Figure 2).

Rules of Participation The shared task was first announced in early March 2019, the initial release of the unified training data became available in mid-April, and the evaluation period ran between July 8 and 25, 2019; during this period, teams obtained the unannotated input strings for the evaluation data and had available a little more than two weeks to prepare and submit parser outputs. Submission of semantic graphs for evaluation was through the online CodaLab infrastructure, which proved a sub-optimal choice—in part due to limited transparency and customization options of the service, in part because technical problems on the CodaLab site caused the entire infrastructure to be unavailable for five days during the MRP evaluation period.

Teams were allowed to make repeated submissions, but only the most recent successful upload to CodaLab within the evaluation period was considered for the official, primary ranking of submissions. Task participants were encouraged to process all inputs using the same general parsing system, but—owing to inevitable fuzziness about what constitutes 'one' parser—this constraint was not formally enforced. Unlike in recent years of other CoNLL shared tasks, processing of the evaluation data was not tied to a uniform virtualization platform (such as TIRA; Potthast et al., 2014), because GPU computing resources are a prerequisite to modern, neural parsing architectures but are not currently available on such platforms.

5 Evaluation

For each of the individual frameworks, there are established ways of evaluating the quality of parser outputs in terms of graph similarity to gold-standard target representations called EDM (Dridan and Oepen, 2011), SMATCH (Cai and Knight, 2013), SDP (Oepen et al., 2014), and UCCA (Hershcovich et al., 2019). There is broad similarity between the framework-specific evaluation metrics used to date, but also some subtle differences. Meaning representation parsing is commonly evaluated in terms of a graph similarity F_1 score at the level of individual node–edge–node and node–property–value triples. Variations in extant metrics relate to among others, how node correspondences across two graphs are established, whether edge labels can optionally be ignored in triple comparison, and how top nodes (and other node properties, including anchoring) are evaluated.

	DM	PSD	EDS	UCCA	AMR
Top Nodes	✓	✓	✓	✓	✓
Node Labels	✓	✓	✓	✗	✓
Node Properties	✓	✓	✓	✗	✓
Node Anchoring	✓	✓	✓	✓	✗
Labeled Edges	✓	✓	✓	✓	✓
Edge Attributes	✗	✗	✗	✓	✗

Table 3: Different tuple types per framework.

Background In a nutshell, semantic graphs in all frameworks can be broken down into 'atomic' component pieces, i.e. tuples capturing (a) top nodes, (b) node labels, (c) node properties, (d) node anchoring, (e) labeled edges, and (f) edge attributes.[6] Not all tuple types apply to all frameworks, however, as is summarized in Table 3.

To evaluate any of these tuple types, a correspondence relation must be established between nodes (and edges) from the gold-standard vs. the system graphs. This relation presupposes a notion of node (and edge) identities, which is where the various flavors and frameworks differ. In bi-lexical (semantic) dependencies—e.g. DM and PSD, our Flavor (0)—the nodes are surface lexical units (tokens); their identities are uniquely determined as the character range of the corresponding sub-strings (rather than by token indices, which would not be robust to tokenization mis-matches). In the Flavor (1) graphs (EDS and UCCA), multiple distinct nodes can have overlapping or even identical anchors; in EDS, for example, the semantics of an adverb like *today* is decomposed into four nodes, all anchored to the same substring:

$$\text{implicit\_q } x : \text{time\_n}(x) \wedge \\ \text{\_today\_a\_1}(x) \ \wedge \ \text{temp\_loc}(e, x)\,.$$

The standard EDS and UCCA evaluation metrics determine node identities through anchors (and

[6]In principle, one could further view unlabeled edges and their labels as two distinct pieces of information, but the task design shies away from such formal purity for both linguistic and practical reasons. First, it does not appear desirable to try and give credit for edges with incompatible labels (e.g. an ARG1 with an ARG3); and, second, it would make the search for node-to-node correspondences somewhat less tractable.

transitively the union of child anchors, in the case of UCCA) and allow many-to-many correspondences across the gold-standard and system graphs (Oepen et al., 2014; Hershcovich et al., 2019). Finally, as a Flavor (2) framework, nodes in AMR graphs are unanchored. Thus, node-to-node correspondences need to be established (as one-to-one equivalence classes of node identifiers), to maximize the set of shared tuples between each pair of graphs. Abstractly, this is an instance of the NP-hard maximum common edge subgraph isomorphism problem (where node-local tuples can be modeled as 'pseudo-edges' with globally unique target nodes). The standard SMATCH scorer for AMR approximates a solution through a hill-climbing search for high-scoring correspondences, with a fixed number of random restarts (Cai and Knight, 2013).

Unified Evaluation For the shared task, we have implemented a generalization of existing, framework-specific metrics, along the lines above. Our goal is for the unified MRP metric to (a) be applicable across different flavors of semantic graphs, (b) enable labeled and unlabeled variants, as much as possible, (c) not require corresponding node anchoring, but (d) minimize the impact of non-deterministic approximations, and (e) take advantage of anchoring information when available. The official MRP metric for the task is the average F_1 score across frameworks over all tuple types.

The basic principle is that all information presented in the MRP graph representations is scored with equal weight, i.e. all applicable tuple types for each framework. There is no special status (or 'primacy') to anchoring in this scheme: Unlike the original SDP, EDM, and UCCA metrics, the MRP scorer searches for a correspondence relation between the gold-standard and system graphs that maximizes tuple overlap. Thus, the MRP approach is abstractly similar to SMATCH, but using a search algorithm that considers the full range of different tuple types and finds an exact solution in the majority of cases.[7]

Anchoring (for all frameworks but AMR) in this scheme is treated on a par with node labels and properties, labeled edges, and edge attributes. Likewise, the `pos` and `frame` (or sense) node properties in DM and PSD are scored with equal weight as the node labels (which are lemmas for the bi-lexical semantic graphs), given that the three properties jointly determine the semantic predicate.

For AMR evaluation, there is an exception to the above principle that all information in MRP graphs be scored equally: The MRP encodings of AMR graphs preserve the tree-like topology used in AMR annotations, using 'inverted' edges with labels like ARG0-of (see § 3 above). To make explicit which AMR edges actually are inverted, the MRP encoding in JSON provides an additional `normal` property, which is present only an inverted edges and provides the effective 'base' label (e.g. ARG0). AMR graphs are standardly evaluated in normalized form, i.e. with inverted edges restored to their 'base' directionality and label.

Software Support MRP scoring is implemented in the open-source `mtool` software (the Swiss Army Knife of Meaning Representation), which is hosted in a public Microsoft GitHub repository to stimulate community engagement.[8] `mtool` implements a refinement of the maximum common edge subgraph (MCES) algorithm by McGregor (1982), initializing and scheduling candidate node-to-node correspondences based on pre-computed per-node rewards and upper bounds on adjacent edge correspondences.[9] In addition to the cross-framework MRP metric, the tool also provides reference implementations of the SDP, EDM, SMATCH, and UCCA metrics, in the case of SDP and UCCA generalized to support character-based anchoring (rather than using token indices).

Value comparison in MRP evaluation is robust to 'uninteresting' variation, i.e. different encodings of essentially the same information. Specifically, literal values will always be compared as case-insensitive strings, such that for example `42` (an integer) and `"42"` (a string) are considered equivalent, as are `"Pierre"` and `"pierre"`; this applies to node and edge labels, node properties, and edge attributes. Anchor values are normalized for comparison into sets of non-whitespace character positions. For example, assuming the underlying

[7]The MRP scorer further avoids a few known implementation issues in SMATCH related to over-counting, incomplete normalization, and top nodes.

[8]See `https://github.com/cfmrp/mtool` for access to the software and available documentation.

[9]For the *ordered* DM and PSD graphs, an optimal initialization regarding node-local information can be efficiently computed, using an adaptation of the dynamic programming algorithm for minimum-edit–distance problems. For these graphs, scheduling of variant correspondences is further constrained to search for local variations first, i.e. alternate node–node pairings are considered in increasing node distance relative to the initial candidate correspondences.

Teams	**DM**	**PSD**	**EDS**	**UCCA**	**AMR**	**MTL**	**Approach**	**Reference**
ERGǂ§†	✓	✗	✓	✗	✗	✗	Composition	Oepen and Flickinger (2019)
TUPA§†	✓	✓	✓	✓	✓	✗	Transition	Hershcovich and Arviv (2019)
TUPA§†	✓	✓	✓	✓	✓	✓	Transition	Hershcovich and Arviv (2019)
HIT-SCIR	✓	✓	✓	✓	✓	✗	Transition	Che et al. (2019)
SJTU–NICT	✓	✓	✓	✓	✓	✗	Factorization	Li et al. (2019)
SUDA–Alibaba	✓	✓	✓	✓	✓	(✓)	Factorization	Zhang et al. (2019c)
Saarland	✓	✓	✓	✓	✓	✗	Composition	Donatelli et al. (2019)
Hitachi	✓	✓	✓	✓	✓	(✓)	Factorization	Koreeda et al. (2019)
ÚFAL MRPipe	✓	✓	✓	✓	✓	✗	Transition	Straka and Straková (2019)
ShanghaiTech	✓	✓	✓	✗	✓	✗	Factorization	Wang et al. (2019)
Amazon	✓	✓	✗	✗	✓	✗	Factorization	Cao et al. (2019)
JBNU	✓	✓	✗	✓	✗	✗	Factorization	Na et al. (2019)
SJTU	✓	✓	✓	✓	✓	✓	Transition	Bai and Zhao (2019)
ÚFAL–Oslo	✓	✓	✓	✗	✗	✗	Transition	Droganova et al. (2019)
HKUST	✓	✓	✗	✓	✗	?		
Bocharov	✗	✗	✗	✗	✓	?		
ÚFAL MRPipe§	✓	✓	✓	✓	✓	✗	Transition	Straka and Straková (2019)
Pekingǂ	✓	✓	✓	✓	✗	✗	Factorization	Chen et al. (2019)
ÚFAL–Oslo§	✓	✓	✓	✓	✓	✗	Transition	Droganova et al. (2019)
CUHK§	✓	✓	✓	✓	✓	✓	Transition	Lai et al. (2019)
Anonymous§	✗	✓	✗	✗	✗	?		
Pekingǂ§	✓	✓	✓	✓	✗	✗	Composition	Chen et al. (2019)

Table 4: Overview of participating teams. The top and bottom blocks represent 'unofficial' submissions, which are not considered for the *primary* ranking because they used training data beyond the white-listed resources (indicated by the symbol "ǂ"), arrived after the closing deadline ("§"), or were prepared by the task co-organizers as points of reference ("†"). The *secondary* ranking (see § 6) considers *all* submissions by genuine task participants (excluding co-organizers), i.e. both the middle and bottom blocks (but not the 'reference' systems from the top block).

input string contains whitespace at character position 6, the following are considered equivalent: $\{\langle 0:13\rangle\}$ and $\{\langle 0:6\rangle, \langle 7:13\rangle\}$.

Furthermore, character positions corresponding to basic punctuation marks in the left or right periphery of a normalized anchor are discarded for comparison:

```
. ? ! : ; , “ " ” ‘ ' ’ ( ) [ ] { }
```

6 Submissions and Results

The task received submissions from sixteen teams, plus another two 'reference' submissions prepared by the task co-organizers (Hershcovich and Arviv, 2019; Oepen and Flickinger, 2019). These reference points are not considered in the overall ranking. Non-reference submissions are further subdivided into 'official' and 'unofficial' ones, where the latter are characterized by either arriving after the closing deadline of the evaluation period or using training data beyond the official resources provided (and white-listed) for the task; see § 4 above.

Table 4 provides an inventory of participating teams, where the top block corresponds to reference submissions from the co-organizers, and the bottom block shows unofficial submissions by task participants. In two cases, participants discovered serialization or other technical issues in their submissions shortly after the closing date and provided corrected parser outputs (ÚFAL MRPipe and ÚFAL–Oslo). The two submissions from the Peking team are considered unofficial because they incorporate EDS-specific training data beyond the white-listed resources for the shared task (see § 4 above).[10] And, finally, the Anonymous and CUHK submissions only became available a few days after the closing date of the evaluation period.

It is evident in Table 4 that some submissions are partial, in the sense of not providing parser outputs for all target frameworks. Albeit not the ultimate goal of the cross-framework shared task design, such partiality was explicitly allowed to lower the technical barrier to entry and make it possible to include framework-specific parsers in the comparison. Seven (of thirteen) of the official submissions, as well as the two TUPA baselines, provide semantic graphs for all five frameworks. Three highly par-

[10] In the case of the factorization-based Peking submission, the extra training data is limited to gold-standard tokenization from the original EDS annotations, which in hindsight could in principle have been white-listed.

tial submissions declined the invitation to submit a system description for publication in the shared task proceedings (and one team asked to remain anonymous), such that only limited information is available about these parsers, and they will not be considered in further detail in § 7.

Finally, based on input by task participants, Table 4 also provides an indication of which submissions employed multi-task learning (MTL) and a high-level characterization of the overall parsing approach. The distinction between *transition*-, *factorization*-, and *composition*-based architectures follows Koller et al. (2019) and is discussed in more detail in § 7 below. In some submissions there can of course be elements of more than one of these high-level architecture types. Also, not all of the teams who indicate the use of multi-task learning actually apply it across different semantic graph frameworks, but in some cases rather to multiple sub-tasks within the parsing architecture for a single framework.[11]

The main task results are summarized in Table 6, showing average MRP scores across frameworks, broken down by the different component pieces (see § 5 above). These cross-framework averages can only be meaningfully compared for parsers that support all five frameworks, indicated with italics in the table. The top-three submissions achieve performance levels in the mid-80s F_1 range, followed by a competitive middle field of complete submissions that perform comparably to the TUPA baselines and well above. Despite fundamental architectural differences, there are emergent patterns in the average performance levels for different graph elements. Except for the binary top property, node-local information (fine-grained labels and properties) tend to be harder to predict than labeled edges. Edge attributes are only present in UCCA, encoding a binary distinction between primary and remote edges, which none of the parsers appear to predict successfully.

The correlation between the primary ranking of the official submissions (by overall average MRP F_1) and per-framework ranks is indicated in Table 5. The top-performing HIT-SCIR submission performs best on only one of the five frameworks (UCCA), but achieves uniformly strong results across the board; the picture is similar for the second-ranked SJTU–NICT submission (which has the best performance on DM). For the other top-performing submissions, there is more variation across frameworks: SUDA–Alibaba is strongest on the Flavor (1) EDS and UCCA graphs, and Saarland and Hitachi rank first and second, respectively, on the PSD graphs, but are not among the top-three ranks for the other frameworks.

As indicated, Table 5 shows the primary ranking, and unofficial submissions are not included. The complete summary of quantitative results from the task (see § 9 below) also provides a secondary ranking, considering all submissions (but not reference points) and excluding those entries that are superseded by others from the same team, viz. the earlier submissions from ÚFAL MRPipe and ÚFAL–Oslo and the EDS-only composition-based entry from Peking. In terms of secondary ranks, the unofficial ÚFAL MRPipe entry (correcting a minor bug in the original submission) would come in third overall (outranking SUDA–Alibaba), and the factorization-based Peking submission would take an overall seventh rank (outranking ShanghaiTech, and notably showing overall best performance for the EDS framework). Remaining secondary ranks are eleventh, thirteenth, and sixteenth, for ÚFAL–Oslo, CUHK, and Anonymous, respectively.

Table 5 also contrasts the ranking obtained from the official, cross-framework MRP metric in comparison to the pre-existing framework-specific metrics. For EDS, UCCA, and AMR there are only few

System	**DM**	**PSD**	**EDS**	**UCCA**	**AMR**
HIT-SCIR	2 : 2	4 : 3	2 : 3	1 : 1	2 : 2
SJTU–NICT	1 : 3	3 : 1	3 : 2	3 : 3	3 : 4
SUDA–Alibaba	7 : 7	8 : 8	1 : 1	2 : 2	5 : 5
Saarland	4 : 6	1 : 6	4 : 5	6 : 6	6 : 6
Hitachi	8 : 4	2 : 4	6 : 6	5 : 5	8 : 8
ÚFAL MRPipe	9 : 10	9 : 10	7 : 7	4 : 4	4 : 3
ShanghaiTech	3 : 1	6 : 2	5 : 4	10 : 10	7 : 7
Amazon	6 : 9	5 : 9	10 : 10	10 : 10	1 : 1
JBNU	5 : 5	7 : 5	10 : 10	7 : 8	11 : 11
SJTU	11 : 11	11 : 12	8 : 8	9 : 9	9 : 9
ÚFAL–Oslo	10 : 8	10 : 7	9 : 9	10 : 10	11 : 11
HKUST	12 : 12	12 : 11	10 : 10	8 : 7	11 : 11
Bocharov	13 : 13	13 : 13	10 : 10	10 : 10	10 : 10

Table 5: Per-framework rankings of the official submissions, contrasting the cross-framework MRP metric (first in each cell) and framework-specific evaluation (second). The order of entries reflects the primary ranking by overall average MRP F_1. Team names in italics indicate submissions that support all five frameworks.

[11]In the case of the SUDA–Alibaba submission, multi-task learning is only applied for the two bi-lexical frameworks; and for the Hitachi team it was only enabled in follow-up work after completion of the official evaluation period, as discussed in the system description by Koreeda et al. (2019).

	Tops			Labels			Properties			Anchors			Edges			Attributes			All		
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
ERG	.36	.36	.364	.39	.39	.390	.38	.38	.383	.39	.39	.391	.37	.37	.368	–	–	–	.38	.38	.383
	.38	.38	.376	.39	.39	.390	.36	.37	.368	.39	.40	.396	.37	.37	.372	–	–	–	.39	.39	.386
TUPA single	.65	.56	.603	.59	.76	.664	.41	.58	.479	.87	.81	.837	.34	.54	.414	.12	.22	.152	.51	.67	.577
	.76	.68	.718	.61	.76	.677	.43	.61	.501	.90	.79	.842	.31	.56	.401	.24	.24	.240	.50	.67	.575
TUPA multi	.67	.57	.616	.40	.55	.457	.29	.42	.327	.68	.60	.626	.30	.45	.347	.02	.02	.018	.39	.57	.453
	.75	.68	.714	.43	.55	.470	.21	.36	.234	.69	.64	.658	.35	.48	.390	.06	.03	.037	.45	.61	.506
HIT-SCIR	.91	.90	.904	.72	.70	.709	.70	.69	.699	.78	.77	.776	.81	.78	.794	.13	.12	.124	.87	.85	.862
	.93	.93	.932	.69	.68	.685	.60	.66	.625	.78	.78	.779	.80	.78	.786	.11	.08	.097	.85	.85	.848
SJTU–NICT	.92	.91	.915	.73	.70	.712	.71	.67	.687	.78	.77	.776	.80	.75	.777	.13	.07	.094	.87	.83	.853
	.94	.92	.931	.71	.70	.702	.50	.52	.505	.78	.77	.778	.79	.76	.773	.10	.05	.069	.85	.84	.842
SUDA–Alibaba	.88	.84	.860	.69	.70	.695	.68	.68	.682	.77	.77	.771	.77	.76	.768	.11	.07	.082	.84	.84	.840
	.90	.87	.884	.65	.67	.662	.60	.67	.636	.77	.78	.775	.77	.77	.770	.13	.05	.076	.82	.84	.832
Saarland	.83	.92	.867	.72	.71	.713	.72	.56	.611	.76	.75	.751	.76	.74	.750	–	–	–	.83	.80	.819
	.88	.93	.905	.72	.72	.723	.61	.58	.586	.77	.77	.771	.79	.77	.778	–	–	–	.85	.85	.849
Hitachi	.89	.90	.893	.64	.64	.641	.56	.54	.519	.75	.75	.755	.70	.69	.696	.08	.03	.042	.77	.75	.760
	.91	.92	.917	.62	.63	.624	.48	.43	.374	.75	.77	.760	.71	.70	.703	.10	.02	.034	.75	.77	.762
ÚFAL MRPipe	.83	.71	.751	.71	.59	.640	.70	.50	.565	.76	.64	.695	.70	.56	.622	.10	.06	.079	.83	.69	.747
	.85	.72	.758	.67	.55	.604	.68	.47	.539	.76	.63	.686	.69	.55	.608	.12	.05	.068	.80	.67	.729
ShanghaiTech	.73	.73	.733	.66	.67	.668	.59	.69	.633	.58	.57	.577	.63	.63	.628	–	–	–	.66	.68	.670
	.75	.75	.748	.65	.65	.649	.48	.61	.507	.58	.58	.578	.64	.64	.640	–	–	–	.66	.68	.668
Amazon	.45	.42	.438	.55	.54	.547	.53	.52	.525	.39	.39	.394	.46	.44	.450	–	–	–	.52	.51	.513
	.49	.47	.484	.52	.53	.526	.45	.49	.471	.39	.39	.392	.45	.46	.454	–	–	–	.50	.51	.502
JBNU	.56	.56	.560	.35	.35	.353	.37	.36	.365	.55	.55	.551	.41	.39	.400	.04	.02	.028	.47	.46	.465
	.57	.57	.566	.33	.33	.331	.34	.37	.355	.57	.58	.575	.44	.43	.431	.03	.01	.018	.48	.48	.483
SJTU	.68	.44	.527	.45	.42	.428	.29	.38	.321	.69	.45	.547	.36	.27	.295	.00	.00	.002	.46	.43	.430
	.74	.52	.602	.45	.45	.443	.22	.31	.249	.70	.47	.560	.37	.29	.308	.00	.00	.001	.47	.46	.451
ÚFAL–Oslo	.51	.51	.514	.20	.29	.239	.21	.37	.261	.43	.53	.464	.48	.40	.432	–	–	–	.30	.42	.344
	.53	.54	.534	.18	.28	.222	.19	.38	.239	.40	.54	.455	.50	.43	.459	–	–	–	.28	.43	.334
HKUST	.48	.45	.463	.20	.29	.238	–	–	–	.36	.49	.417	.25	.22	.230	.09	.04	.057	.22	.28	.245
	.43	.41	.420	.18	.28	.222	–	–	–	.37	.51	.426	.27	.23	.248	.07	.03	.046	.24	.30	.258
Bocharov	.17	.17	.167	.09	.07	.079	.01	.01	.011	–	–	–	.06	.05	.057	–	–	–	.07	.06	.065
	.17	.17	.172	.07	.09	.076	.02	.06	.027	–	–	–	.04	.07	.055	–	–	–	.06	.09	.068
ÚFAL MRPipe	.89	.78	.815	.74	.72	.731	.71	.69	.700	.77	.77	.772	.75	.73	.739	.10	.06	.079	.85	.83	.840
	.91	.78	.822	.71	.71	.710	.62	.65	.634	.77	.78	.775	.75	.74	.744	.12	.05	.068	.84	.83	.833
Peking	.74	.71	.725	.55	.54	.544	.56	.56	.560	.78	.78	.779	.67	.66	.666	.05	.07	.062	.71	.71	.711
	.76	.73	.744	.51	.52	.515	.43	.55	.480	.78	.78	.781	.67	.66	.663	.06	.03	.041	.70	.70	.702
ÚFAL–Oslo	.86	.78	.812	.34	.36	.332	.35	.42	.326	.49	.56	.502	.57	.44	.484	–	–	–	.46	.49	.439
	.88	.87	.871	.32	.42	.357	.33	.45	.335	.46	.60	.513	.57	.49	.527	–	–	–	.43	.56	.473
CUHK	.51	.50	.502	.34	.40	.365	.29	.35	.317	.55	.59	.568	.10	.10	.095	–	–	–	.36	.41	.378
	.51	.51	.514	.30	.39	.340	.24	.35	.283	.52	.62	.565	.09	.09	.087	–	–	–	.33	.42	.365
Anonymous	.04	.03	.035	.08	.13	.101	–	–	–	–	–	–	–	–	–	–	–	–	.02	.03	.022
	.04	.04	.038	.07	.11	.084	–	–	–	–	–	–	–	–	–	–	–	–	.01	.03	.019
Peking	.16	.16	.163	.19	.18	.185	.19	.19	.188	.19	.19	.187	.18	.18	.179	–	–	–	.18	.18	.184
	.17	.17	.174	.18	.18	.181	.16	.18	.166	.19	.19	.190	.18	.18	.178	–	–	–	.18	.19	.183

Table 6: Official results using the cross-framework MRP metric, broken down by 'atomic' component pieces. For each component we report precision (P), recall (R), and F_1 score (F). Entries are split into the same three blocks as in Table 4: references (top), official submissions (middle), and unofficial submissions (bottom). For each system, the first row shows MRP scores on the full evaluation set, while the second shows results on the public 100-sentence subset sampled from *The Little Prince*. The official and unofficial submissions are sorted by overall average F_1. Team names in italics indicate submissions that support all five frameworks.

and 'local' divergences in the rankings obtained from the different scoring approaches: In total, there are four instances of pairs of teams swapping ranks when comparing MRP vs. framework-specific results (the absolute per-framework scores in Table 7 suggest that such 'fluctuation' primarily reflects minor differences in performance). For DM and PSD, on the other hand, Table 5 reveals greater differences between the two ranks indicated in each cell: ShanghaiTech, for example, ranks much higher in the framework-specific SDP metric than in the official MRP ranks. These divergences likely reflect the more limited scope of the SDP approach to scoring, which essentially only considers labeled edges (and top nodes, as a pseudo-edge) but ignores node labels, properties, and anchors (which all used to be provided as part of the parser inputs in the original SDP parsing tasks; see § 3 above).

Finally, Tables 7 and 8 complement the breakdown of official results from the shared task with two per-framework views, using the official MRP metric and earlier framework-specific metrics, respectively. On both views, there are stark differences in overall parser accuracy across frameworks—ranging from the low-70s to mid-90s F_1 ranges—with mostly decreasing performance when moving from the bi-lexical Flavor (0) graphs to the unanchored Flavor (2) ones. Given the cross-framework MRP metric, these results become comparable for the first time (within the same parsing system at least, and assuming optimistically that it has been engineered and tuned at comparable effort levels for all frameworks). As such, it is tempting to interpret these differences as indicative of framework-specific parsing difficulty.

However, the volume, uniformity, and quality of available training data (and its similarity to evaluation data, in each framework) inevitably also must factor into such comparison; for example, gold-standard UCCA annotations count at less than one fifth the tokens of the other frameworks. Breaking down results further, viz. into component-wise per-framework scores (available through the task web site; see § 9), suggests that scoring the more technical anchoring information at equal weight as the genuinely linguistic node and edge properties contributes to higher average MRP accuracies, in particular for the bi-lexical frameworks where anchors essentially encode tokenization. Ultimately, to put these differences into perspective more, contrastive, phenomena-oriented studies would likely be called for, as for example the comparison of parsing accuracies for EDS vs. AMR by Lin and Xue (2019).

7 Overview of Approaches

The participating systems in the shared task have approached this multi-meaning representation task in a variety of ways, which we characterize into three broad families of approaches: transition-, factorization-, or composition-based architectures.

Transition-Based Architectures In these parsing system, the meaning representation graph is generated via a series of actions, in a process that is very similar to dependency tree parsing, with the difference being that the actions for graph parsing need to allow reentrancies, as well as (possibly) non-token nodes, labels, properties, and attributes. At any given point in the parsing process, a parser state, which typically consists of a stack that holds already processed elements in the input and a buffer for yet-to-be processed elements, needs to be maintained. Which action to take next is predicted by a classifier using a representation of the parser state as input. When this parsing procedure is complete, the sequence of parsing actions will be used to deterministically reconstitute the meaning representation graph.

This basic method allows variations in various aspects of the parsing process. First of all, the set of actions can vary from system to system. Apart from the standard actions used in syntactic dependency parsing such as SHIFT, LEFTARC, RIGHTARC, and REDUCE (Nivre, 2003; Yamada and Matsumoto, 2003), transition systems in meaning representation parsing also include actions to create reentrant edges, such as LEFTREMOTE and RIGHTREMOTE from the pre-task version of TUPA (Hershcovich et al., 2017). It may also include actions to create abstract concepts that do not correspond to a word token in the input sentence, such as the NODE action from TUPA, and actions that allow the transition to skip a word token in the input when it does not have semantic content, such as the PASS action from HIT-SCIR. The transition set may also include actions that label the nodes or edges, such as LABEL in the version of TUPA used in the shared task. CUHK developed a transition-based parser with a general transition system suited for all five frameworks, by including a variable-arity RESOLVE action.

	DM			PSD			EDS			UCCA			AMR		
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
ERG	.96	.96	.961	–	–	–	.95	.95	.952	–	–	–	–	–	–
	.97	.97	.973	–	–	–	.96	.96	.959	–	–	–	–	–	–
TUPA single	.47	.67	.555	.44	.63	.518	.83	.79	.810	.20	.45	.276	.42	.48	.447
	.50	.70	.586	.52	.68	.589	.83	.79	.814	.31	.57	.401	.43	.51	.470
TUPA multi	.31	.69	.427	.45	.63	.526	.74	.74	.740	.17	.38	.236	.29	.41	.338
	.28	.68	.395	.47	.65	.545	.74	.76	.748	.34	.52	.410	.45	.42	.434
HIT-SCIR	.95	.95	.951	.90	.91	.905	.91	.90	.907	.83	.81	.817	.77	.69	.729
	.95	.95	.950	.85	.90	.874	.89	.90	.898	.84	.82	.826	.72	.66	.690
SJTU–NICT	.96	.95	.955	.91	.91	.912	.95	.86	.899	.80	.76	.778	.75	.69	.720
	.95	.95	.949	.86	.91	.885	.94	.88	.912	.77	.74	.755	.72	.70	.706
SUDA–Alibaba	.91	.93	.923	.85	.86	.856	.92	.92	.918	.81	.76	.784	.73	.70	.717
	.89	.92	.907	.79	.87	.828	.92	.93	.925	.85	.80	.821	.67	.69	.679
Saarland	.95	.95	.947	.91	.91	.913	.90	.88	.891	.71	.65	.675	.70	.63	.667
	.94	.95	.948	.86	.91	.883	.93	.91	.920	.78	.74	.762	.74	.72	.731
Hitachi	.91	.91	.910	.91	.92	.912	.84	.84	.837	.72	.68	.704	.47	.41	.439
	.89	.90	.894	.86	.91	.884	.78	.84	.811	.78	.73	.750	.47	.47	.470
ÚFAL MRPipe	.91	.79	.850	.87	.68	.763	.82	.57	.674	.76	.71	.732	.77	.67	.718
	.91	.80	.854	.82	.60	.691	.77	.57	.651	.78	.71	.741	.74	.67	.707
ShanghaiTech	.95	.95	.949	.90	.89	.895	.86	.88	.869	–	–	–	.61	.66	.636
	.94	.94	.943	.83	.88	.852	.86	.89	.875	–	–	–	.66	.67	.668
Amazon	.94	.93	.933	.90	.90	.900	–	–	–	–	–	–	.75	.71	.734
	.92	.92	.921	.85	.91	.879	–	–	–	–	–	–	.71	.72	.711
JBNU	.94	.94	.940	.88	.88	.879	–	–	–	.53	.49	.507	–	–	–
	.92	.92	.924	.84	.88	.857	–	–	–	.66	.62	.636	–	–	–
SJTU	.36	.53	.431	.48	.48	.476	.75	.41	.532	.31	.35	.327	.40	.37	.385
	.35	.53	.419	.47	.51	.488	.74	.44	.553	.31	.40	.353	.46	.42	.441
ÚFAL–Oslo	.72	.91	.805	.48	.83	.609	.27	.35	.306	–	–	–	–	–	–
	.68	.91	.778	.43	.83	.566	.26	.43	.326	–	–	–	–	–	–
HKUST	.34	.41	.370	.28	.48	.353	–	–	–	.51	.50	.502	–	–	–
	.32	.42	.364	.26	.48	.334	–	–	–	.61	.58	.592	–	–	–
Bocharov	–	–	–	–	–	–	–	–	–	–	–	–	.37	.29	.327
	–	–	–	–	–	–	–	–	–	–	–	–	.28	.44	.342
ÚFAL MRPipe	.94	.95	.947	.90	.92	.910	.90	.89	.891	.76	.71	.732	.77	.67	.718
	.93	.95	.943	.85	.91	.878	.89	.90	.896	.78	.71	.740	.74	.67	.707
Peking	.94	.94	.944	.90	.89	.893	.95	.94	.945	.78	.77	.772	–	–	–
	.92	.93	.925	.83	.88	.853	.92	.93	.928	.82	.78	.803	–	–	–
ÚFAL–Oslo	.72	.91	.805	.48	.83	.609	.27	.35	.306	.23	.07	.112	.58	.27	.364
	.68	.91	.778	.43	.83	.566	.26	.43	.326	.23	.14	.175	.54	.50	.519
CUHK	.63	.75	.687	.60	.71	.648	.31	.25	.276	.18	.22	.196	.06	.12	.081
	.57	.73	.644	.51	.70	.590	.31	.32	.313	.22	.26	.235	.03	.08	.042
Anonymous	–	–	–	.08	.16	.109	–	–	–	–	–	–	–	–	–
	–	–	–	.07	.15	.095	–	–	–	–	–	–	–	–	–
Peking	–	–	–	–	–	–	.92	.92	.918	–	–	–	–	–	–
	–	–	–	–	–	–	.90	.93	.914	–	–	–	–	–	–

Table 7: Per-framework results using the official MRP metric. For each framework we report precision (P), recall (R), and F_1 score (F). Entries are split and sorted into the same three blocks as in Tables 4 and 6, and again the two rows per submission correspond to the full evaluation data and the *Little Prince* subset.

	DM			PSD			EDS			UCCA			AMR		
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
ERG	.91	.91	.912	–	–	–	.93	.92	.926	–	–	–	–	–	–
	.93	.93	.929	–	–	–	.94	.95	.944	–	–	–	–	–	–
TUPA single	.65	.69	.670	.51	.60	.552	.77	.71	.741	.28	.19	.224	.41	.47	.438
	.66	.71	.690	.55	.63	.585	.77	.72	.744	.32	.25	.284	.42	.49	.451
TUPA multi	.51	.62	.562	.47	.53	.501	.68	.64	.656	.28	.19	.224	.28	.39	.328
	.50	.63	.557	.52	.59	.553	.67	.65	.660	.32	.25	.284	.42	.40	.411
HIT-SCIR	.93	.92	.925	.81	.81	.810	.87	.86	.866	.68	.66	.667	.77	.69	.725
	.94	.94	.937	.79	.80	.794	.85	.86	.857	.66	.63	.644	.71	.65	.680
SJTU–NICT	.93	.92	.924	.82	.81	.817	.93	.83	.877	.63	.59	.609	.75	.68	.714
	.94	.93	.936	.81	.81	.810	.93	.87	.897	.63	.57	.597	.71	.69	.696
SUDA–Alibaba	.89	.91	.898	.76	.76	.760	.90	.89	.893	.66	.62	.639	.73	.70	.713
	.88	.91	.895	.75	.77	.759	.90	.91	.903	.69	.63	.662	.66	.69	.674
Saarland	.90	.91	.906	.80	.80	.796	.80	.78	.794	.34	.31	.324	.70	.63	.661
	.91	.93	.919	.79	.80	.798	.87	.85	.860	.52	.49	.505	.73	.71	.722
Hitachi	.91	.93	.919	.80	.82	.808	.78	.78	.783	.39	.37	.381	.46	.40	.425
	.92	.94	.927	.80	.82	.807	.73	.79	.757	.47	.44	.454	.45	.45	.453
ÚFAL MRPipe	.80	.70	.745	.69	.52	.594	.73	.49	.587	.42	.38	.396	.77	.67	.716
	.81	.72	.759	.68	.45	.539	.67	.48	.560	.48	.42	.445	.74	.67	.700
ShanghaiTech	.94	.92	.930	.83	.81	.816	.81	.82	.814	–	–	–	.61	.66	.631
	.95	.94	.945	.82	.82	.819	.81	.84	.825	–	–	–	.65	.66	.659
Amazon	.87	.86	.866	.76	.72	.742	–	–	–	–	–	–	.75	.71	.730
	.87	.87	.869	.77	.78	.771	–	–	–	–	–	–	.70	.71	.704
JBNU	.92	.90	.912	.80	.80	.800	–	–	–	.19	.17	.177	–	–	–
	.93	.92	.926	.82	.81	.815	–	–	–	.34	.31	.325	–	–	–
SJTU	.51	.30	.379	.49	.26	.340	.66	.33	.435	.05	.04	.045	.39	.36	.373
	.45	.27	.335	.52	.28	.359	.64	.34	.449	.06	.05	.055	.43	.39	.411
ÚFAL–Oslo	.90	.86	.880	.81	.73	.769	.14	.21	.168	–	–	–	–	–	–
	.90	.88	.888	.82	.77	.795	.15	.27	.192	–	–	–	–	–	–
HKUST	.33	.27	.297	.45	.36	.398	–	–	–	.21	.20	.203	–	–	–
	.33	.27	.299	.47	.36	.412	–	–	–	.25	.24	.244	–	–	–
Bocharov	–	–	–	–	–	–	–	–	–	–	–	–	.35	.28	.314
	–	–	–	–	–	–	–	–	–	–	–	–	.26	.41	.321
ÚFAL MRPipe	.87	.90	.881	.76	.79	.775	.87	.85	.859	.42	.38	.396	.77	.67	.716
	.87	.91	.893	.77	.80	.782	.87	.87	.869	.48	.41	.442	.73	.67	.699
Peking	.92	.92	.924	.81	.80	.808	.93	.91	.919	.63	.61	.620	–	–	–
	.93	.93	.925	.80	.80	.797	.90	.91	.906	.67	.62	.640	–	–	–
ÚFAL–Oslo	.90	.86	.880	.81	.73	.769	.14	.21	.168	–	–	.002	.56	.26	.351
	.90	.88	.888	.82	.77	.795	.15	.27	.192	–	–	.001	.53	.49	.508
CUHK	.10	.12	.108	.06	.06	.057	.05	.04	.047	.01	.01	.007	.05	.09	.060
	.10	.12	.109	.04	.05	.042	.08	.08	.083	.02	.01	.018	–	.01	.005
Anonymous	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Peking	–	–	–	–	–	–	.88	.88	.879	–	–	–	–	–	–
	–	–	–	–	–	–	.88	.90	.890	–	–	–	–	–	–

Table 8: Results using the framework-specific (labeled) metrics: SDP (for DM and PSD), EDM (for EDS), UCCA, and SMTACH (for AMR); see §5 above. For each framework (and its metric) we report precision (P), recall (R), and F_1 score (F). Entries are split and sorted into the same three blocks as in Tables 4 and 6, and again the two rows per submission correspond to the full evaluation data and the *Little Prince* subset.

Second, the classifier used to predict the action for any given state can also vary a great deal. For example, the HIT-SCIR system aggregates information from the action history, the stack, the list, and the buffer with a stack LSTM and then predicts the action by taking a softmax over the output of the LSTM. The CUHK system uses a regular LSTM to aggregate information from the stack, the sequence of words before the current word token, and the sequence of words after the current token, and then predict the action with a softmax. The TUPA system uses a BiLSTM with an MLP and softmax layer, with the BiLSTM running over the sequence of input tokens.

Factorization-Based Architectures These parsing models for meaning representation also have their roots in syntactic dependency parsing (where they are often called *graph-based*; McDonald and Pereira, 2006). Given a set of nodes, the basic idea of the factorization-based approach is to find the graph that has the highest score among all possible graphs. In the case of dependency parsing, the goal is to find the Maximum Spanning Tree, and this has been extended to meaning representation parsing, where the goal is to find the Maximum Spanning Connected Subgraphs (Flanigan et al., 2014). To make the computation of the score of a graph practical, the typical strategy is to factorize the score of a graph into the sum of the scores of its subgraphs, and in the case of first-order factorization, into the sum of the scores of its nodes and edges. A popular choice for predicting the edge is to feed the output of an LSTM encoder to a biaffine classifier to predict if an edge exists between a pair of nodes as well as the label of the edge (SJTU–NICT, SUDA–Alibaba, Hitachi, and JBNU), with slight variations as to the input to the LSTM encoder. Due to the difference in anchoring between the nodes in the graph and the word tokens in the sentence, the way to identify nodes also differs from framework to framework. ÚFAL–Oslo used the factorization-based NeurboParser (Peng et al., 2017) for DM and PSD, and for EDS they simply submitted graphs identical to the DM ones. They also used the factorization-based JAMR (Flanigan et al., 2014, 2016) for AMR, and further adjusted JAMR to support UCCA graphs, by converting UCCA to the standard AMR serialization.

Composition-Based Architectures Finally, this approach to meaning representation parsing emphasizes the principle of compositionality in meaning construction and assumes an explict inventory of operations that combine pieces of meaning into larger fragments. Typically grounded in some kind of formal derivation process, composition-based architectures associate meaning fragments with lexical items (leaf nodes in the derivation) and apply a designated composition operation for each step in the derivation. What differentiates composition-based approaches from transition-based or factorization-based ones is that the derivations are licensed by some form of 'grammar' (explicit or implicit), where illegitimate derivations can be ruled out by the structural constraints over the lexical items and the rules of derivation. The MRP shared task attracted two (and a half) composition-based systems, the Apply-Modify (AM) algebra based system from Saarland and the Peking parser based on Synchronous Hyperedge Replacement Grammar (SHRG) for EDS.[12] For composition-based approaches, the extraction of lexical items from a sentence is a crucial component of the system. In the case of the Saarland parser, the lexical items are produced by a BiLSTM-based supertagger, and the best derivation is selected in a tree dependency parsing process where the edge between a head and its argument or modifier is labeled with the derivation operation. In the case of the Peking system, the SHRG rules are extracted with a context-free parser, and the derivation is scored by a sum of the scores of its subgraphs.

Other Approaches The transition-, factorization-, and composition-based systems represent the main approaches in the shared task, but there are a few systems that stretch the dividing lines of this this categorization. When parsing the UCCA framework, a number of systems—e.g. SJTU–NICT, SUDA–Alibaba, and Amazon—adopt an approach where 'remote' (reentrancy) edges are first removed to create constituent tree structures to train standard constituent tree parsers using neural network–based models, and then in a postprocessing stage, the remote edges are added back with a separate classifier, following Jiang et al. (2019).

The MRPipe system could be said to define its own category. It differs from transition-based systems in that it does not use the typical actions

[12]The unofficial submission of DM and EDS reference graphs obtained from parsing with the ERG also represents a composition-based approach.

used in transition-based systems and it also does not maintain a typical parser state. It also differs from factorization-based systems in that it builds the meaning representation iteratively, while in a factorization-based systems all possible graphs are (conceptually) enumerated at once and the focus is on finding the graph with the highest score.

Anchoring One difference among the five meaning representation frameworks covered in the shared task is the correspondence relation between the concepts (graph nodes) and word tokens in the sentence (see § 2). In Flavors (0) and (1) (DM, PSD, EDS, and UCCA), this alignment is explicit, while in Flavor (2) AMRs there is no explicit anchoring. How to tackle anchoring in the parsing system has a significant impact on parser performance. Some of the participating systems follow early approaches in AMR parsing and use a separate 'alignment' model to provide hard anchorings and then proceed with the rest of the parsing process (e.g. the HIT-SCIR system) assuming the alignments are already in place. Other submissions use a soft alignment component that is trained jointly with other components of their systems. For example, the Amazon and the SUDA–Alibaba parsers jointly model anchoring, node detection, and edge detection, adopting the approach of Lyu and Titov (2018), while the SJTU–NICT system uses a sequence-to-sequence model with a pointer-generator network to predict the concepts in AMR, following Zhang et al. (2019a). That sequence-to-sequence model is trained jointly with other components of their system.

Cross-Framework Architecture Design One question that the co-organizers would like to help answer through the shared task is to what degree the same general architecture can be used to effectively parse all five meaning representation frameworks. The answer to this question is tentatively in the affirmative. The HIT-SCIR and TUPA systems use a transition-based system to parse all five meaning representations, with the caveat that the transitions for the five meaning representations vary in the actions that are used. The CUHK parser, on the other hand, uses a uniform transition set for all frameworks. The Saarland system uses the same AM algebra composition system to parse all five meaning representations, but has to do a considerable amount of pre-processing to convert the meaning representations into well-formed terms of the AM algrebra (accordingly, some of the pre-processing effects need to be undone in post-processing). The MRPipe system adopts an approach in which the meaning representation graph is built up iteratively with two operations, AddNodes and AddEdges, and applies this model successfully to all five meaning representations. Other participating systems adopt the strategy of using the model that they consider to be the most appropriate for a particular flavor of meaning representation. For example, the SJTU–NICT submission uses a factorization-based model for DM, PSD, and EDS parsing, but uses a constituent tree parsing approach for UCCA, as it is not obvious how a factorization-based model would be extended to also handle UCCA parsing. The Amazon system uses a factorization-based model for DM, PSD, and AMR while adopting a constituent tree parsing approach for UCCA and EDS. The SUDA–Alibaba system also adopts a constituent tree parsing approach to UCCA, similar to Jiang et al. (2019).

Benefits of Multi-Task Learning Another research question the shared task seeks to advance is whether and how multi-task learning (MTL) helps with multi-framework meaning representation parsing. The term, in fact, seems to be applied somewhat variably in the system descriptions. In one sense, it is equated with traditional joint learning, where different components of the SUDA–Alibaba system are trained jointly by combining their objectives. The sense of the term that was intended by the organizers is whether pooling the training data for all five frameworks in a multi-task learning framework can improve the parser performance of one particular framework. A number of participating systems attempted MTL in the latter sense, and the results are mixed and not definitive. The MTL version of the TUPA system performs much worse than its single-task version, but this might be attributed to inadequate training strategies and incomplete tuning. The Hitachi systems (in a post-competition experiment) show MTL results that are slightly better than single framework results, but the difference is probably not statistically significant.

8 On the State of the Art

Prior to the shared task, various methods have been proposed for semantic graph parsing, including transition-, factorization-, and composition-based, as well as sequence-to-sequence systems. Existing parsers also diverge in terms of their assumptions regarding the syntax–semantics interface, some parsing raw text directly to meaning representation graphs, and some producing the graphs from or in parallel with syntactic derivations.

While some meaning representations have parsers for languages other than English (Oepen et al., 2015; Wang et al., 2018; Damonte and Cohen, 2018; Hershcovich et al., 2019), we limit the discussion here to the state of the art in English meaning representation parsing, as has been the focus of the current shared task.

DM and PSD were both among the representations targeted in two SemEval shared tasks on Semantic Dependency Parsing (Oepen et al., 2014, 2015), where the winning system (Kanerva et al., 2015) utilized SVM-based sequence labeling. The runner-up (Du et al., 2014, 2015) used an ensemble based on factorization-based weighted tree approximation. More recently, Peng et al. (2017, 2018a,b) improved upon previous approaches by using a neural factorization-based multi-task system, sharing parameters between representations and applying joint inference. Stanovsky and Dagan (2018) linearized the bi-lexical graphs and modeled the parsing task as a sequence-to-sequence problem. They also used multi-task learning, adapting multilingual machine translation algorithms to 'translate' between text and meaning representations, outperforming the previous best results on PSD. Lindemann et al. (2019) trained a composition-based parser on DM, PAS, PSD, AMR and EDS, using the Apply–Modify algebra, on which the Saarland submission to the shared task is based. They employed multi-task training with all tackled semantic frameworks and UD, establishing the state of the art on all graph banks but AMR 2017.

AMR has been a challenging target representation for parsing, due to the fact that AMRs are Flavor (2), unanchored graphs. AMR parsing was pioneered by Flanigan et al. (2014), who performed alignment as a preprocessing step during training. They developed their own rule-based alignment method, complemented by Pourdamghani et al. (2014), who adapted methods from machine translation. Some transition-based AMR parsers also perform rule-based alignment (Damonte et al., 2017; Damonte and Cohen, 2018; Ballesteros and Al-Onaizan, 2017; Naseem et al., 2019), while others derive AMRs from syntactic dependencies by applying transitions (Wang et al., 2015; Wang and Xue, 2017). The latter approach reached the best performance (Wang et al., 2016; Nguyen and Nguyen, 2017) in two SemEval shared tasks on AMR parsing (May, 2016; May and Priyadarshi, 2017), where in the former it performed as well as a novel character-level neural translation based AMR parser (Barzdins and Gosko, 2016). Composition-based AMR parsers include Artzi et al. (2015), who combined CCG grammar induction with AMR parsing. Sequence-to-sequence attention-based approaches (Konstas et al., 2017; van Noord and Bos, 2017) use techniques from machine translation to directly generate (linearized) graphs from text. Lyu and Titov (2018) parsed AMR using a joint probabilistic model with latent alignments, avoiding cascading errors due to alignment inaccuracies and outperforming previous approaches. The factorization-based parser by Zhang et al. (2019a,b) uses an attention-based architecture, but derives target graphs directly instead of a linearization, also treating alignment as a latent variable with a copy mechanism. Their parser additionally supports UCCA and SDP, and establishes the state-of-the-art in AMR parsing, though without using multi-task training across frameworks.

UCCA parsing was first tackled by Hershcovich et al. (2017), who used a neural transition-based parser. Hershcovich et al. (2018) further showed that multi-task learning with AMR, DM, and UD as auxiliary tasks improves UCCA parsing performance. UCCA also recently featured in a SemEval shared task (Hershcovich et al., 2019), where the composition-based best system (Jiang et al., 2019) outperformed the transition-based baseline by treating the task as constituency tree parsing with the recovery of remote edges as a postprocessing task.

EDS, being a result of automatic conversion from English Resource Semantics (Bender et al., 2015), can be derived by any ERG parser (e.g. Callmeier, 2002; Packard, 2012). Buys and Blunsom (2017) were the first to build a purely data-driven EDS parser, combining graph linearization with a custom transition system. Chen et al. (2018) established the state of the art on data-driven EDS parsing, using a neural SHRG-based, ERG-guided parser. Their comparison on in-domain WSJ evaluation data showed parsing accuracies on par or in excess of the full, grammar-based ACE parser of Packard (2012).

While some shared task submissions are based on existing systems that have been specifically improved, direct comparison to previously published results is impossible: Our definition of the SDP task, for example, is different from Oepen et al. (2014, 2015); prior EDS work has mostly tested on WSJ only; the UCCA annotations have been revised and extended; we are using a new, forthcoming version of AMRbank; and gold-standard tokenization is not provided for any of the frameworks. Also, even some of our framework-specific metrics are not exactly what was used previously: We have made SDP and UCCA character-based (for increased robustness to tokenization mismatches), and we un-invert edges more thoroughly in AMR graphs before calling SMATCH for scoring. However, overall performance levels and general trends observed in § 6 appear consistent with recent developments in the field: By and large, the transition-, factorization-, and composition-based approaches all can yield competitive parsers, where cross-framework multi-task learning sometimes helps but only slightly so. While general methods for meaning representation graph parsing are clearly beneficial, there is yet progress to be made (so far) in sharing information between parsers for different frameworks and making better use of their overlap.

9 Reflections and Outlook

The MRP 2019 shared task was a first step in a new direction, aiming to more closely (inter)relate the representations and parsing approaches across a diverse range of semantic graph frameworks. Despite new uniformity in packaging and evaluation, cumulative overall complexity and inherent technical and linguistic diversity of the frameworks deemed participation in the competition a demanding challenge. The problem attracted broad interest: Some 140 individuals have subscribed to the shared task mailing list, and 38 teams obtained the training data package from the LDC (of these, sixteen submitted parser outputs for evaluation). In a post-evaluation questionnaire and through informal communication, several prospective participants have indicated that they had started to work towards a system submission but in the end simply ran out of time for the official evaluation period.

Possibly related to the high technological barrier to participation is the comparatively low proportion of submissions that successfully utilize multi-task learning (across frameworks). Even though some of the participating teams have previously applied multi-task learning for semantic graph parsing, it appears some may have shied away from increased training times and tuning effort and instead had to focus their work on developing strong end-to-end parsers for individual frameworks. As task co-organizers, we remain committed to enabling continued research along these lines, and we will ultimately make all training and evaluation data generally available. In the interim, however, we are delighted (and a little frightened) to confirm that CoNLL has invited us to orchestrate a follow-up shared task on Cross-Framework Meaning Representation Parsing in 2020.

Deciding on the task parameters for MRP 2020 will be a balancing act between keeping overall complexity manageable, in particular for 'newcomer' participants, and pushing further in the direction of learning from complementary knowledge sources. Above all, the mid- to long-term goals of the cross-framework meaning representation initiative are to advance our understanding of degrees of complementarity among the various frameworks. Current plans foresee inclusion of one additional framework, viz. a graph-based encoding of the Discourse Representation Structures of Basile et al. (2012). Further, we plan on refining and extending the available training data (in particular for UCCA) and will put greater focus on the systematic exploration of variant evaluation perspectives, for example scoring at the level of larger sub-graphs in the spirit of the 'complete predications' metric of Oepen et al. (2015), or 'semantic n-grams' along the lines of the SemBleu proposal by Song and Gildea (2019). Aiming for increased linguistic diversity, it will of course also be tempting to seek to include meaning representations for additional languages. For each of the frameworks involved (six in total for MRP 2020), gold-standard annotations are in principle available for at least one language besides English, but in most cases these would be different languages for each framework. Thus, it remains yet to be decided how best to balance cross-linguistic and multi-task perspectives on the MRP problem.

All technical information regarding the MRP 2019 shared task, including system submissions, detailed official results, and links to supporting resources and software are available from the task web site at:

`http://mrp.nlpl.eu`

Acknowledgments

Many colleagues have assisted in designing the task and preparing its data and software resources. Emily M. Bender and Dan Flickinger provided a critical review of (a sample of) the DM and EDS graphs. Sebastian Schuster made available a pre-release of the converter from PTB-style constituent trees to (basic) UD 2.x dependency graphs. Dotan Dvir has coordinated the team of UCCA annotators, always ensuring that the corpora were ready in time. Andrey Kutuzov helped with the preparation of morpho-syntactic companion trees for the evaluation data.

The task design and implementation has benefited from input by the Steering Committee of the ACL Special Interest Group on Natural Language Learning, notably Xavier Carreras and Julia Hockenmaier, as well as by the CoNLL 2019 Programme Chairs, Mohit Bansal and Aline Villavicencio.

We are grateful to the Nordic e-Infrastructure Collaboration for their support to the Nordic Language Processing Laboratory (NLPL), which has provided technical infrastructure for the MRP 2019 task. Also, we thankfully acknowledge the assistance of the Linguistic Data Consortium in distributing the training data for the task to participants at no cost to anyone.

We acknowledge the support of the Czech Ministry of Education, Youth, and Sports, through project CZ.02.1.01/0.0/0.0/16_013/0001781 (EF16_013/0001781); Czech Ministry of Culture, project DG16P02R019, the support for the data and services used by the Research Infrastructure projects LM2015071 and LM2018101, also of the Czech Ministry of Education, Youth, and Sports, and the support of the Grant Agency of the Czech Republic, project No. GA17-07313S. The work on UCCA and TUPA was partially supported by the Israel Science Foundation (grant no. 929/17). We also acknowledge the support of the US National Science Foundation on the Uniform Meaning Representation project via Award No. 1763926. All views expressed in this paper are those of the authors and do not necessarily represent the view of the National Science Foundation.

References

Omri Abend and Ari Rappoport. 2013. UCCA. A semantics-based grammatical annotation scheme. In *Proceedings of the 10th International Conference on Computational Semantics*, pages 1 – 12, Potsdam, Germany.

Yoav Artzi, Kenton Lee, and Luke Zettlemoyer. 2015. Broad-coverage CCG semantic parsing with AMR. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1699 – 1710, Lisbon, Portugal.

Hongxiao Bai and Hai Zhao. 2019. SJTU at MRP 2019: A transition-based multi-task parser for cross-framework meaning representation parsing. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 86 – 94, Hong Kong, China.

Miguel Ballesteros and Yaser Al-Onaizan. 2017. AMR parsing using stack-LSTMs. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1269 – 1275, Copenhagen, Denmark.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178 – 186, Sofia, Bulgaria.

Guntis Barzdins and Didzis Gosko. 2016. RIGA at SemEval-2016 task 8: Impact of Smatch extensions and character-level neural translation on AMR parsing accuracy. In *Proceedings of the 10th International Workshop on Semantic Evaluation*, pages 1143 – 1147, San Diego, CA, USA.

Valerio Basile, Johan Bos, Kilian Evang, and Noortje Venhuizen. 2012. Developing a large semantically annotated corpus. In *Proceedings of the 8th International Conference on Language Resources and Evaluation*, pages 3196 – 3200, Istanbul, Turkey.

Emily M. Bender, Dan Flickinger, Stephan Oepen, Woodley Packard, and Ann Copestake. 2015. Layers of interpretation. On grammar and compositionality. In *Proceedings of the 11th International Conference on Computational Semantics*, pages 239 – 249, London, UK.

Alexandra Birch, Omri Abend, Ondřej Bojar, and Barry Haddow. 2016. HUME. Human UCCA-based evaluation of machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1264 – 1274, Austin, TX, USA.

Jan Buys and Phil Blunsom. 2017. Robust incremental neural semantic graph parsing. In *Proceedings* *of the 55th Meeting of the Association for Computational Linguistics*, pages 158 – 167, Vancouver, Canada.

Shu Cai and Kevin Knight. 2013. Smatch. An evaluation metric for semantic feature structures. In *Proceedings of the 51th Meeting of the Association for Computational Linguistics*, pages 748 – 752, Sofia, Bulgaria.

Ulrich Callmeier. 2002. Preprocessing and encoding techniques in PET. In Stephan Oepen, Daniel Flickinger, J. Tsujii, and Hans Uszkoreit, editors, *Collaborative Language Engineering. A Case Study in Efficient Grammar-based Processing*, pages 127 – 140. CSLI Publications, Stanford, CA.

Jie Cao, Yi Zhang, Adel Youssef, and Vivek Srikumar. 2019. Amazon at MRP 2019: Parsing meaning representations with lexical and phrasal anchoring. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 138 – 148, Hong Kong, China.

Wanxiang Che, Longxu Dou, Yang Xu, Yuxuan Wang, Yijia Liu, and Ting Liu. 2019. HIT-SCIR at MRP 2019: A unified pipeline for meaning representation parsing via efficient training and effective encoding. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 76 – 85, Hong Kong, China.

Yufei Chen, Weiwei Sun, and Xiaojun Wan. 2018. Accurate SHRG-based semantic parsing. In *Proceedings of the 56th Meeting of the Association for Computational Linguistics*, pages 408 – 418, Melbourne, Australia.

Yufei Chen, Yajie Ye, and Weiwei Sun. 2019. Peking at MRP 2019: Factorization- and composition-based parsing for Elementary Dependency Structures. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 166 – 176, Hong Kong, China.

Leshem Choshen and Omri Abend. 2018. Reference-less measure of faithfulness for grammatical error correction. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics*, New Orleans, LA, USA.

Ann Copestake, Dan Flickinger, Carl Pollard, and Ivan A. Sag. 2005. Minimal Recursion Semantics. An introduction. *Research on Language and Computation*, 3(4):281 – 332.

Marco Damonte and Shay B. Cohen. 2018. Cross-lingual abstract meaning representation parsing. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1146 – 1155, New Orleans, LA, USA.

Marco Damonte, Shay B. Cohen, and Giorgio Satta. 2017. An incremental parser for abstract meaning representation. In *Proceedings of the 15th Meeting of the European Chapter of the Association for Computational Linguistics*, pages 536 – 546, Valencia, Spain.

Robert M. W. Dixon. 2010/2012. *Basic Linguistic Theory*. Oxford University Press.

Lucia Donatelli, Meaghan Fowlie, Jonas Groschwitz, Alexander Koller, Matthias Lindemann, Mario Mina, and Pia Weißenhorn. 2019. Saarland at MRP 2019: Compositional parsing across all graphbanks. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 66 – 75, Hong Kong, China.

Rebecca Dridan and Stephan Oepen. 2011. Parser evaluation using elementary dependency matching. In *Proceedings of the 12th International Conference on Parsing Technologies*, pages 225 – 230, Dublin, Ireland.

Rebecca Dridan and Stephan Oepen. 2012. Tokenization. Returning to a long solved problem. A survey, contrastive experiment, recommendations, and toolkit. In *Proceedings of the 50th Meeting of the Association for Computational Linguistics*, pages 378 – 382, Jeju, Republic of Korea.

Kira Droganova, Andrey Kutuzov, Nikita Mediankin, and Daniel Zeman. 2019. ÚFAL–oslo at MRP 2019: Garage sale semantic parsing. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 158 – 165, Hong Kong, China.

Yantao Du, Fan Zhang, Weiwei Sun, and Xiaojun Wan. 2014. Peking: Profiling syntactic tree parsing techniques for semantic graph parsing. In *Proceedings of the 8th International Workshop on Semantic Evaluation*, pages 459 – 464, Dublin, Ireland. Association for Computational Linguistics.

Yantao Du, Fan Zhang, Xun Zhang, Weiwei Sun, and Xiaojun Wan. 2015. Peking: Building semantic dependency graphs with a hybrid parser. In *Proceedings of the 9th International Workshop on Semantic Evaluation*, pages 927 – 931, Denver, CO, USA.

Jeffrey Flanigan, Chris Dyer, Noah A. Smith, and Jaime Carbonell. 2016. CMU at SemEval-2016 task 8. Graph-based AMR parsing with infinite ramp loss. In *Proceedings of the 10th International Workshop on Semantic Evaluation*, pages 1202 – 1206, San Diego, CA, USA.

Jeffrey Flanigan, Sam Thomson, Jaime Carbonell, Chris Dyer, and Noah A. Smith. 2014. A discriminative graph-based parser for the Abstract Meaning Representation. In *Proceedings of the 52nd Meeting of the Association for Computational Linguistics*, pages 1426 – 1436, Baltimore, MD, USA.

Dan Flickinger, Stephan Oepen, and Emily M. Bender. 2017. Sustainable development and refinement of complex linguistic annotations at scale. In Nacy Ide and James Pustejovsky, editors, *Handbook of Linguistic Annotation*, pages 353 – 377. Springer, Dordrecht, The Netherlands.

W. Nelson Francis and Henry Kučera. 1982. *Frequency Analysis of English Usage. Lexicon and Grammar*. Houghton Mifflin, New York, USA.

Jan Hajič, Eva Hajičová, Jarmila Panevová, Petr Sgall, Ondřej Bojar, Silvie Cinková, Eva Fučíková, Marie Mikulová, Petr Pajas, Jan Popelka, Jiří Semecký, Jana Šindlerová, Jan Štěpánek, Josef Toman, Zdeňka Urešová, and Zdeněk Žabokrtský. 2012. Announcing Prague Czech-English Dependency Treebank 2.0. In *Proceedings of the 8th International Conference on Language Resources and Evaluation*, pages 3153 – 3160, Istanbul, Turkey.

Daniel Hershcovich, Omri Abend, and Ari Rappoport. 2017. A transition-based directed acyclic graph parser for UCCA. In *Proceedings of the 55th Meeting of the Association for Computational Linguistics*, pages 1127 – 1138, Vancouver, Canada.

Daniel Hershcovich, Omri Abend, and Ari Rappoport. 2018. Multitask parsing across semantic representations. In *Proceedings of the 56th Meeting of the Association for Computational Linguistics*, pages 373 – 385, Melbourne, Australia.

Daniel Hershcovich, Zohar Aizenbud, Leshem Choshen, Elior Sulem, Ari Rappoport, and Omri Abend. 2019. SemEval-2019 task 1. Cross-lingual semantic parsing with UCCA. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1 – 10, Minneapolis, MN, USA.

Daniel Hershcovich and Ofir Arviv. 2019. TUPA at MRP 2019: A multi-task baseline system. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 28 – 39, Hong Kong, China.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. OntoNotes. The 90% solution. In *Proceedings of Human Language Technologies: The 2006 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 57 – 60, New York City, USA.

Angelina Ivanova, Stephan Oepen, Lilja Øvrelid, and Dan Flickinger. 2012. Who did what to whom? A contrastive study of syntacto-semantic dependencies. In *Proceedings of the 6th Linguistic Annotation Workshop*, pages 2 – 11, Jeju, Republic of Korea.

Wei Jiang, Zhenghua Li, Yu Zhang, and Min Zhang. 2019. HLT@SUDA at SemEval-2019 task 1: UCCA graph parsing as constituent tree parsing. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 11 – 15, Minneapolis, MI, USA.

Jenna Kanerva, Juhani Luotolahti, and Filip Ginter. 2015. Turku: Semantic dependency parsing as a sequence classification. In *Proceedings of the 9th International Workshop on Semantic Evaluation*, pages 965 – 969, Denver, CO, USA.

Paul Kingsbury and Martha Palmer. 2002. From TreeBank to PropBank. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation*, pages 1989 – 1993, Las Palmas, Spain.

Alexander Koller, Stephan Oepen, and Weiwei Sun. 2019. Graph-based meaning representations. Design and processing. In *Proceedings of the 57th Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 6 – 11, Florence, Italy.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural AMR: Sequence-to-sequence models for parsing and generation. In *Proceedings of the 55th Meeting of the Association for Computational Linguistics*, pages 146 – 157, Vancouver, Canada.

Yuta Koreeda, Gaku Morio, Terufumi Morishita, Hiroaki Ozaki, and Kohsuke Yanai. 2019. Hitachi at MRP 2019: Unified encoder-to-biaffine network for cross-framework meaning representation parsing. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 114 – 126, Hong Kong, China.

Marco Kuhlmann and Stephan Oepen. 2016. Towards a catalogue of linguistic graph banks. *Computational Linguistics*, 42(4):819 – 827.

Sunny Lai, Chun Hei Lo, Kwong Sak Leung, and Yee Leung. 2019. CUHK at MRP 2019: Transition-based parser with cross-framework variable-arity resolve action. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 104 – 113, Hong Kong, China.

Zuchao Li, Hai Zhao, Zhuosheng Zhang, Rui Wang, Masao Utiyama, and Eiichiro Sumita. 2019. SJTU–NICT at MRP 2019: Multi-task learning for end-to-end uniform semantic graph parsing. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 45 – 54, Hong Kong, China.

Zi Lin and Nianwen Xue. 2019. Parsing meaning representations. Is easier always better? In *Proceedings of the First International Workshop on Designing Meaning Representations*, pages 34 – 43, Florence, Italy.

Matthias Lindemann, Jonas Groschwitz, and Alexander Koller. 2019. Compositional semantic parsing across graphbanks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4576 – 4585, Florence, Italy. Association for Computational Linguistics.

Chunchuan Lyu and Ivan Titov. 2018. AMR parsing as graph prediction with latent alignment. In *Proceedings of the 56th Meeting of the Association for Computational Linguistics*, pages 397 – 407, Melbourne, Australia.

Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English. The Penn Treebank. *Computational Linguistics*, 19:313 – 330.

Jonathan May. 2016. SemEval-2016 Task 8. Meaning representation parsing. In *Proceedings of the 10th International Workshop on Semantic Evaluation*, pages 1063 – 1073, San Diego, CA, USA.

Jonathan May and Jay Priyadarshi. 2017. SemEval-2017 Task 9. Abstract Meaning Representation parsing and generation. In *Proceedings of the 11th International Workshop on Semantic Evaluation*, pages 536 – 545.

Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, and Oscar Täckström. 2013. Universal dependency annotation for multilingual parsing. In *Proceedings of the 51th Meeting of the Association for Computational Linguistics*, pages 92 – 97, Sofia, Bulgaria.

Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *Proceedings of the 11th Meeting of the European Chapter of the Association for Computational Linguistics*, pages 81 – 88, Trento, Italy.

James J. McGregor. 1982. Backtrack search algorithms and the maximal common subgraph problem. *Journal of Software: Practice and Experience*, 12:23 – 34.

Yusuke Miyao, Stephan Oepen, and Daniel Zeman. 2014. In-House. An ensemble of pre-existing off-the-shelf parsers. In *Proceedings of the 8th International Workshop on Semantic Evaluation*, pages 63 – 72, Dublin, Ireland.

Seung-Hoon Na, Jinwoo Min, Kwanghyeon Park, Jong-Hun Shin, and Young-Kil Kim. 2019. Jbnu at MRP 2019: Multi-level biaffine attention for semantic dependency parsing. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 95 – 103, Hong Kong, China.

Tahira Naseem, Abhishek Shah, Hui Wan, Radu Florian, Salim Roukos, and Miguel Ballesteros. 2019. Rewarding Smatch: Transition-based AMR parsing with reinforcement learning. In *Proceedings of the 57th Meeting of the Association for Computational Linguistics*, pages 4586 – 4592, Florence, Italy.

Khoa Nguyen and Dang Nguyen. 2017. UIT-DANGNT-CLNLP at SemEval-2017 task 9: Building scientific concept fixing patterns for improving CAMR. In *Proceedings of the 11th International Workshop on Semantic Evaluation*, pages 909 – 913, Vancouver, Canada.

Joakim Nivre. 2003. An efficient algorithm for projective dependency parsing. In *Proceedings of the 8th International Conference on Parsing Technologies*, pages 149 – 160, Nancy, France.

Joakim Nivre. 2015. Towards a universal grammar of natural language processing. In *Proceedings of the 16th International Conference on Intelligent Text Processing and Computational Linguistics*, Cairo, Egypt.

Rik van Noord and Johan Bos. 2017. Dealing with co-reference in neural semantic parsing. In *Proceedings of the 2nd Workshop on Semantic Deep Learning (SemDeep-2)*, pages 41 – 49, Montpellier, France.

Stephan Oepen and Dan Flickinger. 2019. The ERG at MRP 2019: Radically compositional semantic dependencies. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 40 – 44, Hong Kong, China.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Silvie Cinková, Dan Flickinger, Jan Hajič, and Zdeňka Urešová. 2015. SemEval 2015 Task 18. Broad-coverage semantic dependency parsing. In *Proceedings of the 9th International Workshop on Semantic Evaluation*, pages 915 – 926, Denver, CO, USA.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Dan Flickinger, Jan Hajič, Angelina Ivanova, and Yi Zhang. 2014. SemEval 2014 Task 8. Broad-coverage semantic dependency parsing. In *Proceedings of the 8th International Workshop on Semantic Evaluation*, pages 63 – 72, Dublin, Ireland.

Stephan Oepen and Jan Tore Lønning. 2006. Discriminant-based MRS banking. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*, pages 1250 – 1255, Genoa, Italy.

Woodley Packard. 2012. Choosing an evaluation metric for parser design. In *Proceedings of Human Language Technologies: The 2012 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 29 – 34, Montréal, Canada.

Hao Peng, Sam Thomson, and Noah A. Smith. 2017. Deep multitask learning for semantic dependency parsing. In *Proceedings of the 55th Meeting of the*

Association for Computational Linguistics, pages 2037 – 2048, Vancouver, Canada.

Hao Peng, Sam Thomson, and Noah A. Smith. 2018a. Backpropagating through structured argmax using a SPIGOT. In *Proceedings of the 56th Meeting of the Association for Computational Linguistics*, pages 1863 – 1873, Melbourne, Australia.

Hao Peng, Sam Thomson, Swabha Swayamdipta, and Noah A. Smith. 2018b. Learning joint semantic parsers from disjoint data. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1492 – 1502, New Orleans, LA, USA.

Carl Pollard and Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar*. Studies in Contemporary Linguistics. The University of Chicago Press, Chicago, USA.

Martin Potthast, Tim Gollub, Francisco Rangel, Paolo Rosso, Efstathios Stamatatos, and Benno Stein. 2014. Improving the reproducibility of PAN's shared tasks. Plagiarism detection, author identification, and author profiling. In *Information Access Evaluation meets Multilinguality, Multimodality, and Visualization. 5th International Conference of the CLEF Initiative*, pages 268 – 299, Berlin, Germany. Springer.

Nima Pourdamghani, Yang Gao, Ulf Hermjakob, and Kevin Knight. 2014. Aligning English strings with Abstract Meaning Representation graphs. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 425 – 429, Doha, Qatar.

Sebastian Schuster and Christopher D. Manning. 2016. Enhanced English Universal Dependencies. An improved representation for natural language understanding tasks. In *Proceedings of the 10th International Conference on Language Resources and Evaluation*, Portorož, Slovenia.

Petr Sgall, Eva Hajičová, and Jarmila Panevová. 1986. *The Meaning of the Sentence and Its Semantic and Pragmatic Aspects*. D. Reidel Publishing Company, Dordrecht, The Netherlands.

Natalia Silveira, Timothy Dozat, Marie-Catherine de Marneffe, Samuel Bowman, Miriam Connor, John Bauer, and Christopher D. Manning. 2014. A gold standard dependency corpus for English. In *Proceedings of the 9th International Conference on Language Resources and Evaluation*, Reykjavik, Iceland.

Linfeng Song and Dan Gildea. 2019. SemBleu: A robust metric for AMR parsing evaluation. In *Proceedings of the 57th Meeting of the Association for Computational Linguistics*, pages 4547 – 4552, Florence, Italy.

Gabriel Stanovsky and Ido Dagan. 2018. Semantics as a foreign language. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2412 – 2421, Brussels, Belgium.

Mark Steedman. 2011. *Taking Scope*. MIT Press, Cambridge, MA, USA.

Milan Straka. 2018. UDPipe 2.0 prototype at CoNLL 2018 UD shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 197 – 207.

Milan Straka and Jana Straková. 2019. ÚFAL MRPipe at MRP 2019: UDPipe goes semantic in the Meaning Representation Parsing shared task. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 127 – 137, Hong Kong, China.

Milan Straka, Jana Straková, and Jan Hajič. 2019. Evaluating Contextualized Embeddings on 54 Languages in POS Tagging, Lemmatization and Dependency Parsing. *arXiv preprint arXiv:1908.07448*.

Elior Sulem, Omri Abend, and Ari Rappoport. 2015. Conceptual annotations preserve structure across translations. A French–English case study. In *Proceedings of the 1st Workshop on Semantics-Driven Statistical Machine Translation*, pages 11 – 22.

Elior Sulem, Omri Abend, and Ari Rappoport. 2018a. Semantic structural annotation for text simplification. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics*, New Orleans, LA, USA.

Elior Sulem, Omri Abend, and Ari Rappoport. 2018b. Simple and effective text simplification using semantic and neural methods. In *Proceedings of the 56th Meeting of the Association for Computational Linguistics*, Melbourne, Australia.

Yuka Tateisi, Akane Yakushiji, Tomoko Ohta, and J. Tsujii. 2005. Syntax annotation for the GENIA corpus. In *Proceedings of the 2nd International Joint Conference on Natural Language Processing*, pages 220 – 225, Jeju, Korea.

Zdeňka Urešová, Eva Fučíková, and Jana Šindlerová. 2016. CzEngVallex. A bilingual Czech–English valency lexicon. *The Prague Bulletin of Mathematical Linguistics*, 105:17 – 50.

Chuan Wang, Bin Li, and Nianwen Xue. 2018. Transition-based Chinese AMR parsing. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 247 – 252, New Orleans, LA, USA.

Chuan Wang, Sameer Pradhan, Xiaoman Pan, Heng Ji, and Nianwen Xue. 2016. CAMR at SemEval-2016 task 8: An extended transition-based AMR parser. In *Proceedings of the 10th International Workshop on Semantic Evaluation*, pages 1173 – 1178, San Diego, CA, USA.

Chuan Wang and Nianwen Xue. 2017. Getting the most out of AMR parsing. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1257 – 1268, Copenhagen, Denmark.

Chuan Wang, Nianwen Xue, and Sameer Pradhan. 2015. A transition-based algorithm for AMR parsing. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 366 – 375, Denver, CO, USA.

Xinyu Wang, Yixian Liu, Zixia Jia, Chengyue Jiang, and Kewei Tu. 2019. ShanghaiTech at MRP 2019: Sequence-to-graph transduction with second-order edge inference for cross-framework meaning representation parsing. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 55 – 65, Hong Kong, China.

Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of the 8th International Conference on Parsing Technologies*, pages 195 – 206, Nancy, France.

Sheng Zhang, Xutai Ma, Kevin Duh, and Benjamin Van Durme. 2019a. AMR parsing as sequence-to-graph transduction. In *Proceedings of the 57th Meeting of the Association for Computational Linguistics*, pages 80 – 94, Florence, Italy.

Sheng Zhang, Xutai Ma, Kevin Duh, and Benjamin Van Durme. 2019b. Broad-coverage semantic parsing as transduction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, Hong Kong, China.

Yue Zhang, Wei Jiang, Qingrong Xia, Junjie Cao, Rui Wang, Zhenghua Li, and Min Zhang. 2019c. Suda–alibaba at MRP 2019: Graph-based models with BERT. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 149 – 157, Hong Kong, China.