MRP 2019: Cross-Framework Meaning Representation Parsing

http://mrp.nlpl.eu

Stephan Oepen[♣], Omri Abend[♠], **Jan Hajič**[♡],
Daniel Hershcovich[♢], Marco Kuhlmann[○],
Tim O'Gorman[⋆], and Nianwen Xue[•]

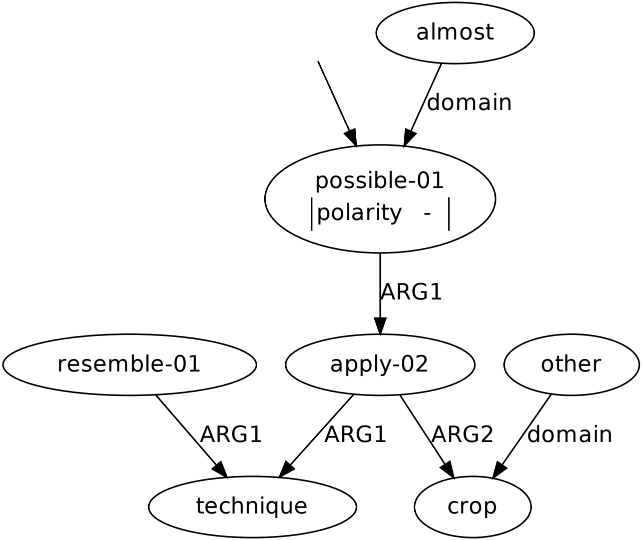
♣ University of Oslo, Department of Informatics
♠ The Hebrew University of Jerusalem, School of Computer Science and Engineering
♡ Charles University in Prague, Institute of Formal and Applied Linguistics
♢ University of Copenhagen, Department of Computer Science
○ Linköping University, Department of Computer and Information Science
⋆ University of Colorado at Boulder, Department of Linguistics
• Brandeis University, Department of Computer Science

mrp-organizers@nlpl.eu

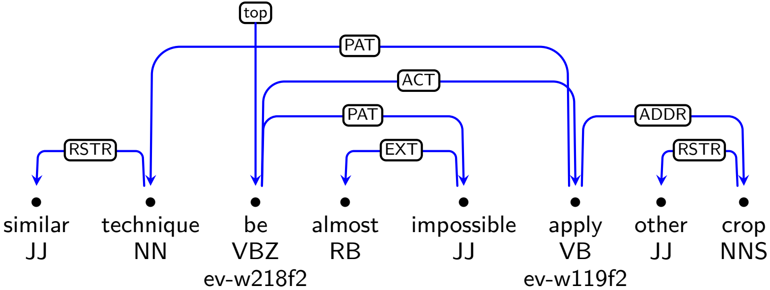
10,000-Meter Perspective: Parsing into Semantic Graphs

A similar technique is almost impossible to apply to other crops.

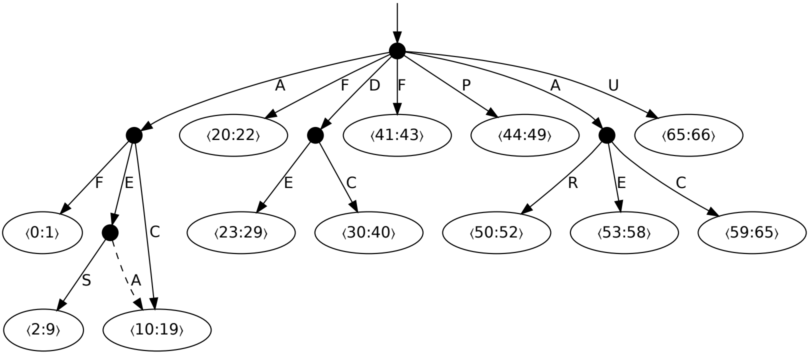
10,000-Meter Perspective: Parsing into Semantic Graphs



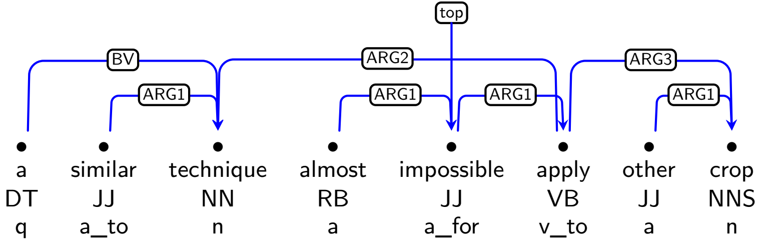
10,000-Meter Perspective: Parsing into Semantic Graphs



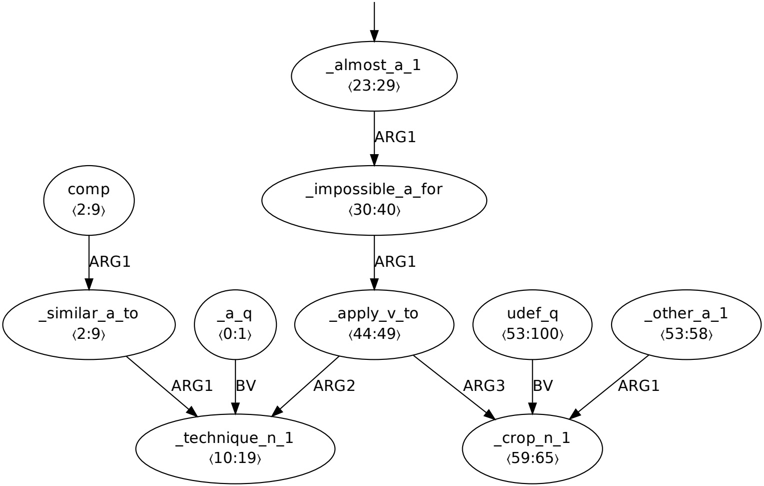
10,000-Meter Perspective: Parsing into Semantic Graphs



10,000-Meter Perspective: Parsing into Semantic Graphs



10,000-Meter Perspective: Parsing into Semantic Graphs

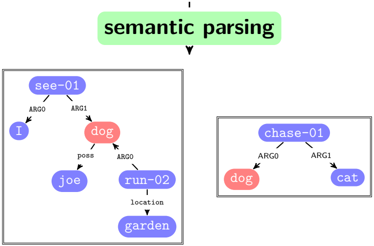


Why Graph-Based Meaning Representation?

I saw Joe's dog, which was running in the garden.
The dog was chasing a cat.

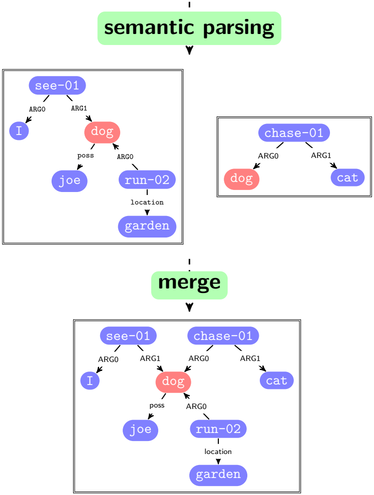
Why Graph-Based Meaning Representation?

I saw Joe's dog, which was running in the garden.
The dog was chasing a cat.



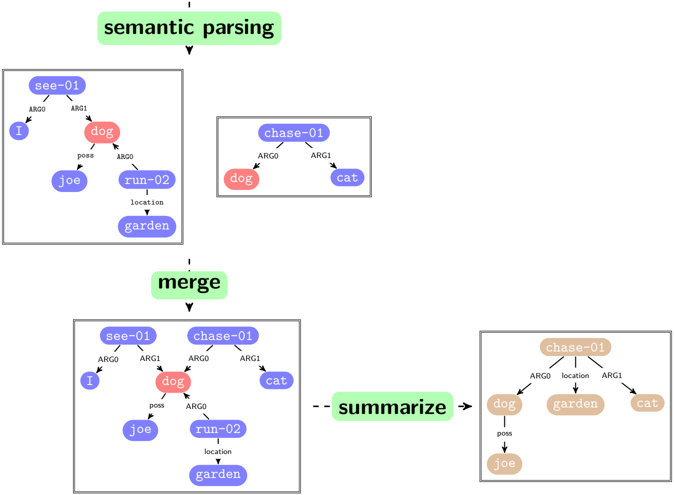
Why Graph-Based Meaning Representation?

I saw Joe's dog, which was running in the garden.
The dog was chasing a cat.

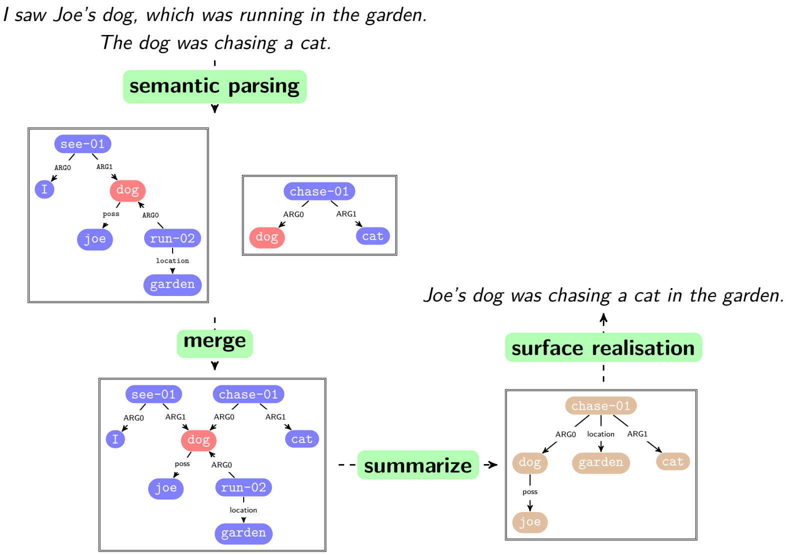


Why Graph-Based Meaning Representation?

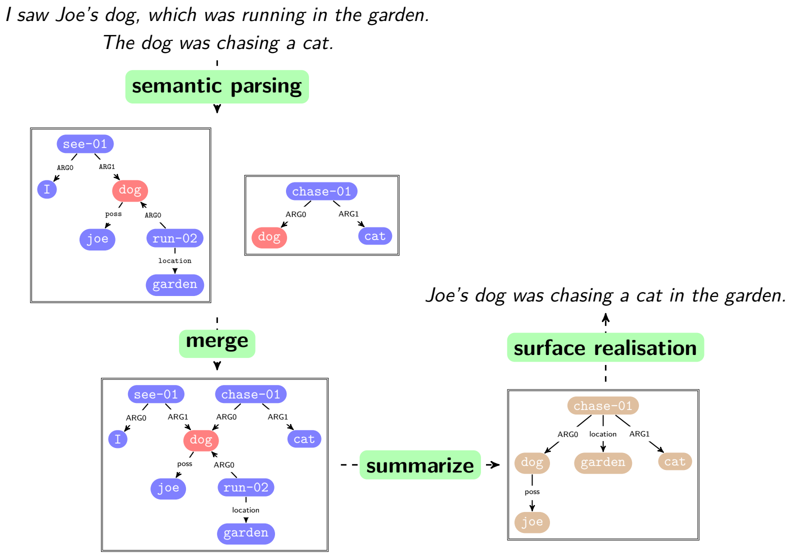
I saw Joe's dog, which was running in the garden.
The dog was chasing a cat.



Why Graph-Based Meaning Representation?

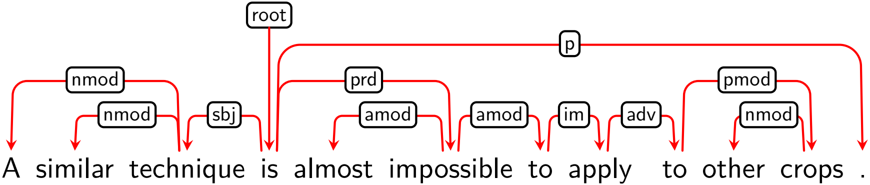


Why Graph-Based Meaning Representation?

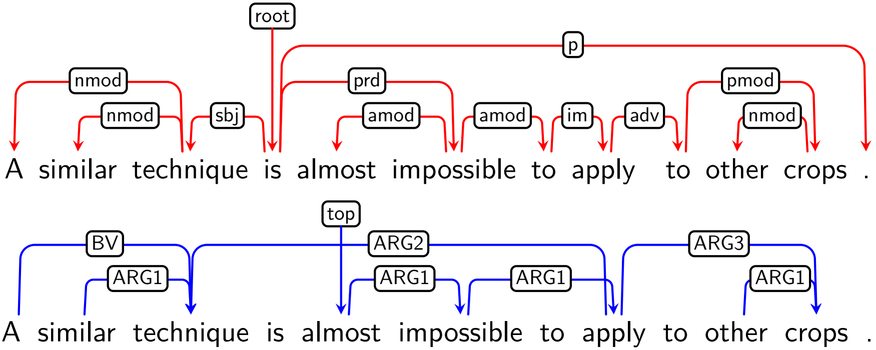


Hardy & Vlachos (2018): 2^+ ROUGE points over strong encoder–decoder.

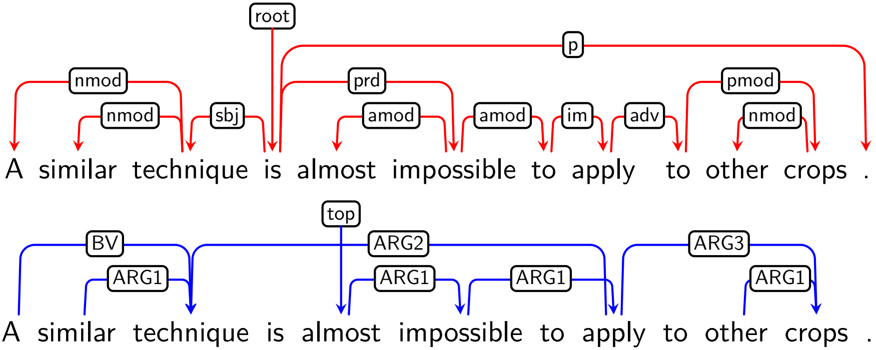
Syntactic Trees vs. Semantic Graphs



Syntactic Trees vs. Semantic Graphs



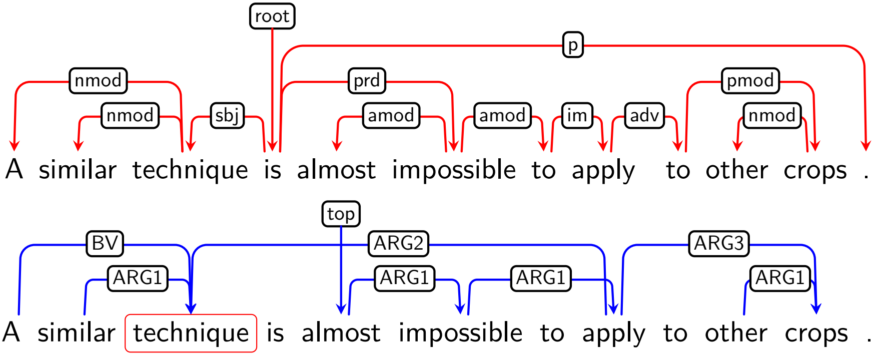
Syntactic Trees vs. Semantic Graphs



$$\exists x : \text{technique'}(x) \wedge \text{similar'}(x), \exists y : \text{crop'}(y) \wedge \text{other'}(y)$$

$$\rightarrow \text{almost'}(\neg\text{possible'}(\text{apply'}(\_, x, y)))$$

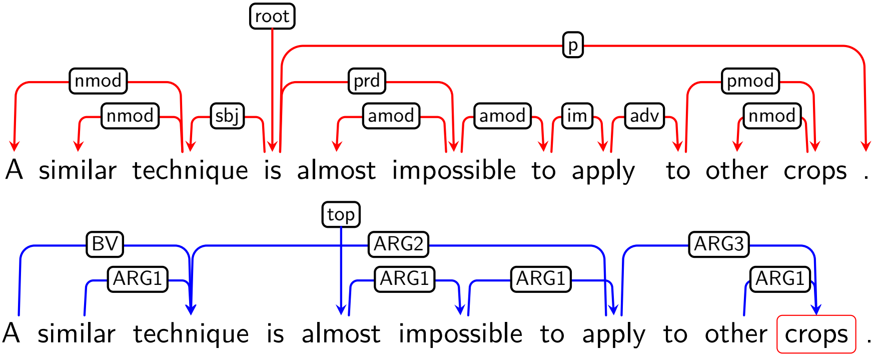
Syntactic Trees vs. Semantic Graphs



$$\exists x : \text{technique'}(x) \wedge \text{similar'}(x), \exists y : \text{crop'}(y) \wedge \text{other'}(y)$$

$$\rightarrow \text{almost'}(\neg\text{possible'}(\text{apply'}(\_, x, y)))$$

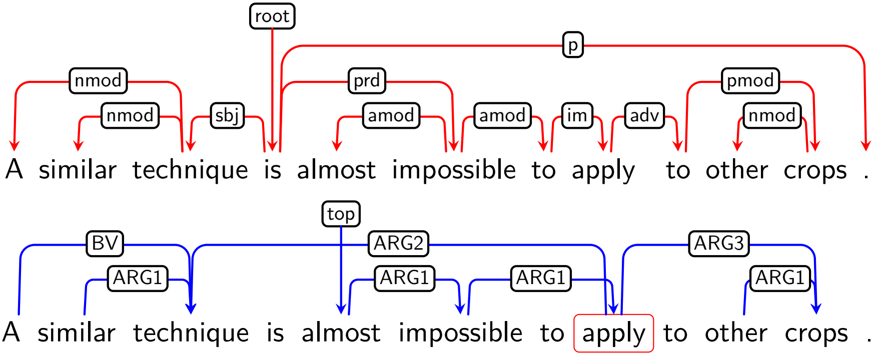
Syntactic Trees vs. Semantic Graphs



$$\exists x : \text{technique'}(x) \land \text{similar'}(x), \exists y : \text{crop'}(y) \land \text{other'}(y)$$

$$\rightarrow \text{almost'}(\neg\text{possible'}(\text{apply'}(\_, x, y)))$$

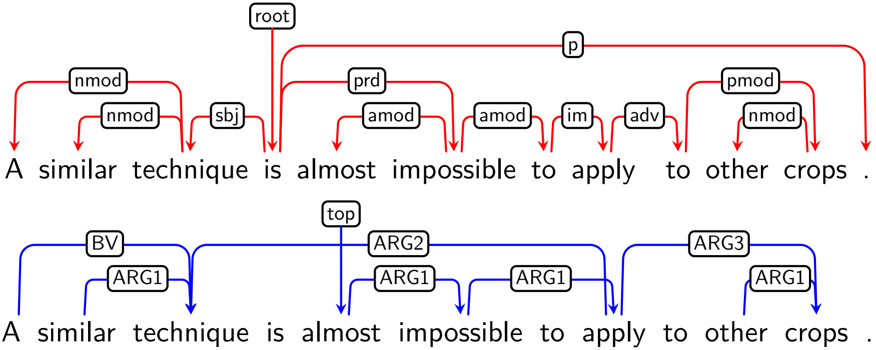
Syntactic Trees vs. Semantic Graphs



$$\exists x : \text{technique'}(x) \land \text{similar'}(x), \exists y : \text{crop'}(y) \land \text{other'}(y)$$

$$\rightarrow \text{almost'}(\neg\text{possible'}(\text{apply'}(\_, x, y)))$$

Syntactic Trees vs. Semantic Graphs



$$\exists x : \text{technique'}(x) \wedge \text{similar'}(x), \exists y : \text{crop'}(y) \wedge \text{other'}(y)$$
$$\rightarrow \text{almost'}(\neg\text{possible'}(\text{apply'}(\_, x, y)))$$

Different Desiderata and Levels of Abstraction

- Grammaticality (e.g. subject–verb agreement) vs. relational structure.

Semi-Formally: Trees vs. Graphs

Structural Wellformedness Conditions on Trees

- Unique root, connected, single parent, free of cycles; maybe: projective;

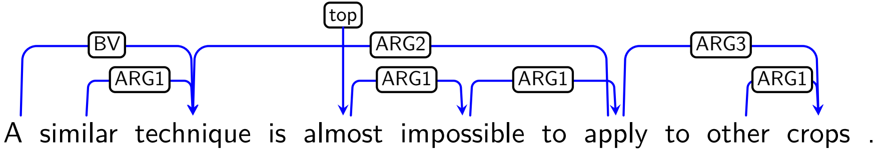
$\rightarrow$ all nodes (but the root) reachable by unique directed path from root.

Semi-Formally: Trees vs. Graphs

Structural Wellformedness Conditions on Trees

- Unique root, connected, single parent, free of cycles; maybe: projective;

$\rightarrow$ all nodes (but the root) reachable by unique directed path from root.

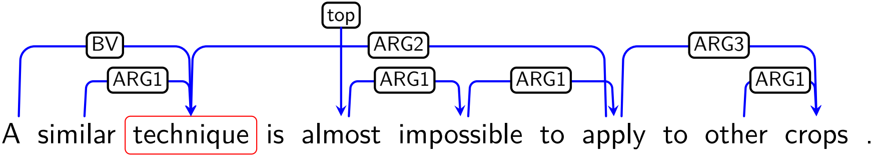


Semi-Formally: Trees vs. Graphs

Structural Wellformedness Conditions on Trees

- Unique root, connected, single parent, free of cycles; maybe: projective;

$\rightarrow$ all nodes (but the root) reachable by unique directed path from root.



Beyond Trees: General Graphs

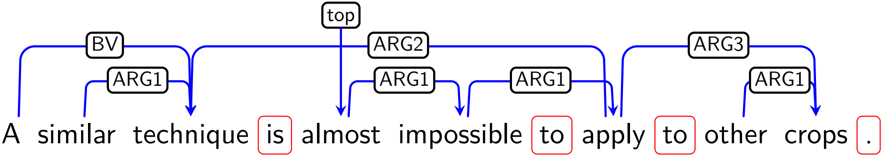
- Argument sharing: nodes with multiple incoming edges (***in***-degree > 1);

Semi-Formally: Trees vs. Graphs

Structural Wellformedness Conditions on Trees

- Unique root, connected, single parent, free of cycles; maybe: projective;

$\rightarrow$ all nodes (but the root) reachable by unique directed path from root.



Beyond Trees: General Graphs

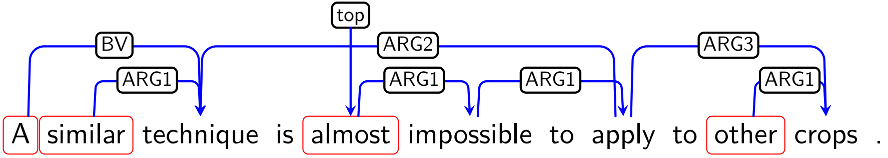
- Argument sharing: nodes with multiple incoming edges (***in***-degree > 1);
- some surface tokens do not contribute (as nodes; many function words);

Semi-Formally: Trees vs. Graphs

Structural Wellformedness Conditions on Trees

- Unique root, connected, single parent, free of cycles; maybe: projective;

→ all nodes (but the root) reachable by unique directed path from root.



Beyond Trees: General Graphs

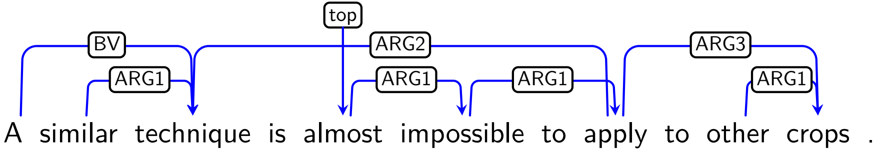
- Argument sharing: nodes with multiple incoming edges (***in***-degree > 1);
- some surface tokens do not contribute (as nodes; many function words);
- (structurally) multi-rooted: more than one node with zero ***in***-degree;

Semi-Formally: Trees vs. Graphs

Structural Wellformedness Conditions on Trees

- Unique root, connected, single parent, free of cycles; maybe: projective;

$\rightarrow$ all nodes (but the root) reachable by unique directed path from root.



Beyond Trees: General Graphs

- Argument sharing: nodes with multiple incoming edges (***in***-degree > 1);
- some surface tokens do not contribute (as nodes; many function words);
- (structurally) multi-rooted: more than one node with zero ***in***-degree;

$\rightarrow$ massive growth in modeling and algorithmic complexity (NP-complete).

High-Level Goals of the Shared Task

Cross-Framework Comparability and Interoperability

- Vast, complex landscape of representing natural language meaning;
- diverse linguistic traditions, modeling assumptions, levels of ambition;

→ clarify concepts and terminology; unify representations and evaluation.

High-Level Goals of the Shared Task

Cross-Framework Comparability and Interoperability

- Vast, complex landscape of representing natural language meaning;
- diverse linguistic traditions, modeling assumptions, levels of ambition;

→ clarify concepts and terminology; unify representations and evaluation.

Parsing into Graph-Structured Representations

- Cottage industry of parsers with output structures beyond rooted trees;
- distinct techniques, e.g. based on transitions, composition, 'translation';
- much framework-internal evolution: design reflects specific assumptions;

→ evaluate across frameworks; learning from complementary knowledge.

High-Level Goals of the Shared Task

Cross-Framework Comparability and Interoperability

- Vast, complex landscape of representing natural language meaning;
- diverse linguistic traditions, modeling assumptions, levels of ambition;

→ clarify concepts and terminology; unify representations and evaluation.

Parsing into Graph-Structured Representations

- Cottage industry of parsers with output structures beyond rooted trees;
- distinct techniques, e.g. based on transitions, composition, ‘translation’;
- much framework-internal evolution: design reflects specific assumptions;

→ evaluate across frameworks; learning from complementary knowledge.

Learning from Complementary Knowledge

- Cross-Framework Perspective: Seek commonality and complementarity.

$$\mathbb{G} = \langle N, E, T \rangle$$

- $\mathbb{G}$ is a directed graph: N is set of nodes; $E \subseteq N \times N$ is set of edges;
- $T \subseteq N$ is possibly empty set of top node(s): the 'main' predicate(s);

$$\mathbb{G} = \langle N, E, T \rangle$$

- $\mathbb{G}$ is a directed graph: N is set of nodes; $E \subseteq N \times N$ is set of edges;
- $T \subseteq N$ is possibly empty set of top node(s): the ‘main’ predicate(s);
- ***in***- and ***out***-degree of $n \in N$ count edges to and from n; ***in*** $= 0$: root;
- top in *Abrams arrived quickly.* is *arrive*, but can be argument of *quick*;
- semantic graphs often multi-rooted: rootness just a structural property;

$$\mathbb{G} = \langle N, E, T \rangle$$

- $\mathbb{G}$ is a directed graph: N is set of nodes; $E \subseteq N \times N$ is set of edges;
- $T \subseteq N$ is possibly empty set of top node(s): the 'main' predicate(s);
- ***in***- and ***out***-degree of $n \in N$ count edges to and from n; ***in*** $= 0$: root;
- top in *Abrams arrived quickly.* is *arrive*, but can be argument of *quick*;
- semantic graphs often multi-rooted: rootness just a structural property;
- a node n is reentrant if ***in***$(n) > 1$ (shared argument across predicates);

$$\mathbb{G} = \langle N, E, T \rangle$$

- $\mathbb{G}$ is a directed graph: N is set of nodes; $E \subseteq N \times N$ is set of edges;
- $T \subseteq N$ is possibly empty set of top node(s): the 'main' predicate(s);
- ***in***- and ***out***-degree of $n \in N$ count edges to and from n; ***in*** $= 0$: root;
- top in *Abrams arrived quickly.* is *arrive*, but can be argument of *quick*;
- semantic graphs often multi-rooted: rootness just a structural property;
- a node n is reentrant if ***in***$(n) > 1$ (shared argument across predicates);
- cycles can occur: directed path from m to n and ('back') from n to m;

$$\mathbb{G} = \langle N, E, T \rangle$$

- $\mathbb{G}$ is a directed graph: N is set of nodes; $E \subseteq N \times N$ is set of edges;
- $T \subseteq N$ is possibly empty set of top node(s): the 'main' predicate(s);
- ***in***- and ***out***-degree of $n \in N$ count edges to and from n; ***in*** $= 0$: root;
- top in *Abrams arrived quickly.* is *arrive*, but can be argument of *quick*;
- semantic graphs often multi-rooted: rootness just a structural property;
- a node n is reentrant if ***in***$(n) > 1$ (shared argument across predicates);
- cycles can occur: directed path from m to n and ('back') from n to m;
- $\mathbb{G}$ is connected if there is an undirected path between all pairs of nodes;

$$\mathbb{G} = \langle N, E, T \rangle$$

- $\mathbb{G}$ is a directed graph: N is set of nodes; $E \subseteq N \times N$ is set of edges;
- $T \subseteq N$ is possibly empty set of top node(s): the 'main' predicate(s);
- ***in***- and ***out***-degree of $n \in N$ count edges to and from n; ***in*** $= 0$: root;
- top in *Abrams arrived quickly.* is *arrive*, but can be argument of *quick*;
- semantic graphs often multi-rooted: rootness just a structural property;
- a node n is reentrant if ***in***$(n) > 1$ (shared argument across predicates);
- cycles can occur: directed path from m to n and ('back') from n to m;
- $\mathbb{G}$ is connected if there is an undirected path between all pairs of nodes;
- $\mathbb{G}$ is a tree if $|T| = 1$ and there is a unique path to all other nodes.

Anchoring in the Surface String

Relating Pieces of Meaning to the Linguistic Signal

- Intuitively, sub-structures of meaning relate to sub-parts of the input;
- semantic frameworks vary in how much weight to put on this relation;

Anchoring in the Surface String

Relating Pieces of Meaning to the Linguistic Signal

- Intuitively, sub-structures of meaning relate to sub-parts of the input;
- semantic frameworks vary in how much weight to put on this relation;
- anchoring of graph elements in sub-strings of the underlying utterance;
- can be part of semantic annotations or not; can take different forms;

Anchoring in the Surface String

Relating Pieces of Meaning to the Linguistic Signal

- Intuitively, sub-structures of meaning relate to sub-parts of the input;
- semantic frameworks vary in how much weight to put on this relation;
- anchoring of graph elements in sub-strings of the underlying utterance;
- can be part of semantic annotations or not; can take different forms;
- hierarchy of anchoring types: Flavor (0)–(2); bilexical graphs strictest;

Flavor	Name	Example	Type of Anchoring
(0)	bilexical	DM, PSD	nodes are sub-set of surface tokens
(1)	anchored	EDS, UCCA	free node–sub-string correspondences
(2)	unanchored	AMR	no explicit sub-string correspondences

Anchoring in the Surface String

Relating Pieces of Meaning to the Linguistic Signal

- Intuitively, sub-structures of meaning relate to sub-parts of the input;
- semantic frameworks vary in how much weight to put on this relation;
- anchoring of graph elements in sub-strings of the underlying utterance;
- can be part of semantic annotations or not; can take different forms;
- hierarchy of anchoring types: Flavor (0)–(2); bilexical graphs strictest;
- anchoring central in parsing, explicit or latent; aka 'alignment' for AMR;
- relevant to at least some downstream tasks; should impact evaluation.

Flavor	Name	Example	Type of Anchoring
(0)	bilexical	DM, PSD	nodes are sub-set of surface tokens
(1)	anchored	EDS, UCCA	free node–sub-string correspondences
(2)	unanchored	AMR	no explicit sub-string correspondences

A Selection of Semantic Graphbanks

Selection Criteria

- 'Full-sentence' semantics: all content-bearing units receive annotations;
- natively graph-based: meaning representation through (directed) graphs;
- large-scale, gold-standard annotations and parsers are publicly available;

A Selection of Semantic Graphbanks

Selection Criteria

- 'Full-sentence' semantics: all content-bearing units receive annotations;
- natively graph-based: meaning representation through (directed) graphs;
- large-scale, gold-standard annotations and parsers are publicly available;

→ five distinct frameworks, bi-lexical to unanchored; sadly, English only.

A Selection of Semantic Graphbanks

Selection Criteria

- 'Full-sentence' semantics: all content-bearing units receive annotations;
- natively graph-based: meaning representation through (directed) graphs;
- large-scale, gold-standard annotations and parsers are publicly available;

→ five distinct frameworks, bi-lexical to unanchored; sadly, English only.

A Selection of Semantic Graphbanks

Selection Criteria

- ‘Full-sentence’ semantics: all content-bearing units receive annotations;
- natively graph-based: meaning representation through (directed) graphs;
- large-scale, gold-standard annotations and parsers are publicly available;

→ five distinct frameworks, bi-lexical to unanchored; sadly, English only.

(With Apologies to) Non-Graph or Non-Meaning Banks

- PropBank (Palmer et al., 2005), Framenet (Baker et al., 1998), . . . ;
- Groningen Parallel Meaning Bank: GMB, PMB (Basile et al., 2012);
- Universal Decompositional Semantics (White et al., 2016);
- Enhanced Universal Dependencies (Schuster & Manning, 2016);
- . . .

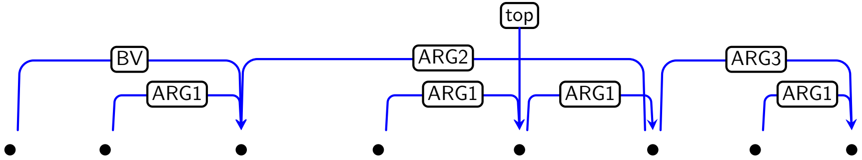
Arguably Basicest: Bi-lexical Semantic Dependencies

- Two decades of great advances in syntactic dependencies and parsing;
- recently, renewed interest in meaning; algorithmic interest in graphs;

Arguably Basicest: Bi-lexical Semantic Dependencies

- Two decades of great advances in syntactic dependencies and parsing;
- recently, renewed interest in meaning; algorithmic interest in graphs;
- nodes limited to surface lexical units (words):

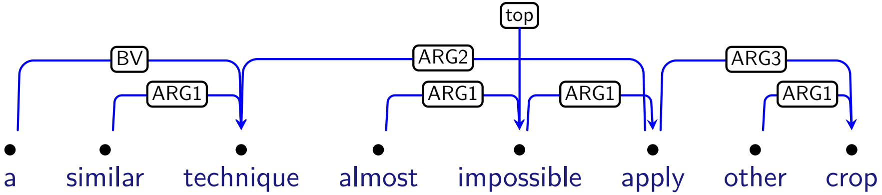
A similar technique is almost impossible to apply to other crops.



Arguably Basicest: Bi-lexical Semantic Dependencies

- Two decades of great advances in syntactic dependencies and parsing;
- recently, renewed interest in meaning; algorithmic interest in graphs;
- nodes limited to surface lexical units (words): lemmas

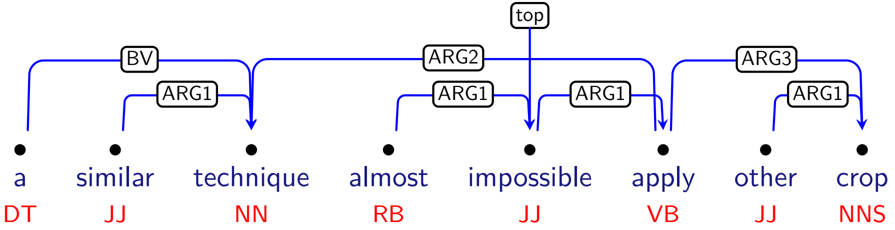
A similar technique is almost impossible to apply to other crops.



Arguably Basicest: Bi-lexical Semantic Dependencies

- Two decades of great advances in syntactic dependencies and parsing;
- recently, renewed interest in meaning; algorithmic interest in graphs;
- nodes limited to surface lexical units (words): lemmas, PoS

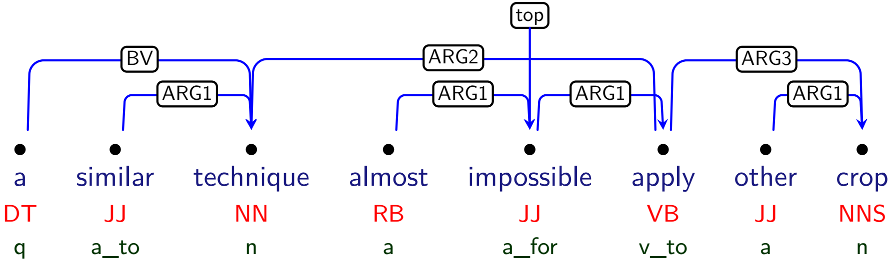
A similar technique is almost impossible to apply to other crops.



Arguably Basicest: Bi-lexical Semantic Dependencies

- Two decades of great advances in syntactic dependencies and parsing;
- recently, renewed interest in meaning; algorithmic interest in graphs;
- nodes limited to surface lexical units (words): lemmas, PoS, frames;

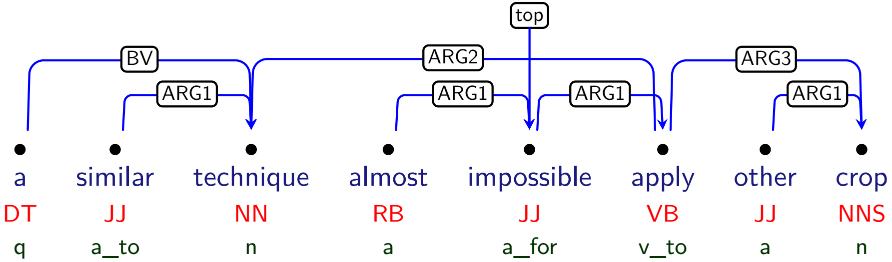
A similar technique is almost impossible to apply to other crops.



Arguably Basicest: Bi-lexical Semantic Dependencies

- Two decades of great advances in syntactic dependencies and parsing;
- recently, renewed interest in meaning; algorithmic interest in graphs;
- nodes limited to surface lexical units (words): lemmas, PoS, frames;
- edges encode argument roles and maybe some construction semantics;

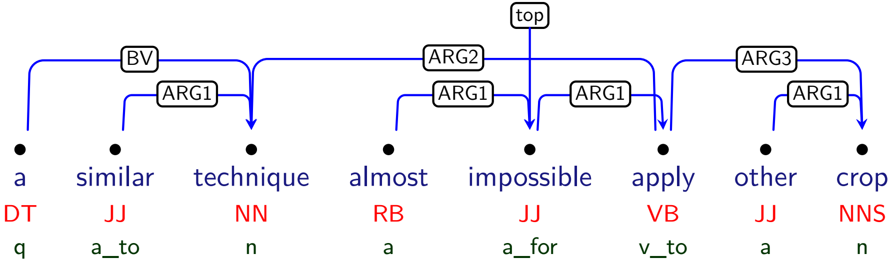
A similar technique is almost impossible to apply to other crops.



Arguably Basicest: Bi-lexical Semantic Dependencies

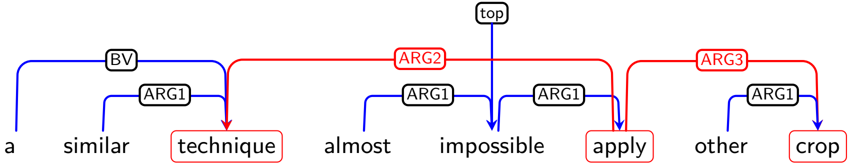
- Two decades of great advances in syntactic dependencies and parsing;
- recently, renewed interest in meaning; algorithmic interest in graphs;
- nodes limited to surface lexical units (words): lemmas, PoS, frames;
- edges encode argument roles and maybe some construction semantics;
- limited expressivity, e.g. no lexical decomposition, no covert meaning.

A similar technique is almost impossible to apply to other crops.



DM: DELPH-IN MRS Bi-Lexical Dependencies (Ivanova et al., 2012)

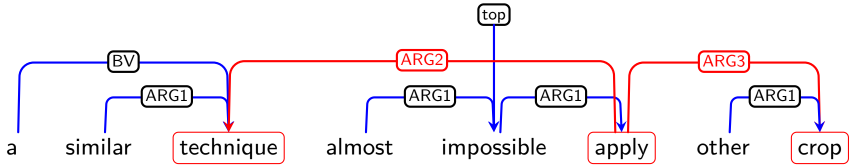
- Simplification from underspecified logical forms (ERS; coming later);



(0) Two Bi-Lexical Frameworks: DM & PSD

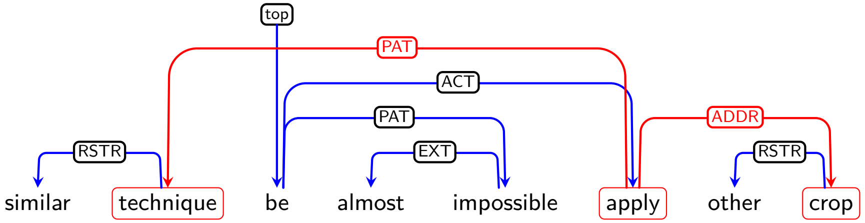
DM: DELPH-IN MRS Bi-Lexical Dependencies (Ivanova et al., 2012)

- Simplification from underspecified logical forms (ERS; coming later);



PSD: Prague Semantic Dependencies (Hajič et al., 2012)

- Simplification from FGD tectogrammatical trees (Sgall et al., 1986).



(1) Elementary Dependency Structures (EDS)

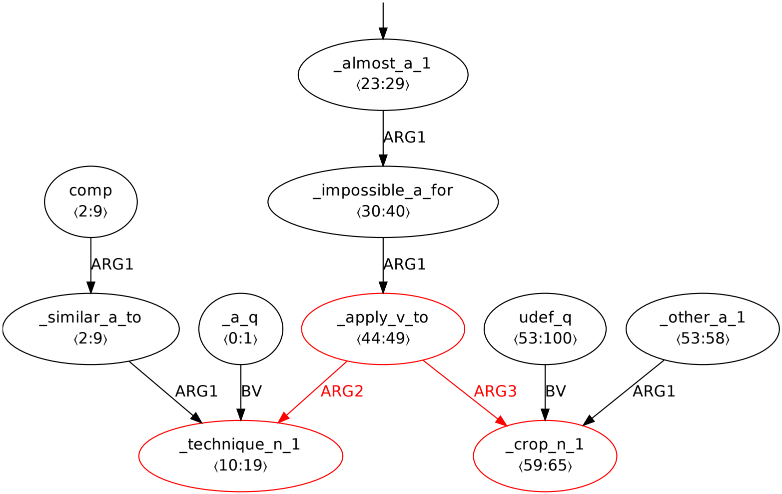
Break Free of Bi-Lexical Limitations (Oepen & Lønning, 2006)

- Decomposition or construction meaning; anchors: arbitrary sub-strings.

(1) Elementary Dependency Structures (EDS)

Break Free of Bi-Lexical Limitations (Oepen & Lønning, 2006)

- Decomposition or construction meaning; anchors: arbitrary sub-strings.

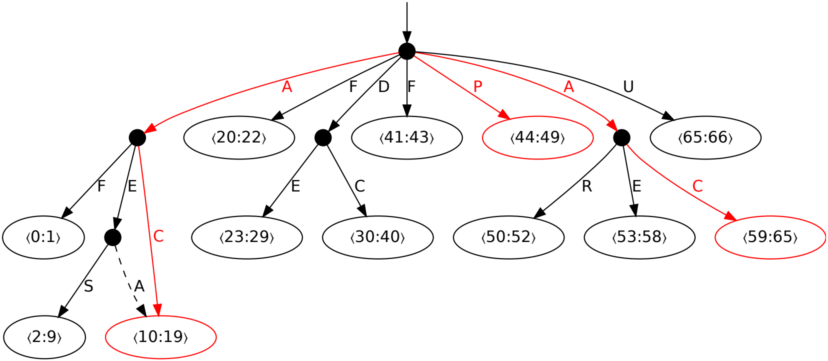


A similar technique is almost impossible to apply to other crops.

(1) Universal Conceptual Cognitive Annotation (UCCA)

Multi-Layered Design (Abend & Rappoport, 2013); Foundational Layer

- Tree backbone: semantic 'constituents' are scenes ('clauses') and units;

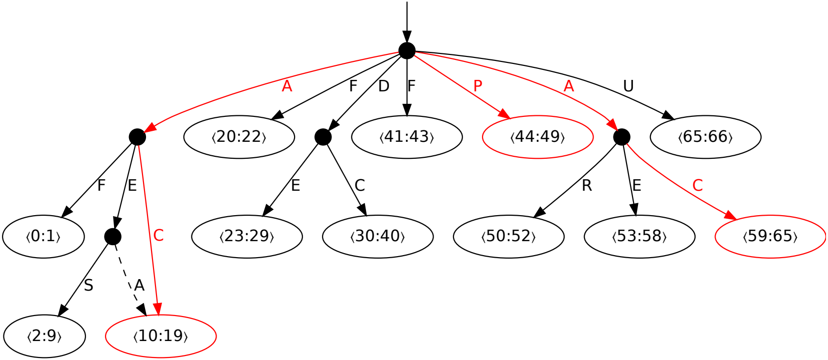


A similar technique is almost impossible to apply to other crops.

(1) Universal Conceptual Cognitive Annotation (UCCA)

Multi-Layered Design (Abend & Rappoport, 2013); Foundational Layer

- Tree backbone: semantic 'constituents' are scenes ('clauses') and units;
- scenes (Process or State): pArticipants and aDverbials (plus F and U);

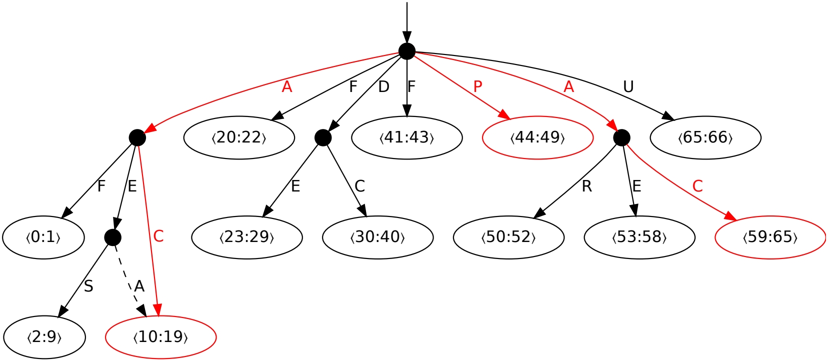


A similar technique is almost impossible to apply to other crops.

(1) Universal Conceptual Cognitive Annotation (UCCA)

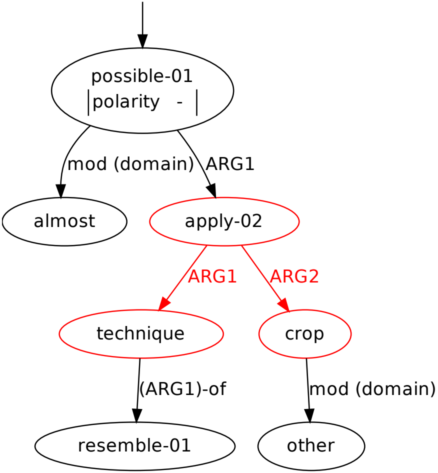
Multi-Layered Design (Abend & Rappoport, 2013); Foundational Layer

- Tree backbone: semantic ‘constituents’ are scenes (‘clauses’) and units;
- scenes (Process or State): pArticipants and aDverbials (plus F and U);
- complex units distinguish Center and Elaborator(s); allow remote edges.



A similar technique is almost impossible to apply to other crops.

(2) Abstract Meaning Representation (AMR)



Banarescu et al. (2013)

- Abstractly (if not linguistically) similar to EDS, but unanchored;
- verbal senses from PropBank^{++};
- negation as node-local property;
- tree-like annotation: inversed edges normalized for evaluation;
- originally designed for (S)MT; various NLU applications to date.

A similar technique is almost impossible to apply to other crops.

Training and Evaluation Data in the Shared Task

		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

- DM, PSD, and ESD annotate the same text (Sections 00–20 of WSJ);
- UCCA: samples of EWT & Wikipedia; AMR: twelve different sources;

Training and Evaluation Data in the Shared Task

		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

- DM, PSD, and ESD annotate the same text (Sections 00–20 of WSJ);
- UCCA: samples of EWT & Wikipedia; AMR: twelve different sources;
- linguistics: 100-item WSJ sample in all frameworks publicly available;

Training and Evaluation Data in the Shared Task

		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

- DM, PSD, and ESD annotate the same text (Sections 00–20 of WSJ);
- UCCA: samples of EWT & Wikipedia; AMR: twelve different sources;
- linguistics: 100-item WSJ sample in all frameworks publicly available;
- evaluation: subset of 100 sentences from *The Little Prince* is public.

Graphbank Statistics (Kuhlmann & Oepen, 2016)

			DM	PSD	EDS	UCCA	AMR^{-1}
counts	(01)	number of graphs	35,656	35,656	35,656	6,572	56,240
	(01)	number of tokens	802,717	802,717	802,717	138,268	1,000,217
	(02)	average number of tokens	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)	$\%_g$ 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
	(14)	percentage non-top roots	44.94	4.34	54.85	0.00	20.09
order	(15)	average edge length	2.684	3.320	–	–	–
	(16)	$\%_g$ noncrossing	69.21	64.61	–	–	–
	(17)	$\%_g$ pagenumber two	99.59	98.08	–	–	–

Graphbank Statistics (Kuhlmann & Oepen, 2016)

			DM	PSD	EDS	UCCA	AMR^{-1}
counts	(01)	number of graphs	35,656	35,656	35,656	6,572	56,240
	(01)	number of tokens	802,717	802,717	802,717	138,268	1,000,217
	(02)	average number of tokens	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)	$\%_g$ 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
	(14)	percentage non-top roots	44.94	4.34	54.85	0.00	20.09
order	(15)	average edge length	2.684	3.320	–	–	–
	(16)	$\%_g$ noncrossing	69.21	64.61	–	–	–
	(17)	$\%_g$ pagenumber two	99.59	98.08	–	–	–

Graphbank Statistics (Kuhlmann & Oepen, 2016)

			DM	PSD	EDS	UCCA	AMR^{-1}
counts	(01)	number of graphs	35,656	35,656	35,656	6,572	56,240
	(01)	number of tokens	802,717	802,717	802,717	138,268	1,000,217
	(02)	average number of tokens	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)	$\%_g$ 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
	(14)	percentage non-top roots	44.94	4.34	54.85	0.00	20.09
order	(15)	average edge length	2.684	3.320	–	–	–
	(16)	$\%_g$ noncrossing	69.21	64.61	–	–	–
	(17)	$\%_g$ pagenumber two	99.59	98.08	–	–	–

Graphbank Statistics (Kuhlmann & Oepen, 2016)

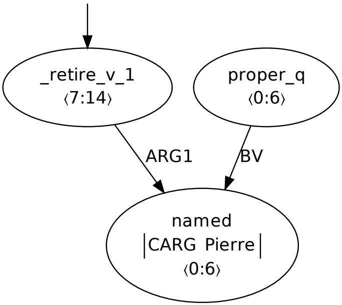
			DM	PSD	EDS	UCCA	AMR^{-1}
counts	(01)	number of graphs	35,656	35,656	35,656	6,572	56,240
	(01)	number of tokens	802,717	802,717	802,717	138,268	1,000,217
	(02)	average number of tokens	22.51	22.51	22.51	21.03	17,78
	(03)	average nodes per token	0.77	0.64	1.29	1.37	0.65
	(04)	number of edge labels	59	90	10	15	101
treeness	(05)	$\%_g$ 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
	(14)	percentage non-top roots	44.94	4.34	54.85	0.00	20.09
order	(15)	average edge length	2.684	3.320	–	–	–
	(16)	$\%_g$ noncrossing	69.21	64.61	–	–	–
	(17)	$\%_g$ pagenumber two	99.59	98.08	–	–	–

Graphbank Statistics (Kuhlmann & Oepen, 2016)

			DM	PSD	EDS	UCCA	AMR^{-1}
counts	(01)	number of graphs	35,656	35,656	35,656	6,572	56,240
	(01)	number of tokens	802,717	802,717	802,717	138,268	1,000,217
	(02)	average number of tokens	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)	$\%_g$ 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
	(14)	percentage non-top roots	44.94	4.34	54.85	0.00	20.09
order	(15)	average edge length	2.684	3.320	–	–	–
	(16)	$\%_g$ noncrossing	69.21	64.61	–	–	–
	(17)	$\%_g$ pagenumber two	99.59	98.08	–	–	–

Cross-Framework Evaluation: MRP Graph Similarity

- Break down graphs into types of information: per-type and overall F_1;



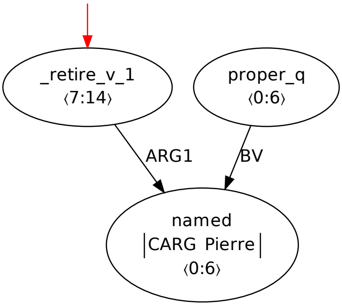
Pierre retired.

Different Types of Semantic Graph 'Atoms'

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

Cross-Framework Evaluation: MRP Graph Similarity

- Break down graphs into types of information: per-type and overall F_1;
- tops



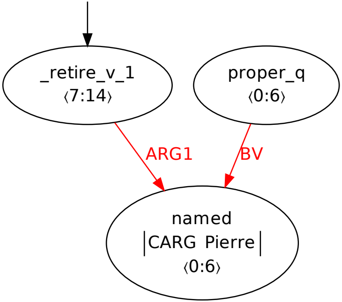
Pierre retired.

Different Types of Semantic Graph 'Atoms'

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

Cross-Framework Evaluation: MRP Graph Similarity

- Break down graphs into types of information: per-type and overall F_1;
- tops and (labeled) edges;

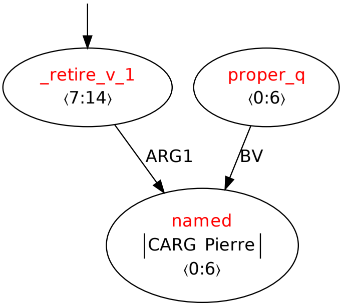


Pierre retired.

Different Types of Semantic Graph 'Atoms'

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

- Break down graphs into types of information: per-type and overall F_1;
- tops and (labeled) edges; labels,



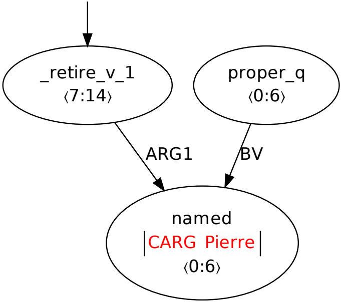
Pierre retired.

Different Types of Semantic Graph 'Atoms'

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

Cross-Framework Evaluation: MRP Graph Similarity

- Break down graphs into types of information: per-type and overall F_1;
- tops and (labeled) edges; labels, properties,



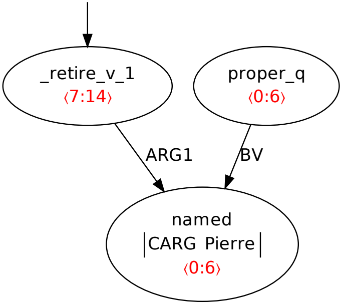
Pierre retired.

Different Types of Semantic Graph 'Atoms'

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

Cross-Framework Evaluation: MRP Graph Similarity

- Break down graphs into types of information: per-type and overall F_1;
- tops and (labeled) edges; labels, properties, anchors,



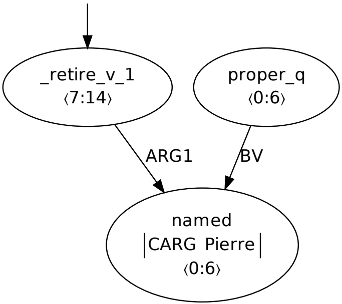
Pierre retired.

Different Types of Semantic Graph 'Atoms'

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

Cross-Framework Evaluation: MRP Graph Similarity

- Break down graphs into types of information: per-type and overall F_1;
- tops and (labeled) edges; labels, properties, anchors, and attributes;



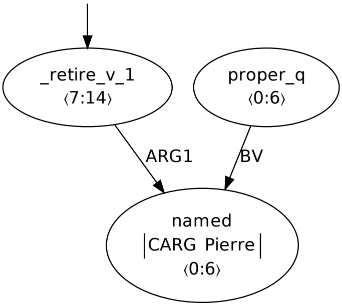
Pierre retired.

Different Types of Semantic Graph 'Atoms'

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

Cross-Framework Evaluation: MRP Graph Similarity

- Break down graphs into types of information: per-type and overall F_1;
- tops and (labeled) edges; labels, properties, anchors, and attributes;
- requires node–node correspondences; search for overall maximum score;
- maximum common edge subgraph isomorphism (MCES) is NP-hard;



Pierre retired.

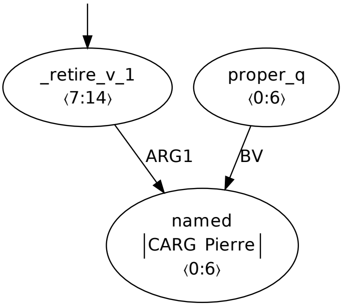
Different Types of Semantic Graph 'Atoms'

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

Cross-Framework Evaluation: MRP Graph Similarity

- Break down graphs into types of information: per-type and overall F_1;
- tops and (labeled) edges; labels, properties, anchors, and attributes;
- requires node–node correspondences; search for overall maximum score;
- maximum common edge subgraph isomorphism (MCES) is NP-hard;

→ smart initialization, scheduling, and pruning yield strong approximation.



Pierre retired.

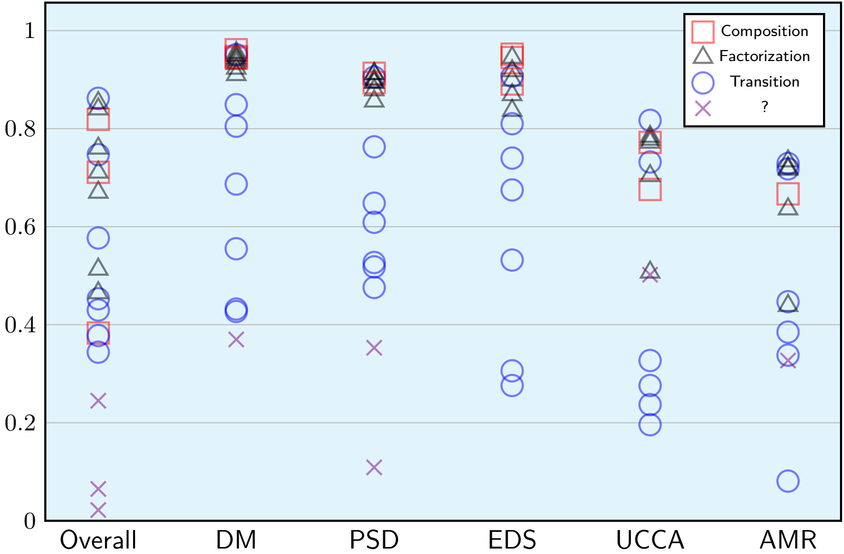
Different Types of Semantic Graph 'Atoms'

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

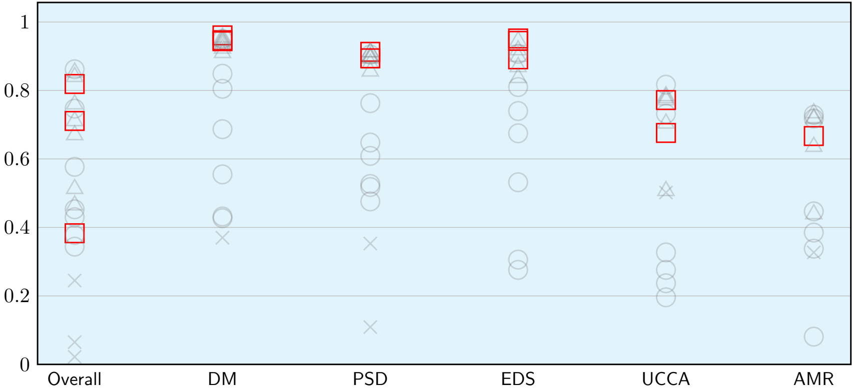
High-Level Overview of Submissions

Teams	DM	PSD	EDS	UCCA	AMR	MTL	Approach
ERG[♯§†]	✓	✗	✓	✗	✗	✗	Composition
TUPA[§†]	✓	✓	✓	✓	✓	✗/✓	Transition
HIT-SCIR	✓	✓	✓	✓	✓	✗	Transition
SJTU–NICT	✓	✓	✓	✓	✓	✗	Factorization
SUDA–Alibaba	✓	✓	✓	✓	✓	(✓)	Factorization
Saarland	✓	✓	✓	✓	✓	✗	Composition
Hitachi	✓	✓	✓	✓	✓	(✓)	Factorization
ÚFAL MRPipe	✓	✓	✓	✓	✓	✗	Transition
ShanghaiTech	✓	✓	✓	✗	✓	✗	Factorization
Amazon	✓	✓	✗	✗	✓	✗	Factorization
JBNU	✓	✓	✗	✓	✗	✗	Factorization
SJTU	✓	✓	✓	✓	✓	✓	Transition
ÚFAL–Oslo	✓	✓	✓	✓	✓	✗	Transition
HKUST	✓	✓	✗	✓	✗	?	
Bocharov	✗	✗	✗	✗	✓	?	
Peking[♯]	✓	✓	✓	✓	✗	✗	Factorization
CUHK[§]	✓	✓	✓	✓	✓	✓	Transition
Anonymous[§]	✗	✓	✗	✗	✗	?	

Score Distributions

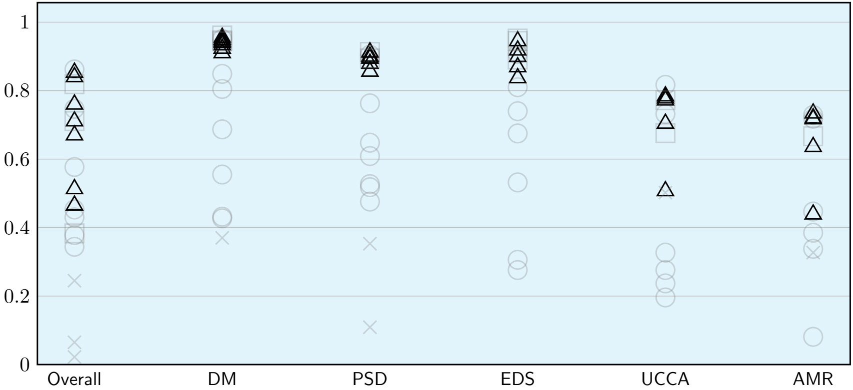


Composition-Based Approaches



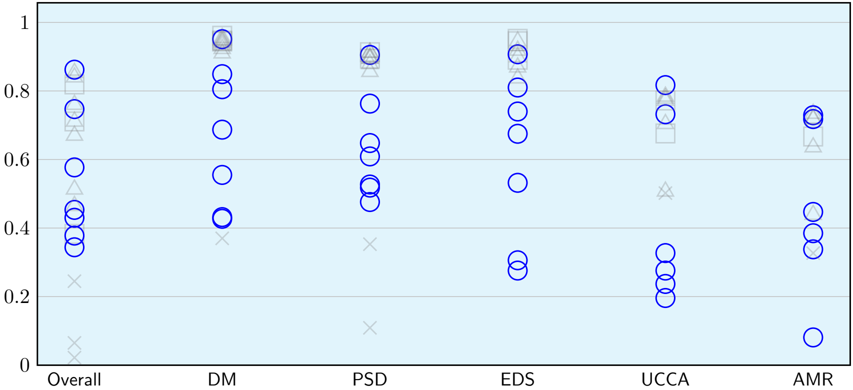
- Explicitly modeling the derivation process.
- A parser evaluates a derivation licensed by a symbolic system.

Factorization-Based Approaches



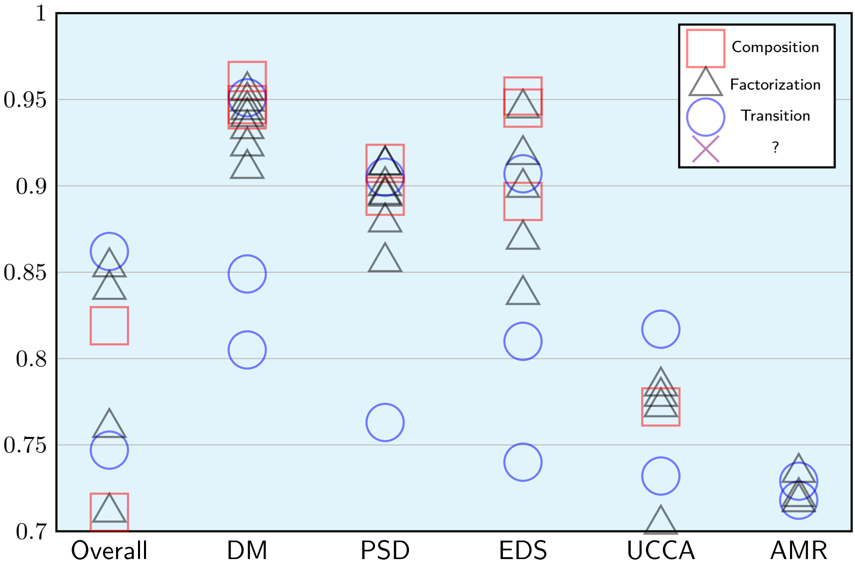
- Inspired by graph-based dependency parsers.
- Explicitly modeling the target structure.
- A parser evaluates factors of a candidate graph.

Transition-Based Approaches

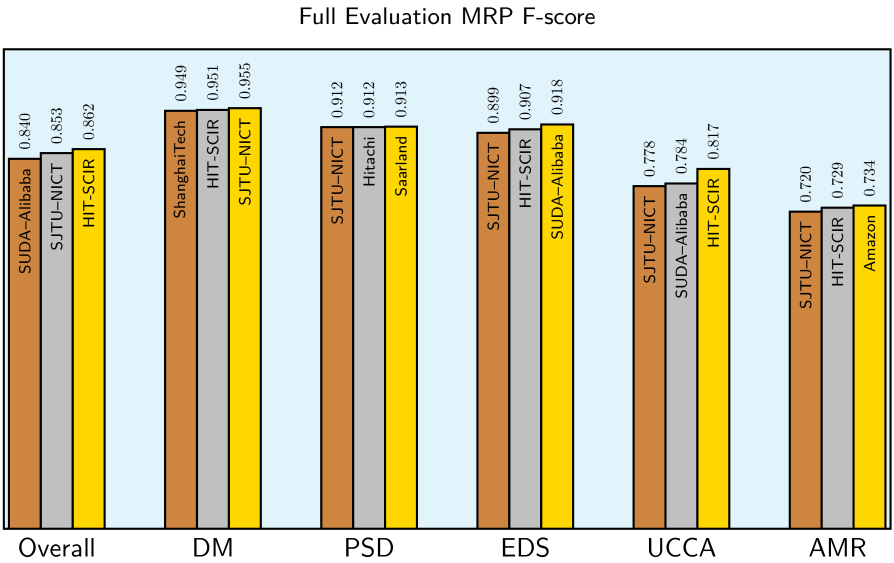


- Inspired by transition-based dependency parsers.
- Incremental (left-to-right, word-by-word).
- Partial parse constrains subsequent actions.
- Greedy/beam search to get a parse.

Score Distributions: Zoom In

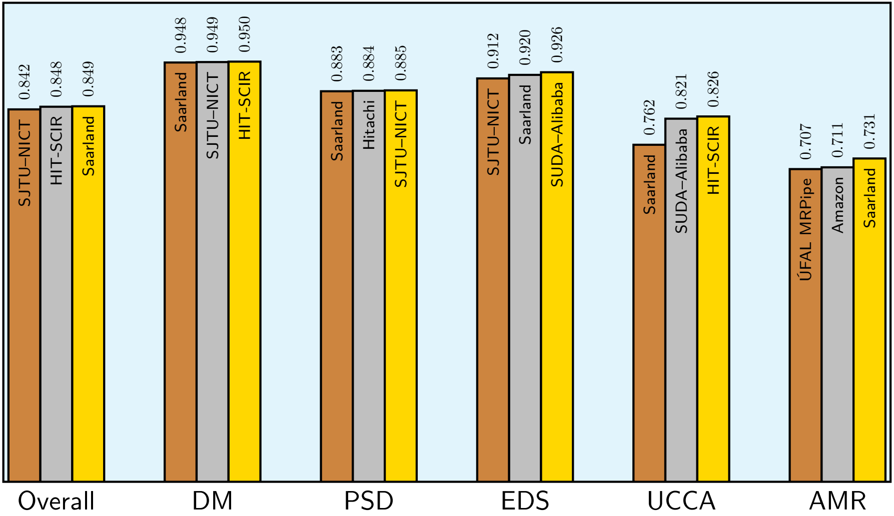


Official Leaderboard: All Evaluation Data



Leaderboard: LPPS Subset of Evaluation Data

The Little Prince Subset MRP F-score

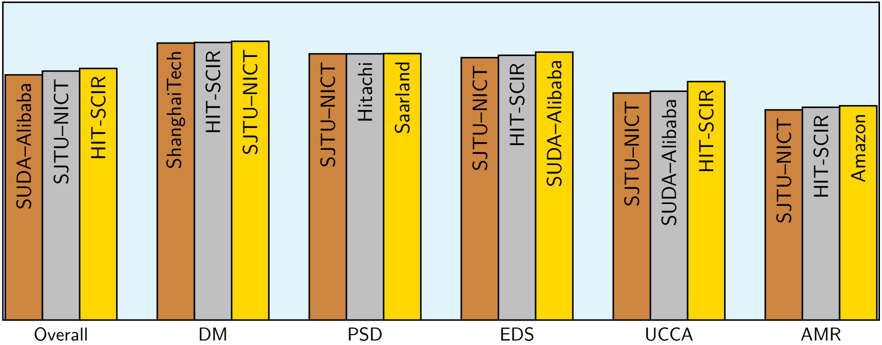


State of the Art

Submissions from established top-performing teams:

- ShanghaiTech (DM, PSD)
- Peking (EDS)
- SUDA–Alibaba (UCCA)
- Saarland (AMR)

Outperformed in most cases!



Limiting Factors in Comparison to State of the Art

- New cross-framework metric: MRP

Limiting Factors in Comparison to State of the Art

- New cross-framework metric: MRP
- Different task definition (DM, PSD: nodes, not just edges)
- Different evaluation set (EDS: not just WSJ)
- Different normalization (AMR: inverted edges)
- Revised and extended annotation (UCCA, AMR)

Limiting Factors in Comparison to State of the Art

- New cross-framework metric: MRP
- Different task definition (DM, PSD: nodes, not just edges)
- Different evaluation set (EDS: not just WSJ)
- Different normalization (AMR: inverted edges)
- Revised and extended annotation (UCCA, AMR)
- No gold tokenization (or tags or lemmas)!

Interim Conclusions & Outlook

Lessons Learned

- Great community interest: 160 subscribers; 38 data licenses (via LDC);
- task complexity is technical barrier to entry: $16 + 2$ teams submitted;

Interim Conclusions & Outlook

Lessons Learned

- Great community interest: 160 subscribers; 38 data licenses (via LDC);
- task complexity is technical barrier to entry: $16 + 2$ teams submitted;

$\rightarrow$ advanced state of the art on four frameworks (but possibly not AMR);

Interim Conclusions & Outlook

Lessons Learned

- Great community interest: 160 subscribers; 38 data licenses (via LDC);
- task complexity is technical barrier to entry: $16 + 2$ teams submitted;

→ advanced state of the art on four frameworks (but possibly not AMR);

→ greatly increased cross-framework uniformity; but limited MTL so far.

Interim Conclusions & Outlook

Lessons Learned

- Great community interest: 160 subscribers; 38 data licenses (via LDC);
- task complexity is technical barrier to entry: $16 + 2$ teams submitted;

→ advanced state of the art on four frameworks (but possibly not AMR);

→ greatly increased cross-framework uniformity; but limited MTL so far.

Outlook: Toward MRP 2020

- Invitation from SIGNLL to re-run (a follow-up variant) at CoNLL 2020;

Interim Conclusions & Outlook

Lessons Learned

- Great community interest: 160 subscribers; 38 data licenses (via LDC);
- task complexity is technical barrier to entry: $16 + 2$ teams submitted;

→ advanced state of the art on four frameworks (but possibly not AMR);

→ greatly increased cross-framework uniformity; but limited MTL so far.

Outlook: Toward MRP 2020

- Invitation from SIGNLL to re-run (a follow-up variant) at CoNLL 2020;

? add Discourse Representation Graphs; maybe a few other languages;

? increased focus on evaluation metrics: score 'larger pieces'; SEMBLEU;

Interim Conclusions & Outlook

Lessons Learned

- Great community interest: 160 subscribers; 38 data licenses (via LDC);
- task complexity is technical barrier to entry: $16 + 2$ teams submitted;

→ advanced state of the art on four frameworks (but possibly not AMR);

→ greatly increased cross-framework uniformity; but limited MTL so far.

Outlook: Toward MRP 2020

- Invitation from SIGNLL to re-run (a follow-up variant) at CoNLL 2020;

? add Discourse Representation Graphs; maybe a few other languages;

? increased focus on evaluation metrics: score 'larger pieces'; SEMBLEU;

→ open discussion with 2019 participants towards the end of this session.

Acknowledgments

Mohit Bansal, Emily M. Bender, Xavier Carreras,

Jayeol Chun, Dotan Dvir, Dan Flickinger,

Julia Hockenmaier, Andrey Kutuzov,

Sebastian Schuster, Milan Straka,

Zdeňka Urešová, and Aline Villavicencio

Linguistic Data Consortium,

Nordic Language Processing Laboratory

References I

Omri Abend & Ari Rappoport. 2013. Universal Conceptual Cognitive Annotation (UCCA). In Proceedings of the 51th Meeting of the Association for Computational Linguistics, pages 228–238, Sofia, Bulgaria.

Collin F. Baker, Charles J. Fillmore, & John B. Lowe. 1998. The Berkeley FrameNet project. In Proceedings of the 17th International Conference on Computational Linguistics and the 36th Meeting of the Association for Computational Linguistics, pages 86–90, Stroudsburg, PA, USA.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, & 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.

Valerio Basile, Johan Bos, Kilian Evang, & 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.

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, & 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.

Hardy & Andreas Vlachos. 2018. Guided neural language generation for abstractive summarization using abstract meaning representation. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium.

Angelina Ivanova, Stephan Oepen, Lilja Øvrelid, & 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.

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

Stephan Oepen & 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.

Martha Palmer, Dan Gildea, & Paul Kingsbury. 2005. The Proposition Bank. A corpus annotated with semantic roles. Computational Linguistics, 31(1):71–106.

Sebastian Schuster & 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.

References IV

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

Aaron Steven White, Drew Reisinger, Keisuke Sakaguchi, Tim Vieira, Sheng Zhang, Rachel Rudinger, Kyle Rawlins, & Benjamin Van Durme. 2016. Universal Decompositional Semantics on Universal Dependencies. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 1713–1723, Austin, TX, USA.

TUPA at MRP 2019

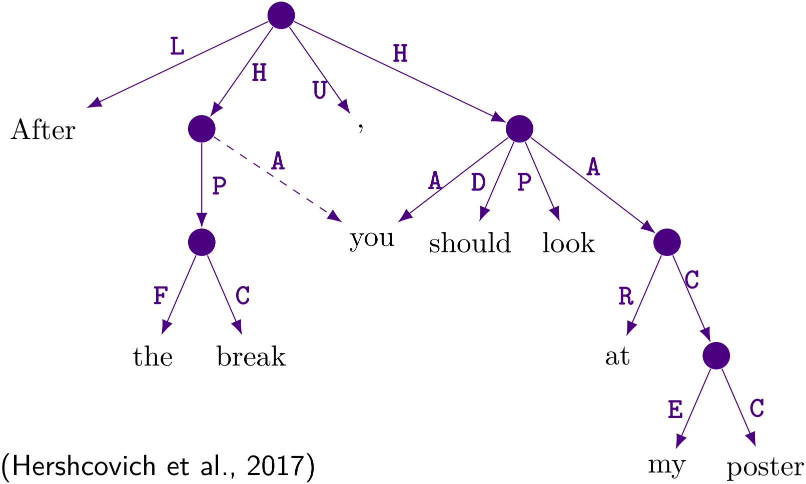
A Multi-Task Baseline System

Daniel Hershcovich[1], Ofir Arviv[2]

[1]University of Copenhagen
[2]Hebrew University of Jerusalem

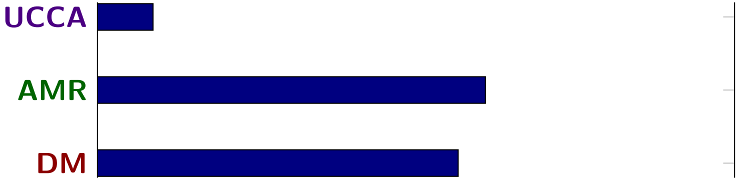
CoNLL Shared Task
3 November 2019

Transition-Based UCCA Parser

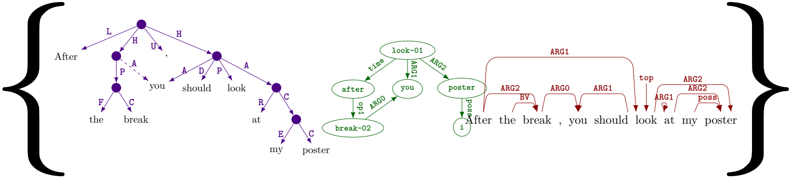


(Hershcovich et al., 2017)

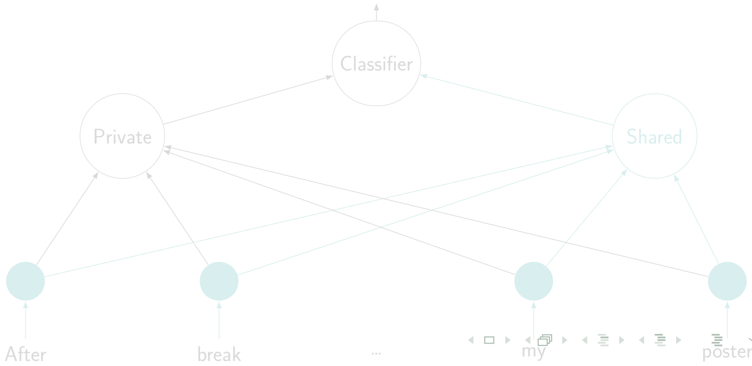
Scarcity of Training Data



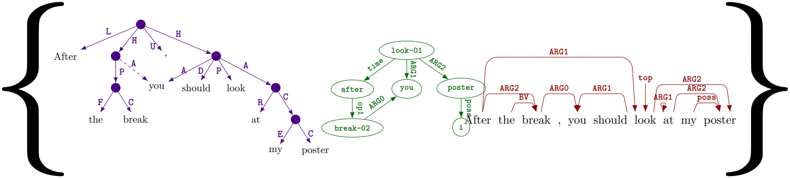
Multi-Task Parser



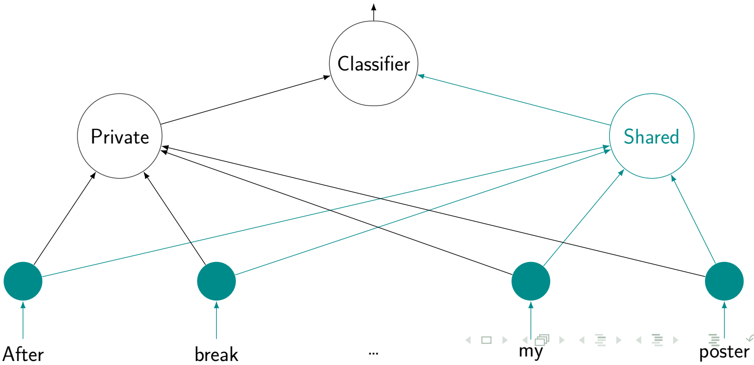
Multi-task improves UCCA parsing (Hershcovich et al., 2018).



Multi-Task Parser

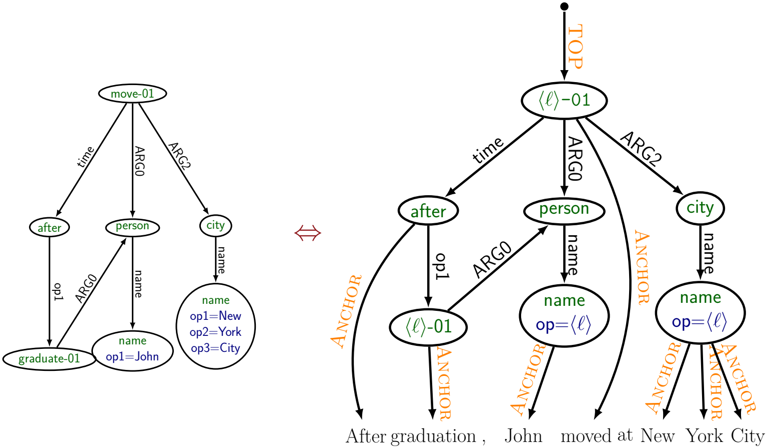


Multi-task improves UCCA parsing (Hershcovich et al., 2018).



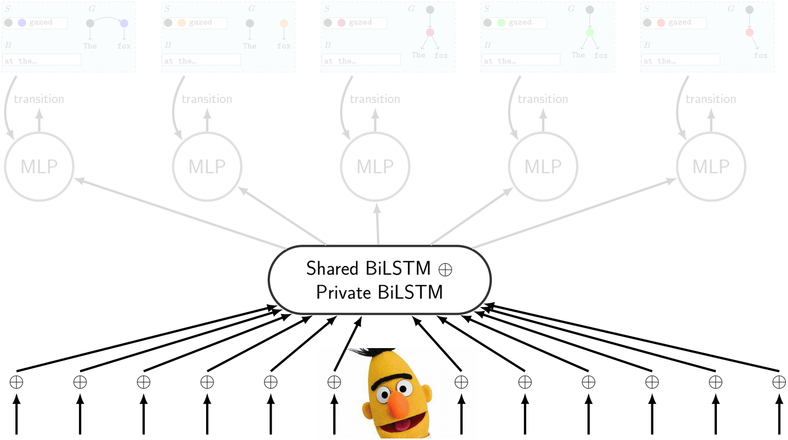
MRP

Intermediate graph representation,
extended transition system.



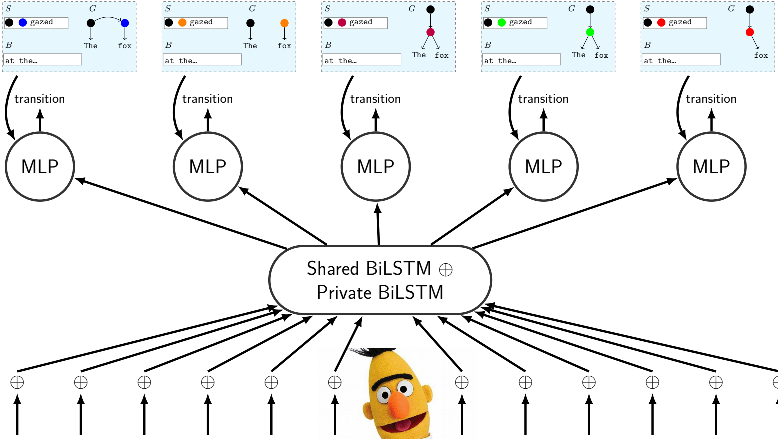
Transition Classifier

BiLSTM + BERT (Devlin et al., 2019).



Transition Classifier

BiLSTM + BERT (Devlin et al., 2019).



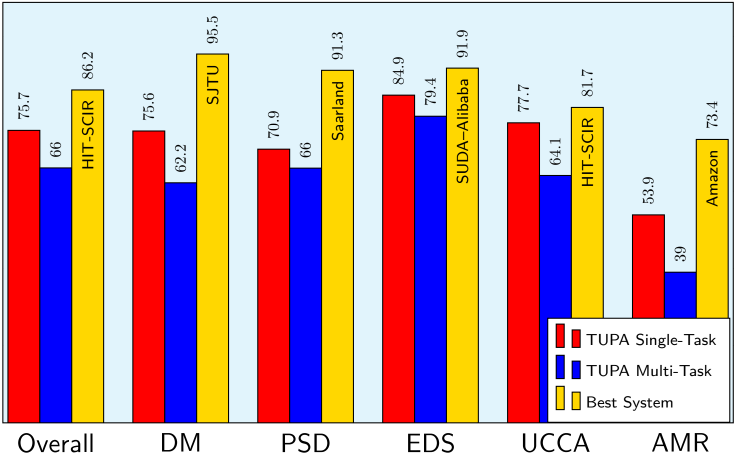
Results

Baseline: single-task + multi-task.

Full Evaluation MRP F-score (%). TUPA scores are post-evaluation.

Results

Baseline: single-task + multi-task.



Full Evaluation MRP F-score (%). TUPA scores are post-evaluation.

TUPA at MRP 2019

A Multi-Task Baseline System

Daniel Hershcovich[1], Ofir Arviv[2]

[1]University of Copenhagen
[2]Hebrew University of Jerusalem

CoNLL Shared Task
3 November 2019

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of NAACL*, pages 4171–4186.

Daniel Hershcovich, Omri Abend, and Ari Rappoport. 2017. A transition-based directed acyclic graph parser for UCCA. In *Proc. of ACL*, pages 1127–1138.

Daniel Hershcovich, Omri Abend, and Ari Rappoport. 2018. Multitask parsing across semantic representations. In *Proc. of ACL*, pages 373–385.

The ERG at MRP 2019

Radically Compositional Semantic Dependencies

http://erg.delph-in.net

Stephan Oepen♣ and Dan Flickinger♠

♣ University of Oslo, Department of Informatics

♠ Stanford University, Center for the Study of Language and Information

oe@ifi.uio.no , danf@stanford.edu

Background: English Resource Semantics (ERS)

LinGO English Resource Grammar (Flickinger, 2000; Flickinger et al., 2017)

- Hand-designed computational grammar for English in HPSG framework;
- declarative, unification-based: parsing and realization; multiple engines;
- 25^{+} person years; coverage of 85–95 % of running text across domains;

Background: English Resource Semantics (ERS)

LinGO English Resource Grammar (Flickinger, 2000; Flickinger et al., 2017)

- Hand-designed computational grammar for English in HPSG framework;
- declarative, unification-based: parsing and realization; multiple engines;
- 25^{+} person years; coverage of 85–95 % of running text across domains;
- underspecified meaning representation in MRS (Copestake et al., 2005).

Background: English Resource Semantics (ERS)

LinGO English Resource Grammar (Flickinger, 2000; Flickinger et al., 2017)

- Hand-designed computational grammar for English in HPSG framework;
- declarative, unification-based: parsing and realization; multiple engines;
- 25^{+} person years; coverage of 85–95 % of running text across domains;
- underspecified meaning representation in MRS (Copestake et al., 2005).

LinGO Redwoods Treebank (Carter, 1997; Oepen et al., 2004)

- Grammar-based annotation: select ‘correct’ reading from parse forest;

Background: English Resource Semantics (ERS)

LinGO English Resource Grammar (Flickinger, 2000; Flickinger et al., 2017)

- Hand-designed computational grammar for English in HPSG framework;
- declarative, unification-based: parsing and realization; multiple engines;
- 25^{+} person years; coverage of 85–95 % of running text across domains;
- underspecified meaning representation in MRS (Copestake et al., 2005).

LinGO Redwoods Treebank (Carter, 1997; Oepen et al., 2004)

- Grammar-based annotation: select ‘correct’ reading from parse forest;
- version 1214: some 85,000 annotated sentences, six^{+} different domains;

Background: English Resource Semantics (ERS)

LinGO English Resource Grammar (Flickinger, 2000; Flickinger et al., 2017)

- Hand-designed computational grammar for English in HPSG framework;
- declarative, unification-based: parsing and realization; multiple engines;
- 25^{+} person years; coverage of 85–95 % of running text across domains;
- underspecified meaning representation in MRS (Copestake et al., 2005).

LinGO Redwoods Treebank (Carter, 1997; Oepen et al., 2004)

- Grammar-based annotation: select 'correct' reading from parse forest;
- version 1214: some 85,000 annotated sentences, six^{+} different domains;
- Bender et al. (2015) report inter-annotator agreement of 0.94 EDM_{na};

Background: English Resource Semantics (ERS)

LinGO English Resource Grammar (Flickinger, 2000; Flickinger et al., 2017)

- Hand-designed computational grammar for English in HPSG framework;
- declarative, unification-based: parsing and realization; multiple engines;
- 25^{+} person years; coverage of 85–95 % of running text across domains;
- underspecified meaning representation in MRS (Copestake et al., 2005).

LinGO Redwoods Treebank (Carter, 1997; Oepen et al., 2004)

- Grammar-based annotation: select 'correct' reading from parse forest;
- version 1214: some 85,000 annotated sentences, six^{+} different domains;
- Bender et al. (2015) report inter-annotator agreement of 0.94 EDM_{na};
- EDS: graph-based simplification of ERS; DM: its bi-lexical 'reduction';

Background: English Resource Semantics (ERS)

LinGO English Resource Grammar (Flickinger, 2000; Flickinger et al., 2017)

- Hand-designed computational grammar for English in HPSG framework;
- declarative, unification-based: parsing and realization; multiple engines;
- 25^+ person years; coverage of 85–95 % of running text across domains;
- underspecified meaning representation in MRS (Copestake et al., 2005).

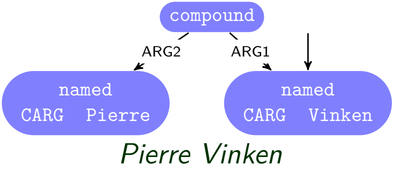
LinGO Redwoods Treebank (Carter, 1997; Oepen et al., 2004)

- Grammar-based annotation: select 'correct' reading from parse forest;
- version 1214: some 85,000 annotated sentences, six^+ different domains;
- Bender et al. (2015) report inter-annotator agreement of 0.94 EDM_{na};
- EDS: graph-based simplification of ERS; DM: its bi-lexical 'reduction';

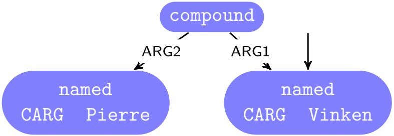
PET Unification-Based Parser (Callmeier, 2002)

- Highly optimized chart parser; (exact) n-best MaxEnt parse selection.

EDSs are ‘Radically Compositional’



EDSs are ‘Radically Compositional’



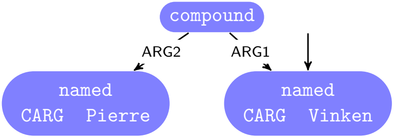
Pierre Vinken

Named Entities

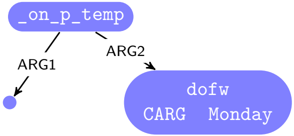
- Underspecified structure in names;
- few, lexically determined sub-types.

Michelle and Barack Obama

EDSs are 'Radically Compositional'



Pierre Vinken



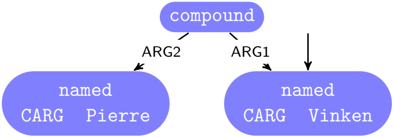
on Monday

Named Entities

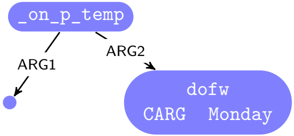
- Underspecified structure in names;
- few, lexically determined sub-types.

Michelle and Barack Obama

EDSs are 'Radically Compositional'



Pierre Vinken



on Monday

Named Entities

- Underspecified structure in names;
- few, lexically determined sub-types.

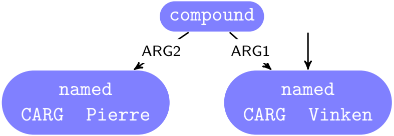
Michelle and Barack Obama

Prepositions (and Similar)

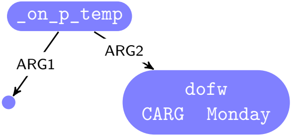
- Predicates: distinct two-place relation;
- specialized sub-senses as appropriate.

before and during the meeting

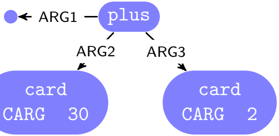
EDSs are ‘Radically Compositional’



Pierre Vinken



on Monday



thirty-two

Named Entities

- Underspecified structure in names;
- few, lexically determined sub-types.

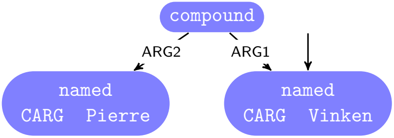
Michelle and Barack Obama

Prepositions (and Similar)

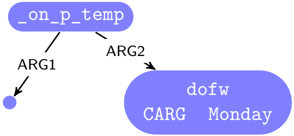
- Predicates: distinct two-place relation;
- specialized sub-senses as appropriate.

before and during the meeting

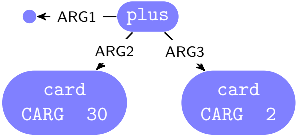
EDSs are 'Radically Compositional'



Pierre Vinken



on Monday



thirty-two

Named Entities

- Underspecified structure in names;
- few, lexically determined sub-types.

Michelle and Barack Obama

Prepositions (and Similar)

- Predicates: distinct two-place relation;
- specialized sub-senses as appropriate.

before and during the meeting

Literal Numbers

- syntax yields arithmetic expressions;
- trivial 'downstream' normalization.

ten to twenty thousand

Comparison to Top-Performing Data-Driven Parsers

		Tops			Labels			Properties			Anchors			Edges		
		P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
DM	ERG	.92	.92	.918	.99	.99	**.987**	.96	.96	**.956**	.99	.99	**.994**	.91	.91	.912
	SJTU–NICT	.93	.93	**.933**	.95	.95	.949	.96	.95	.955	.99	.99	.993	.93	.92	.924
	HIT-SCIR	.93	.93	.926	.93	.93	.930	.95	.95	.953	.99	.99	.993	.93	.92	**.925**
	SUDA–Alibaba	.91	.91	.911	.90	.91	.903	.91	.92	.915	.97	.99	.982	.89	.91	.898
	Peking	.93	.93	.927	.92	.91	.915	.95	.94	.945	.99	.99	.991	.92	.92	.924
EDS	ERG	.90	.90	**.902**	.97	.96	**.965**	.96	.96	**.960**	.96	.96	**.963**	.93	.93	.929
	SUDA–Alibaba	.90	.90	.899	.91	.91	.912	.89	.91	.897	.95	.95	.949	.90	.90	.897
	HIT-SCIR	.88	.82	.852	.90	.89	.894	.89	.91	.895	.95	.94	.943	.89	.88	.888
	SJTU–NICT	.91	.85	.877	.93	.86	.894	.79	.76	.775	.97	.90	.934	.95	.82	.878
	Peking	.83	.83	.829	.95	.94	.946	.91	.96	.936	.96	.96	.961	.94	.93	**.933**

Comparison to Top-Performing Data-Driven Parsers

		Tops			Labels			Properties			Anchors			Edges		
		P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
DM	ERG	.92	.92	.918	.99	.99	**.987**	.96	.96	**.956**	.99	.99	**.994**	.91	.91	.912
	SJTU–NICT	.93	.93	**.933**	.95	.95	.949	.96	.95	.955	.99	.99	.993	.93	.92	.924
	HIT-SCIR	.93	.93	.926	.93	.93	.930	.95	.95	.953	.99	.99	.993	.93	.92	**.925**
	SUDA–Alibaba	.91	.91	.911	.90	.91	.903	.91	.92	.915	.97	.99	.982	.89	.91	.898
	Peking	.93	.93	.927	.92	.91	.915	.95	.94	.945	.99	.99	.991	.92	.92	.924
EDS	ERG	.90	.90	**.902**	.97	.96	**.965**	.96	.96	**.960**	.96	.96	**.963**	.93	.93	.929
	SUDA–Alibaba	.90	.90	.899	.91	.91	.912	.89	.91	.897	.95	.95	.949	.90	.90	.897
	HIT-SCIR	.88	.82	.852	.90	.89	.894	.89	.91	.895	.95	.94	.943	.89	.88	.888
	SJTU–NICT	.91	.85	.877	.93	.86	.894	.79	.76	.775	.97	.90	.934	.95	.82	.878
	Peking	.83	.83	.829	.95	.94	.946	.91	.96	.936	.96	.96	.961	.94	.93	**.933**

Comparison to Top-Performing Data-Driven Parsers

		Tops			Labels			Properties			Anchors			Edges		
		P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
DM	ERG	.92	.92	.918	.99	.99	**.987**	.96	.96	**.956**	.99	.99	**.994**	.91	.91	.912
DM	SJTU–NICT	.93	.93	**.933**	.95	.95	.949	.96	.95	.955	.99	.99	.993	.93	.92	.924
DM	HIT-SCIR	.93	.93	.926	.93	.93	.930	.95	.95	.953	.99	.99	.993	.93	.92	**.925**
DM	SUDA–Alibaba	.91	.91	.911	.90	.91	.903	.91	.92	.915	.97	.99	.982	.89	.91	.898
DM	Peking	.93	.93	.927	.92	.91	.915	.95	.94	.945	.99	.99	.991	.92	.92	.924
EDS	ERG	.90	.90	**.902**	.97	.96	**.965**	.96	.96	**.960**	.96	.96	**.963**	.93	.93	.929
EDS	SUDA–Alibaba	.90	.90	.899	.91	.91	.912	.89	.91	.897	.95	.95	.949	.90	.90	.897
EDS	HIT-SCIR	.88	.82	.852	.90	.89	.894	.89	.91	.895	.95	.94	.943	.89	.88	.888
EDS	SJTU–NICT	.91	.85	.877	.93	.86	.894	.79	.76	.775	.97	.90	.934	.95	.82	.878
EDS	Peking	.83	.83	.829	.95	.94	.946	.91	.96	.936	.96	.96	.961	.94	.93	**.933**

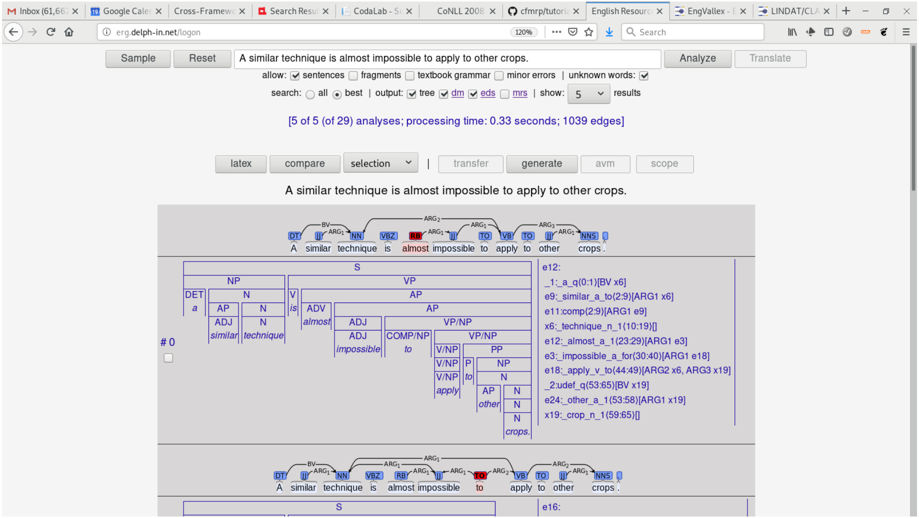
Comparison to Top-Performing Data-Driven Parsers

		Tops			Labels			Properties			Anchors			Edges		
		P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
DM	ERG	.92	.92	.918	.99	.99	**.987**	.96	.96	**.956**	.99	.99	**.994**	.91	.91	.912
	SJTU–NICT	.93	.93	**.933**	.95	.95	.949	.96	.95	.955	.99	.99	.993	.93	.92	.924
	HIT-SCIR	.93	.93	.926	.93	.93	.930	.95	.95	.953	.99	.99	.993	.93	.92	**.925**
	SUDA–Alibaba	.91	.91	.911	.90	.91	.903	.91	.92	.915	.97	.99	.982	.89	.91	.898
	Peking	.93	.93	.927	.92	.91	.915	.95	.94	.945	.99	.99	.991	.92	.92	.924
EDS	ERG	.90	.90	**.902**	.97	.96	**.965**	.96	.96	**.960**	.96	.96	**.963**	.93	.93	.929
	SUDA–Alibaba	.90	.90	.899	.91	.91	.912	.89	.91	.897	.95	.95	.949	.90	.90	.897
	HIT-SCIR	.88	.82	.852	.90	.89	.894	.89	.91	.895	.95	.94	.943	.89	.88	.888
	SJTU–NICT	.91	.85	.877	.93	.86	.894	.79	.76	.775	.97	.90	.934	.95	.82	.878
	Peking	.83	.83	.829	.95	.94	.946	.91	.96	.936	.96	.96	.961	.94	.93	**.933**

Comparison to Top-Performing Data-Driven Parsers

		Tops			Labels			Properties			Anchors			Edges		
		P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
DM	ERG	.92	.92	.918	.99	.99	**.987**	.96	.96	**.956**	.99	.99	**.994**	.91	.91	.912
	SJTU–NICT	.93	.93	**.933**	.95	.95	.949	.96	.95	.955	.99	.99	.993	.93	.92	.924
	HIT-SCIR	.93	.93	.926	.93	.93	.930	.95	.95	.953	.99	.99	.993	.93	.92	**.925**
	SUDA–Alibaba	.91	.91	.911	.90	.91	.903	.91	.92	.915	.97	.99	.982	.89	.91	.898
	Peking	.93	.93	.927	.92	.91	.915	.95	.94	.945	.99	.99	.991	.92	.92	.924
EDS	ERG	.90	.90	**.902**	.97	.96	**.965**	.96	.96	**.960**	.96	.96	**.963**	.93	.93	.929
	SUDA–Alibaba	.90	.90	.899	.91	.91	.912	.89	.91	.897	.95	.95	.949	.90	.90	.897
	HIT-SCIR	.88	.82	.852	.90	.89	.894	.89	.91	.895	.95	.94	.943	.89	.88	.888
	SJTU–NICT	.91	.85	.877	.93	.86	.894	.79	.76	.775	.97	.90	.934	.95	.82	.878
	Peking	.83	.83	.829	.95	.94	.946	.91	.96	.936	.96	.96	.961	.94	.93	**.933**

Background: English Resource Semantics On-Line



http://erg.delph-in.net/

References I

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.

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.

David Carter. 1997. The TreeBanker. A tool for supervised training of parsed corpora. In Proceedings of the Workshop on Computational Environments for Grammar Development and Linguistic Engineering, pages 9–15, Madrid, Spain.

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.

References II

Dan Flickinger. 2000. On building a more efficient grammar by exploiting types. Natural Language Engineering, 6 (1):15–28.

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.

Stephan Oepen, Daniel Flickinger, Kristina Toutanova, and Christopher D. Manning. 2004. LinGO Redwoods. A rich and dynamic treebank for HPSG. Research on Language and Computation, 2(4):575–596.

SJTU-NICT at MRP 2019: Multi-Task Learning for End-to-End Uniform Semantic Graph Parsing

Zuchao Li[1,2,3], **Hai Zhao**[1,2,3,*], **Zhuosheng Zhang**[1,2,3],
Rui Wang[4,*], **Masao Utiyama**[4], and **Eiichiro Sumita**[4]
[1]Department of Computer Science and Engineering, Shanghai Jiao Tong University (SJTU)
[2]Key Laboratory of Shanghai Education Commission for Intelligent Interaction and Cognitive Engineering, Shanghai Jiao Tong University, Shanghai, China
[3]MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University, China
[4]National Institute of Information and Communications Technology (NICT), Kyoto, Japan

Abstract

This paper describes our SJTU-NICT's system for participating in the shared task on Cross-Framework Meaning Representation Parsing (MRP) at the 2019 Conference for Computational Language Learning (CoNLL). Our system uses a graph-based approach to model a variety of semantic graph parsing tasks. Our main contributions in the submitted system are summarized as follows: 1. Our model is fully end-to-end and is capable of being trained only on the given training set which does not rely on any other extra training source including the companion data provided by the

However, due to the variety of semantic graph flavors, the framework-specific "balkanization" of semantic parsing is worth noting. The 2019 Conference on Computational Language Learning (CoNLL) hosts a shared task on Cross-Framework Meaning Representation Parsing (MRP 2019) (Oepen et al., 2019). From the perspective of the formal representation of semantic graphs, MRP 2019 uses the directed graphs to unify the five different semantic representation frameworks: DELPH-IN MRS Bi-Lexical Dependencies (DM), Prague Semantic Dependencies (PSD), Elementary Dependency

ShanghaiTech at MRP 2019: Sequence-to-Graph Transduction with Second-Order Edge Inference for Cross-Framework Meaning Representation Parsing

Xinyu Wang, Yixian Liu, Zixia Jia, Chengyue Jiang, Kewei Tu
School of Information Science and Technology,
ShanghaiTech University, Shanghai, China
{wangxy1,liuyx,jiazx,jiangchy,tukw}@shanghaitech.edu.cn

Abstract

This paper presents the system used in our submission to the *CoNLL 2019 shared task: Cross-Framework Meaning Representation Parsing*. Our system is a graph-based parser which combines an extended pointer-generator network that generates nodes and a second-order mean field variational inference module that predicts edges. Our system achieved 1st and 2nd place for the DM and PSD frameworks respectively on the in-framework ranks and achieved 3rd place for the DM framework on the cross-framework ranks.

tween these frameworks is their level of abstraction from the sentence. SDP is a bi-lexical dependency graph, where graph nodes correspond to tokens in the sentence. EDS and UCCA are general forms of anchored semantic graphs, in which the nodes are anchored to arbitrary spans of the sentence and the spans can have overlaps. AMR is an unanchored graph, which does not consider the correspondence between nodes and the sentence tokens. The shared task also provides a cross-framework metric which evaluates the similarity of graph components in all frameworks.

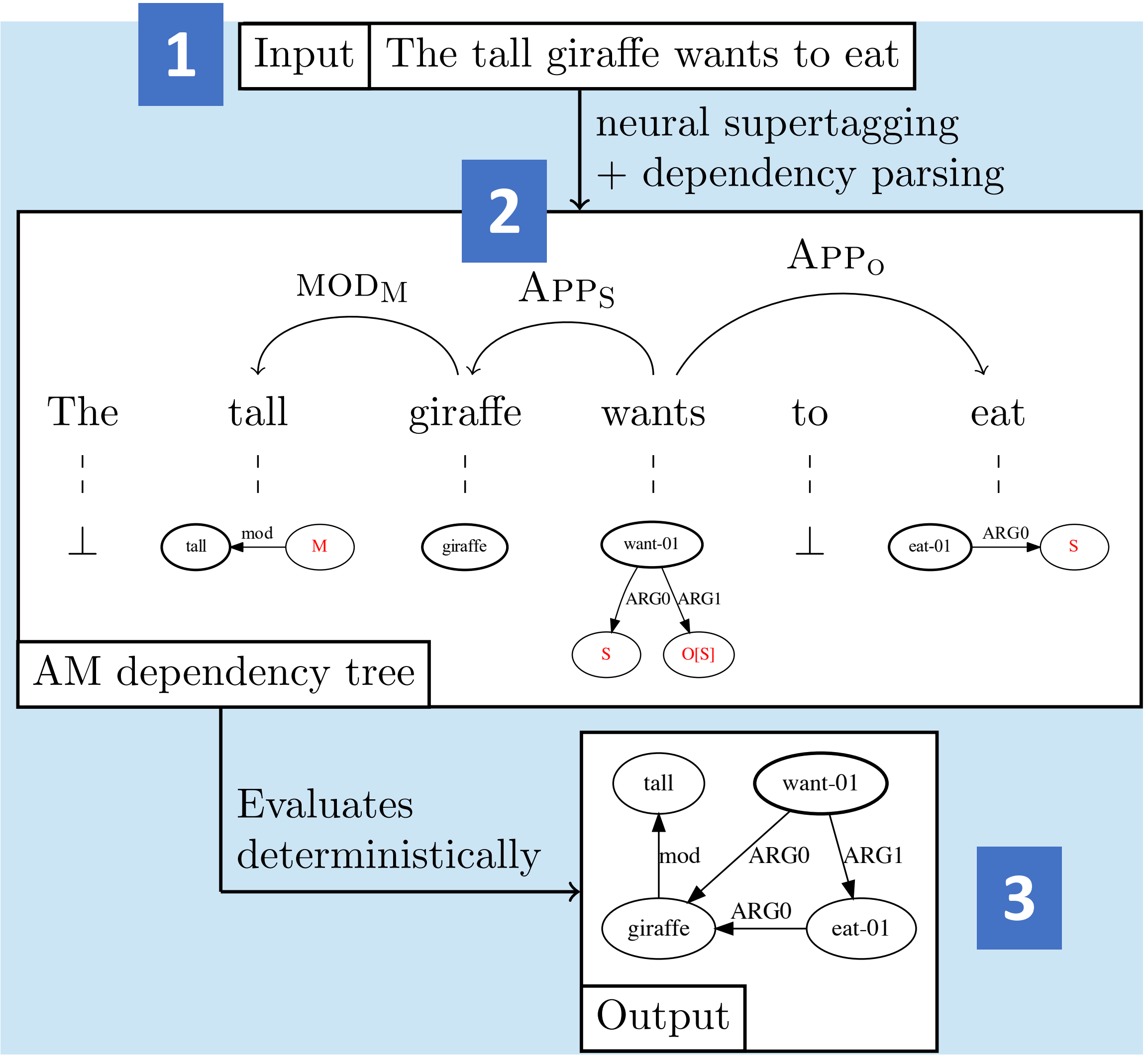
Compositional Parsing Across All Graphbanks

Saarland at MRP 2019

L. Donatelli, M. Fowlie, J. Groschwitz, A. Koller, M. Lindemann, M. Mina, P. Weißenhorn

- Compositional neural parser with competitive results across all MRP shared task graphbanks (only compositional parser to do so!)
  - 4th place overall
  - 1st on PSD
  - 1st *The Little Prince* subset
- Parser previously held SOTA on MRP graphbanks apart from UCCA at ACL 2019

Apply-Modify (AM) Algebra and graph decomposition



1 Sentence

2 AM Dependency tree

3 Graph

- Linguistically-motivated compositional structure
- Diverse meaning representations mapped to similar AM trees

HIT-SCIR at MRP 2019: A Unified Pipeline for Meaning Representation Parsing via Efficient Training and Effective Encoding

Wanxiang Che, Longxu Dou, Yang Xu, Yuxuan Wang, Yijia Liu, Ting Liu

Research Center for Social Computing and Information Retrieval

Harbin Institute of Technology

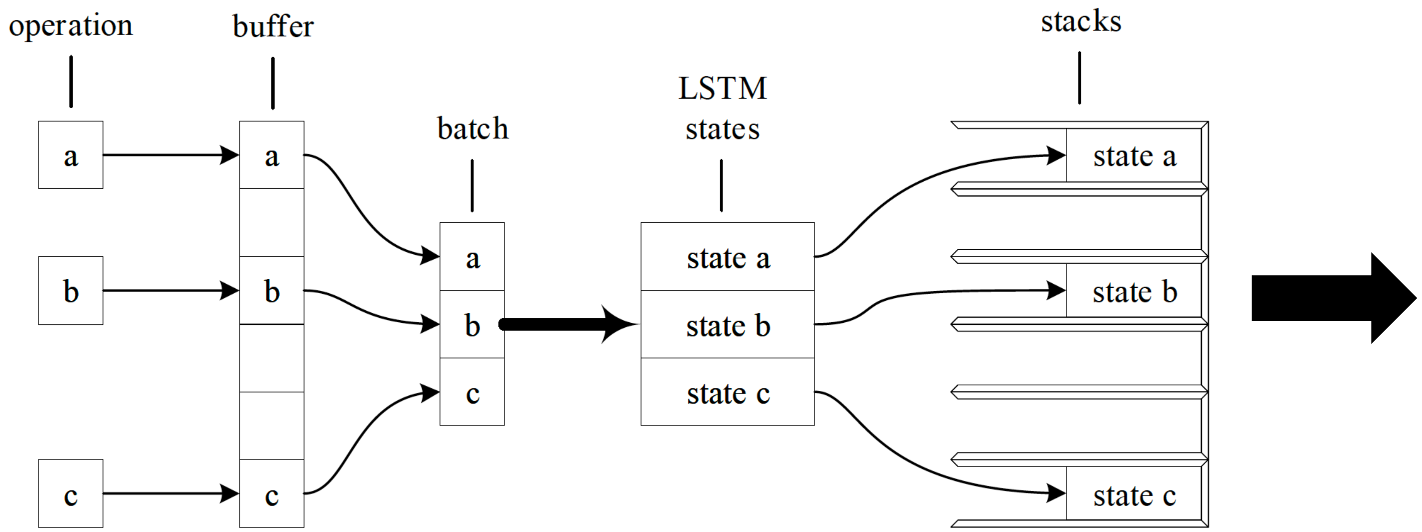
Overview of Our Techniques

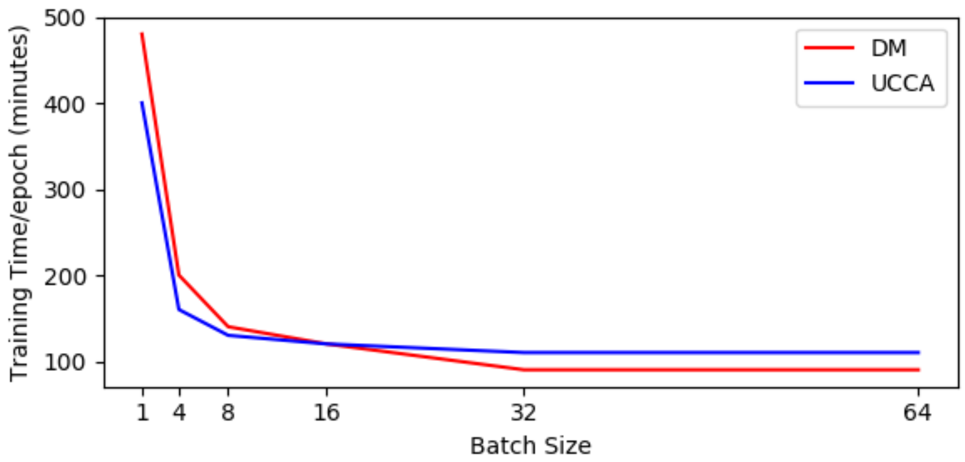
- **Rank 1st according to ALL-F1**
- **Baseline model: Transition-based Parser with Stack LSTM (Dyer et al., 2015)**
- **Our Extensions:**
  - **Efficient Training of Stack LSTM via parallel training**
  - **Effective Encoding via adopting BERT**

System	DM	PSD	EDS	UCCA	AMR	ALL-F1
HIT-SCIR	95.08	90.55	90.75	**81.67**	72.94	**86.2**
SJTU-NICT	**95.50**	91.19	89.90	77.80	71.97	85.3
Suda-Alibaba	92.26	85.56	**91.85**	78.43	71.72	84.0
Saarland	94.69	**91.28**	89.10	67.55	66.72	81.9
Hitachi	91.02	91.21	83.74	70.36	43.86	76.0
Amazon	93.26	89.98	-	-	**73.38**	-

Parallel Training Stack-LSTM

- **Aligning the similar operations in Stack-LSTM within a batch**
- **Computing them simultaneously**





- Conduct experiments with GloVe
- 5.3x on DM
- 2.7x on UCCA

BERT is Amazing!

- **We fine-tune the BERT**
- **Layer-wise scalar weighed BERT is adopted**

Feature	DM	PSD	EDS	UCCA	AMR	Avg
GloVe	87.1	74.1	88.2	87.5	65.3	80.4
BERT(base)	94.3	83.6	91.5	92.8	71.4	86.7

- **Metric: ALL-F1 based on mtool**
- **Dataset: Splited training data on 8:1:1 proportion**

Structure vs Representation

- **Transition-based Parser achieves comparable results with Graph-based Parser**
- **Kulmizev et al. (2019) found similar conclusion in PTB**

Model	Feature	DM		PAS		PSD	
		id	ood	id	ood	id	ood
Wang et al	word2vec	89.3	83.2	91.4	87.2	76.1	73.2
Dozat et al	Glove+char	92.7	87.8	94.0	90.6	80.5	78.6
Transition	GloVe+char	86.1	79.2	89.8	85.2	72.8	68.5
Graph	GloVe+char	91.6	86.1	93.1	89.6	77.4	73.0
Transition	BERT	92.9	89.2	94.4	92.4	**81.6**	**81.0**
Graph	BERT	**94.1**	**90.8**	**94.8**	**92.9**	80.7	79.5

Wang et al: <A Neural Transition-Based Approach for Semantic Dependency Graph Parsing>
Dozat et al: <Simpler but More Accurate Semantic Dependency Parsing>
Kulmizev et al: <Deep Contextualized Word Embeddings in Transition-Based and Graph-Based Dependency Parsing – A Tale of Two Parsers Revisited>

Model Ensemble

- **In follow up experiment, we obtain further improvement on lpps dataset**
- **Ensemble model consists of 5 single model**

Systems	DM	PSD	EDS	UCCA	AMR	Avg
Single	93.98	87.41	89.83	82.61	69.03	84.57
Ensemble	94.00	87.79	89.57	83.41	71.35	85.16

Conclusion

- **Our Contribution:**
  - **Efficient Training of Stack LSTM via parallel training**
  - **Effective Encoding through adopting BERT**

- **The performance gap between Graph and Transition on SDP is almost eliminated under BERT**

- **Our code: https://github.com/HIT-SCIR/HIT-SCIR-CoNLL2019**

No Presentation

SJTU at MRP 2019: A Transition-Based Multi-Task Parser for Cross-Framework Meaning Representation Parsing

Hongxiao Bai[1,2,3], **Hai Zhao**[1,2,3,*]
[1]Department of Computer Science and Engineering, Shanghai Jiao Tong University
[2]Key Laboratory of Shanghai Education Commission for Intelligent Interaction and Cognitive Engineering, Shanghai Jiao Tong University, Shanghai, China
[3]MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University
baippa@sjtu.edu.cn, zhaohai@cs.sjtu.edu.cn

Abstract

This paper describes the system of our team *SJTU* for our participation in the CoNLL 2019 Shared Task: Cross-Framework Meaning Representation Parsing. The goal of the task is to advance data-driven parsing into graph-structured representations of sentence meaning. This task includes five meaning representation frameworks: DM, PSD, EDS, UCCA, and AMR. These frameworks have different properties and structures. To tackle all the frameworks in one model, it is needed to find out the commonality of them. In our work, we define a set of the transition actions to once-

shallow syntax and in particular for representations of the semantic structure. Many works have shown that these meaning representations are beneficial to other tasks such as machine translation and abstractive summarization. However, there are several types of meaning representations with different definitions, structures, and abstractions, which hinder the applications.

The CoNLL 2019 Shared Task (Oepen et al., 2019) combines formally and linguistically different meaning representation in graph form on a uniform training and evaluation setup for the first time. This task includes five MRP frame-

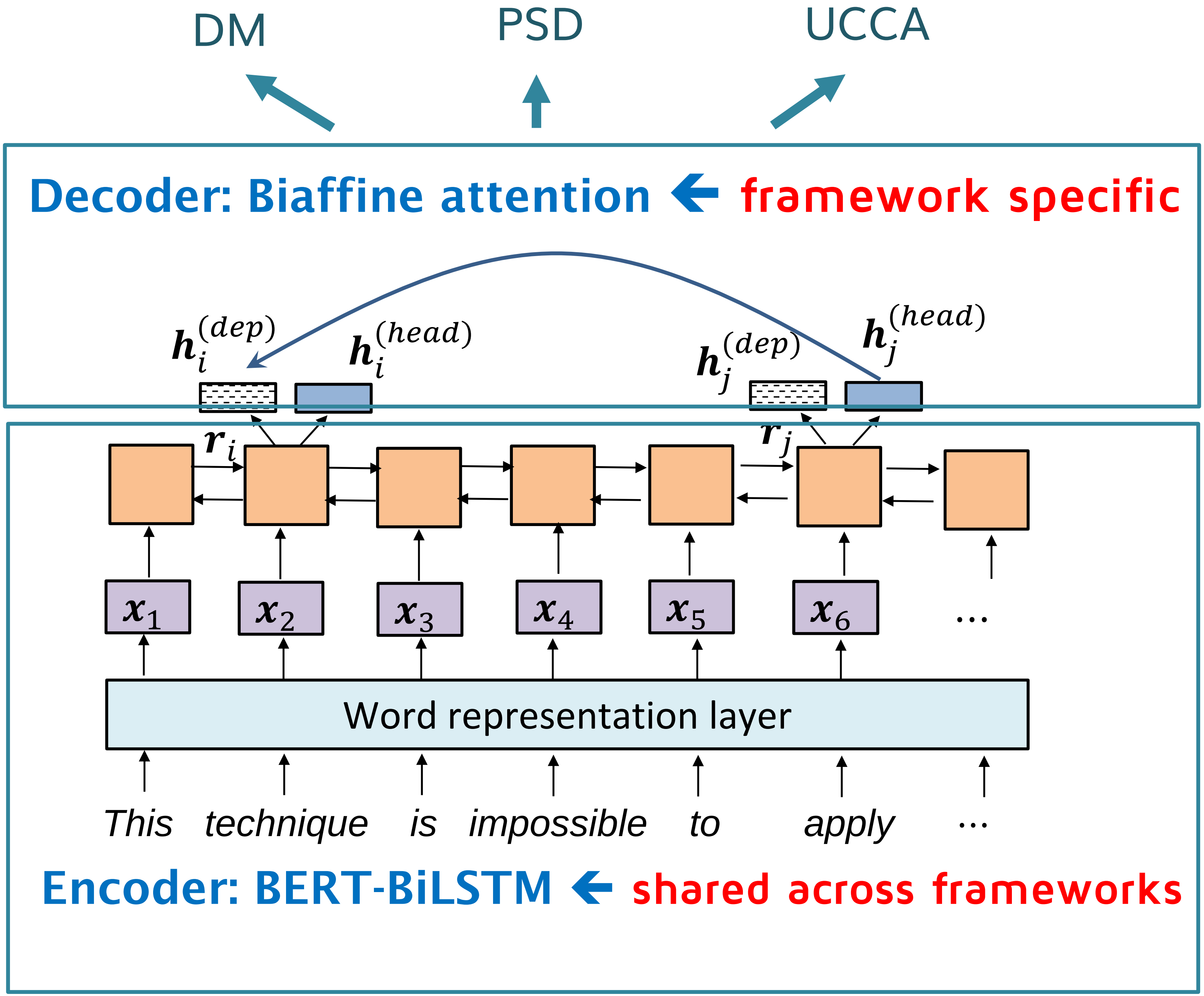
JBNU at MRP 2019: Multi-level Biaffine Attention for Semantic Dependency Parsing

Seung-Hoon Na, Jinwoo Min, Kwanghyeon Park
Dept. Computer Science, Jeonbuk National University

Jong-Hun Shin, Young-Kil Kim
Electronics and Telecommunication Research Institute (ETRI)

Introduction

- Our issue: **Multi-task learning** for **DM/PSD/UCCA**
  - To enable **multi-task learning,** we explicitly make **shared common components** in a neural network architecture across different frameworks
- Models
  - **Biaffine attention**: we propose a unified neural model for the DM/PSD/UCCA frameworks based on the biaffine attention [Dozat and Manning, 2017, 2018; Zhang et al., 2019]
  - **Multi-level biaffine attention**:
    - Motivated by the multi-level architecture of FusionNet in the machine reading comprehension task [Huang et al., 2018]

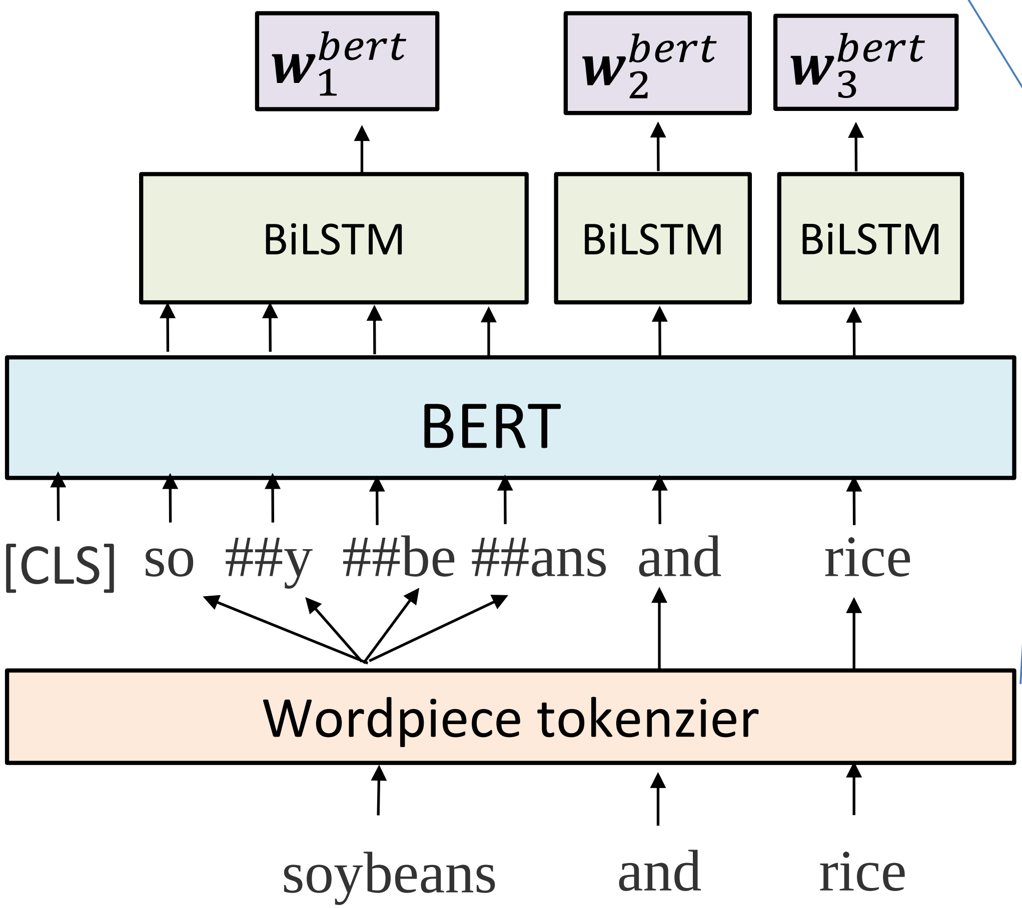


DM
PSD
UCCA
Decoder: Biaffine attention
framework specific
$h_i^{(dep)}$
$h_i^{(head)}$
$h_j^{(dep)}$
$h_j^{(head)}$
r_i
r_j
x_1
x_2
x_3
x_4
x_5
x_6
…
Word representation layer
This technique is impossible to apply …
Encoder: BERT-BiLSTM
shared across frameworks

Encoder: BERT-BiLSTM (shared across frameworks)

- **Word representation layer using BERT**

  Given a sentence, the BERT encoder is applied to its wordpieces and the encoded wordpiece-level represenations are composed to the word-level embeddings based on BiLSTM



BERT word-level embedding

$$\mathbf{x}_i = \left[\mathbf{w}_i^{bert}; \mathbf{e}_i^{glove}; \mathbf{e}_i^{POS}\right]$$

BiLSTM sentence encoding

$$\mathbf{r}_i = BiLSTM_i\left(\mathbf{x}_1 \cdots \mathbf{x}_n\right)$$

Decoder: Biaffine attention (framework specific)

- Biaffine attention is performed on the role-dependent representations to predict the existence of an edge and its labels

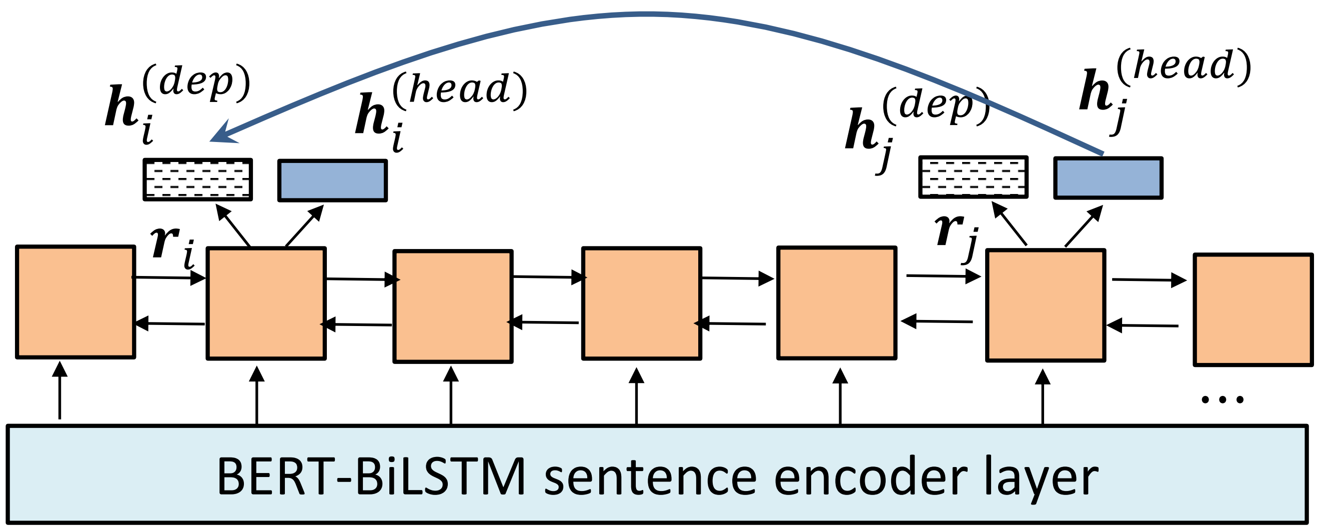
Biaffine attention

$$BiAff_m(\mathbf{x},\mathbf{y}) = \mathbf{x}^T\mathbf{U}^{[1:m]}\mathbf{y} + \mathbf{V}\begin{bmatrix}\mathbf{x}\\ \mathbf{y}\end{bmatrix} + \mathbf{b}$$

$$s_{i,j}^{(edge)} = BiAff_1^{(edge)}\left(\mathbf{h}_i^{(dep)}, \mathbf{h}_j^{(head)}\right)$$

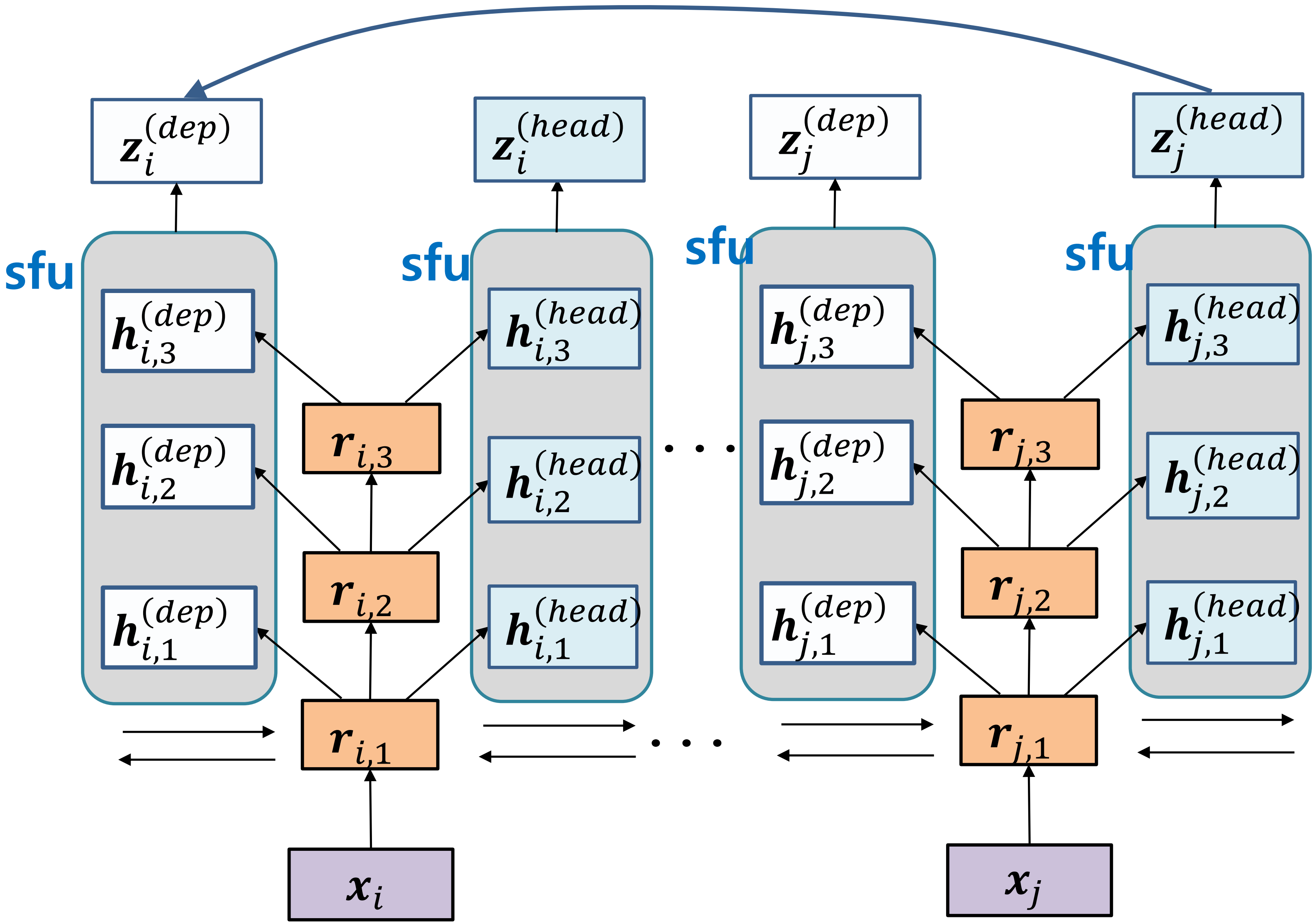
$$\mathbf{s}_{i,j}^{(label)} = BiAff_k^{(label)}\left(\mathbf{h}_i^{(l\text{-}dep)}, \mathbf{h}_j^{(l\text{-}head)}\right)$$

$$s_i^{(top)} = FFN^{(top)}\left(\mathbf{r}_i\right)$$



Multi-level Biaffine attention

The hidden representations at three levels are composed to the final hidden representation $z_i^{(dep)}$, $z_i^{(head)}$ using a **semantic fusion unit**



Preliminary Experiment

Framework	Train	Dev
DM	32091	3565
PSD	32091	3565
UCCA	5915	656

For more details, please visit our poster.
Thank you.

method	DM			PSD			UCCA		
	Top	UF	LF	Top	UF	LF	Top	UF	LF
Biaffine	93.67	92.08	90.86	95.97	90.50	78.21	72.60	69.67	65.17
BERT+Biaffine	95.06	93.85	93.00	96.89	92.30	80.24	77.09	74.85	70.15
BERT+Multi-level Biaffine	95.09	93.86	93.02	96.76	91.95	79.76	78.12	74.42	69.81
BERT+Biaffine+MTL	N/A	93.66	92.73	N/A	92.13	79.63	N/A	75.40	70.59

- **BERT+Biaffine performs** better than Biaffine, in particular, obtaining the increases of about 5% for UF and LF on the UCCA framework
- **BERT+Multi-level Biaffine** does not achieve any further improvements with respect to BERT-Biaffine model
- **BERT+Biaffine+MTL** only achieves small improvements on UCCA framework whereas no improvements on DM and PSD frameworks can be observed

CUHK at MRP2019: Transition-Based Parser with Cross-Framework Variable-Arity **Resolve** Action

Sunny Lai   Chun Hei Lo   Kwong Sak Leung   Yee Leung

香港中文大學
The Chinese University of Hong Kong

INSTITUTE OF
Future Cities

Abstract

Our system:

- **transition-based** parser
- **directed acyclic graph (DAG) to tree preprocessor**
- **cross-framework variable-arity RESOLVE action** that generalizes over five different representations.
- Although we ranked low in the competition, we have shown the current limitations and potentials of including variable-arity action in MRP and concluded with directions for improvements in the future.

Abstract

Arity: is the number of arguments or operands (No. of nodes) that the function takes (Wikipedia)

Standard shift reduce: 2, This paper: n

Our system:

- **transition-based** parser
- **directed acyclic graph (DAG) to tree preprocessor**
- **cross-framework variable-arity RESOLVE action** that generalizes over five different representations.
- Although we ranked low in the competition, we have shown the current limitations and potentials of including variable-arity action in MRP and concluded with directions for improvements in the future.

Motivation

MRP	F	Actions	Author
PSD	0	LEFT-REDUCE(L), RIGHT-SHIFT(L), NO-SHIFT,NO-REDUCE, LEFT-PASS(L), RIGHT-PASS(L), NO-PASS	(Wang et al., 2018)
UCCA	1	SHIFT, REDUCE, **NODE(X)**, LEFT-EDGE(X), RIGHT-EDGE(X), LEFT-REMOTE(X), RIGHT-REMOTE(X), SWAP, FINISH	(Hershcovich et al., 2017)
AMR	2	SHIFT, REDUCE, RIGHT-LABEL(R), LEFT-LABEL(R), SWAP, MERGE, PRED(N), **ENTITY(L)**, GEN(N)	(Guo and Lu, 2018)
*	*	**SHIFT**, **IGNORE**, **RESOLVE**	This paper

We introduce the cross-framework variable-arity **RESOLVE** action as:

1. there is **no need** to include additional **binarization** of the dependencies and reduce the number of transitions
2. It is also more natural to **consider the dependency of multiple nodes jointly** as meaning representations like semantic frames usually involve multiple arguments
3. Learn **cross-framework features to generalize** our model

Motivation

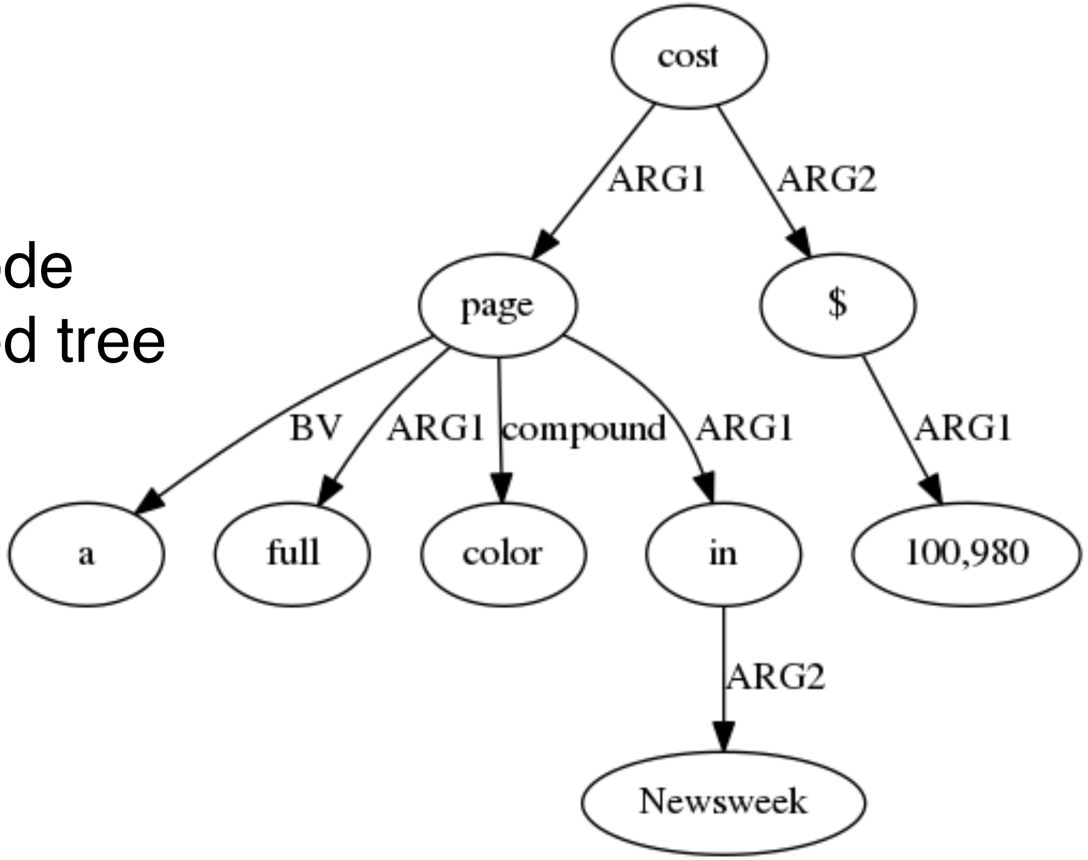
MRP	F	Actions	Author
PSD	0	LEFT-REDUCE(L), RIGHT-SHIFT(L), NO-SHIFT,NO-REDUCE, LEFT-PASS(L), RIGHT-PASS(L), NO-PASS	(Wang et al., 2018)
UCCA	1	SHIFT, REDUCE, **NODE(X)**, LEFT-EDGE(X), RIGHT-EDGE(X), LEFT-REMOTE(X), RIGHT-REMOTE(X), SWAP, FINISH	(Hershcovich et al., 2017)
AMR	2	SHIFT, REDUCE, RIGHT-LABEL(R), LEFT-LABEL(R), SWAP, MERGE, PRED(N), **ENTITY(L)**, GEN(N)	(Guo and Lu, 2018)
*	*	**SHIFT**, **IGNORE**, **RESOLVE**	This paper

We introduce the cross-framework variable-arity **RESOLVE** action as:

1. there is **no need** to include additional **binarization** of the dependencies and reduce the number of transitions
2. It is also more natural to **consider the dependency of multiple nodes jointly** as meaning representations like semantic frames usually involve multiple arguments
3. Learn **cross-framework features to generalize** our model

Approach

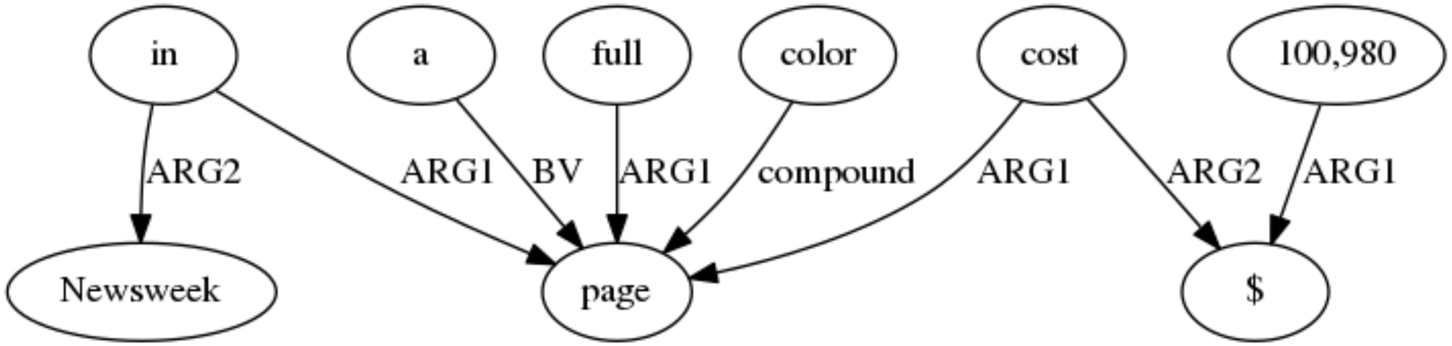
Top-node
oriented tree



MRP

DM

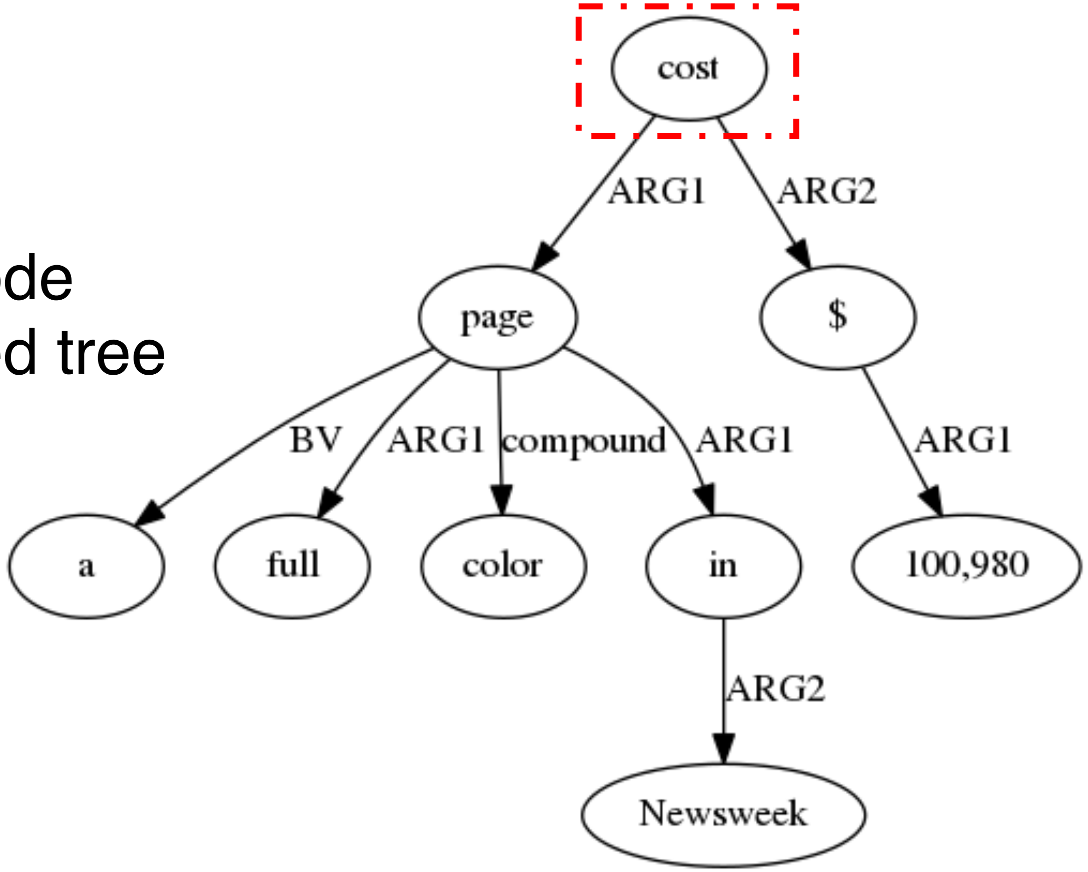
Top-node



Initial tokenized nodes queue: [A, full, ,, four-color, page, in, Newsweek, *will*, cost, $, 100,980]

Approach

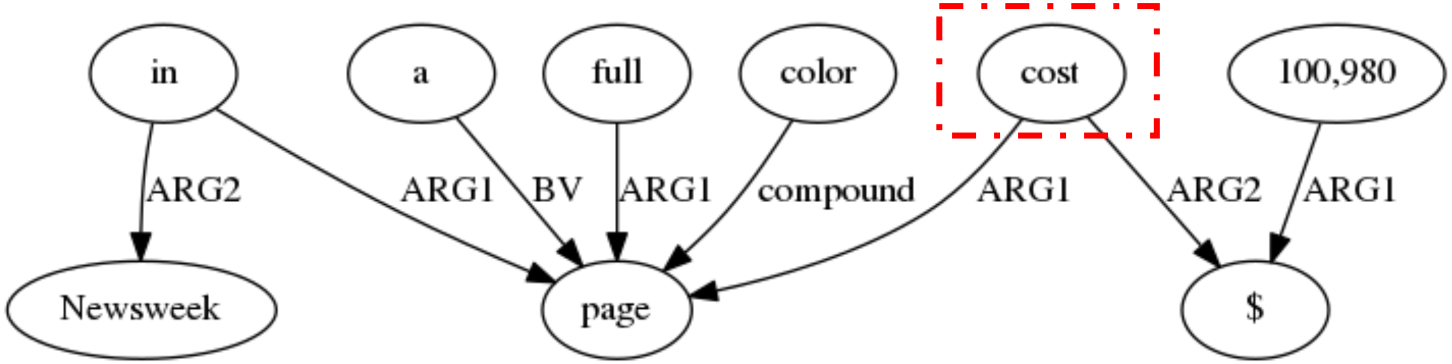
Top-node
oriented tree



MRP

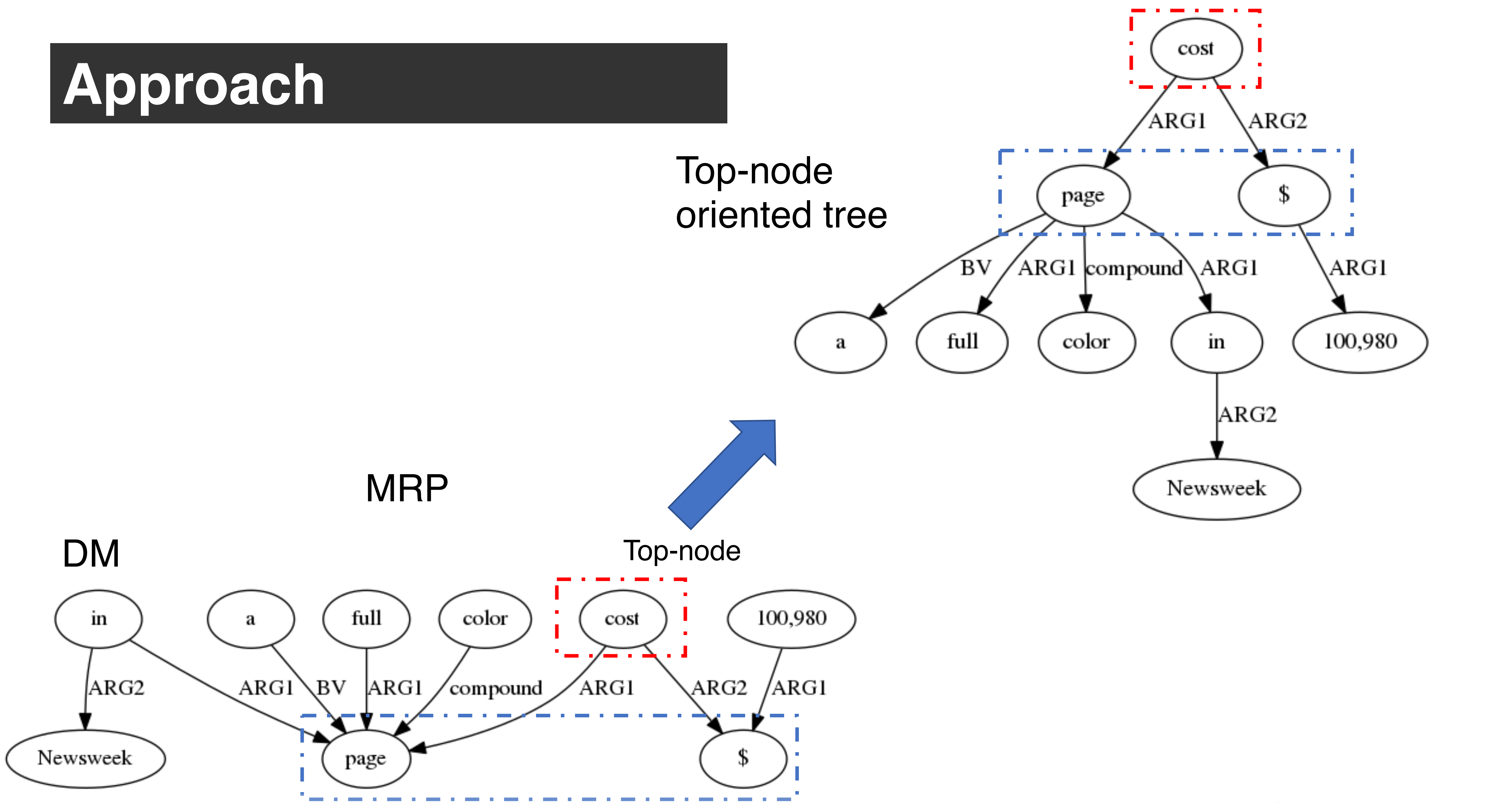
DM

Top-node



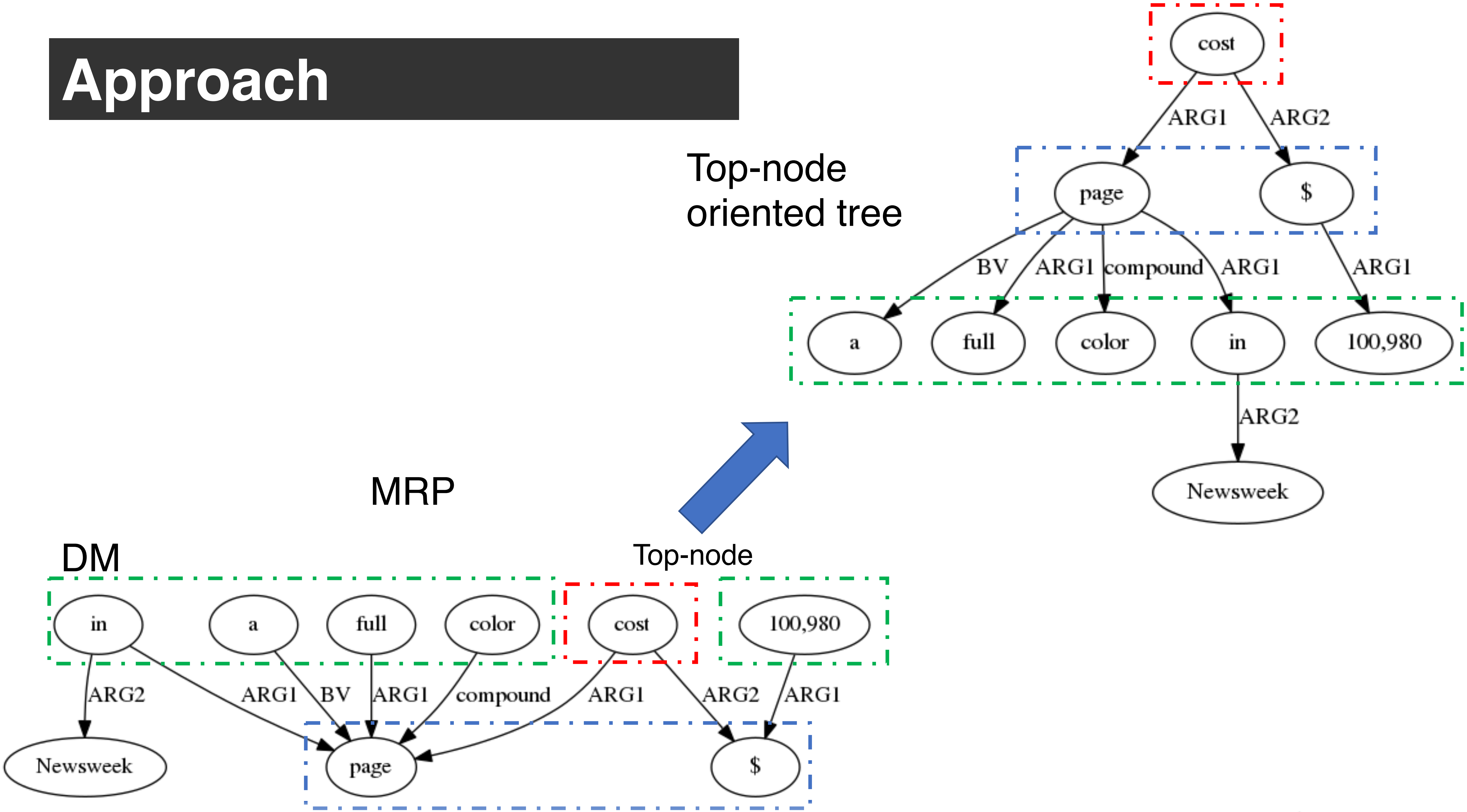
Initial tokenized nodes queue: [A, full, ,, four-color, page, in, Newsweek, *will*, cost, $, 100,980]

Approach



Initial tokenized nodes queue: [A, full, ,, four-color, page, in, Newsweek, *will*, cost, $, 100,980]

Approach



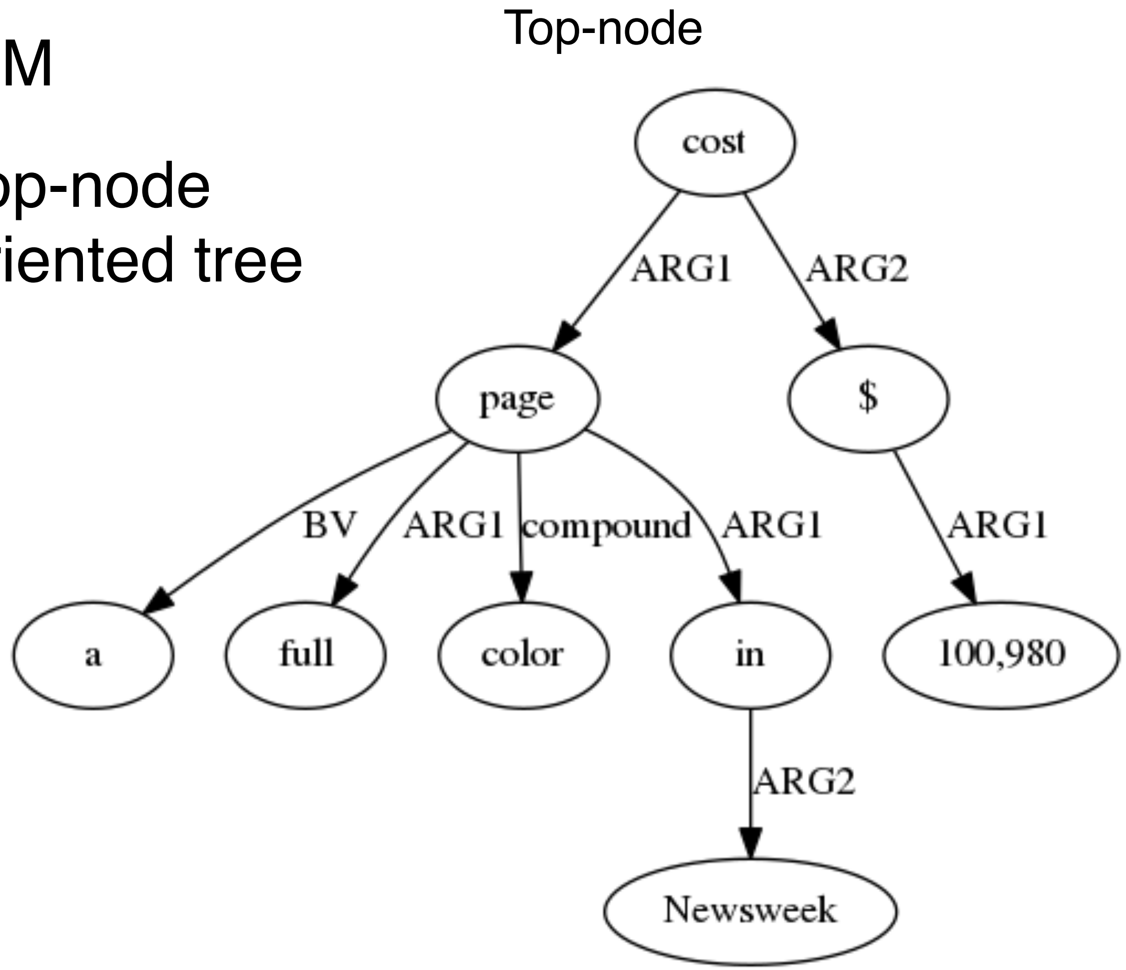
Initial tokenized nodes queue: [A, full, ,, four-color, page, in, Newsweek, *will*, cost, $, 100,980]

Approach

A node can be **RESOLVED**
when all its child is **RESOLVED**

DM

Top-node
oriented tree



RESOLVE:
- Predict node labels (framework specific)
- Build edges (framework specific)

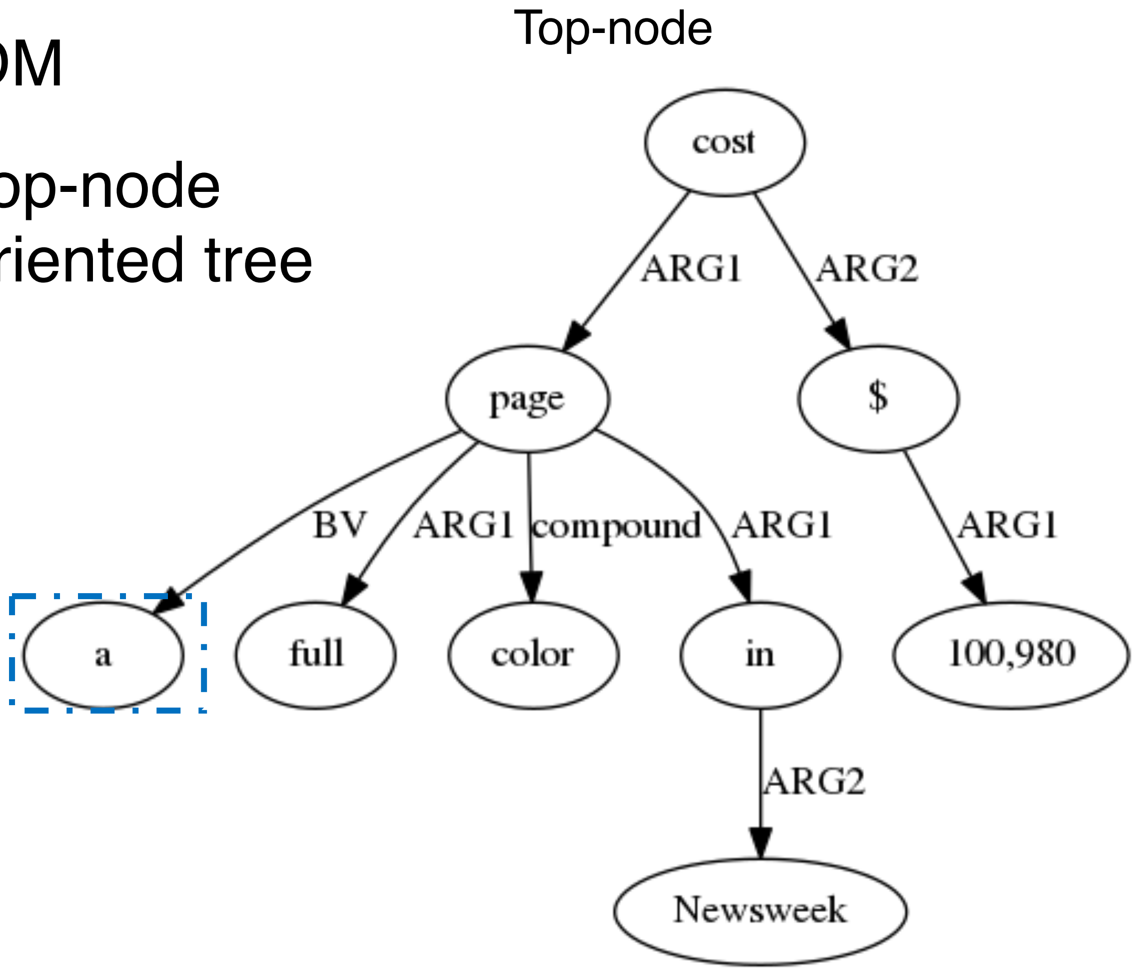
Initial tokenized nodes queue: [A, full, ,, four-color, page, in, Newsweek, *will*, cost, $, 100,980]

Approach

A node can be **RESOLVED** when all its child is **RESOLVED**

DM

Top-node oriented tree



RESOLVE:
- Predict node labels (framework specific)
- Build edges (framework specific)

Action	n	Stack
		[]
SHIFT		[A]
RESOLVE	1	[**a**]

Initial tokenized nodes queue: [A, full, ,, four-color, page, in, Newsweek, *will*, cost, $, 100,980]

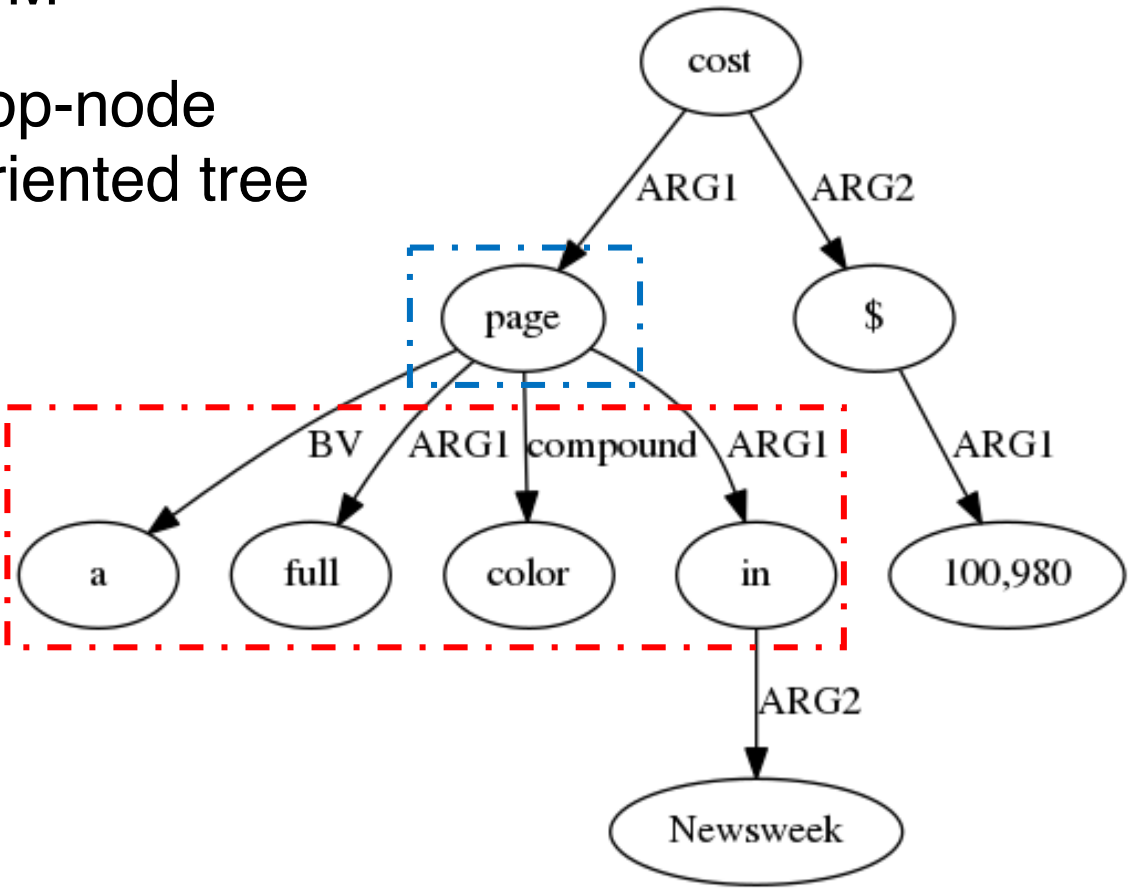
Approach

A node can be **RESOLVED** when all its child is **RESOLVED**

DM

Top-node oriented tree

Top-node



RESOLVE:
- Predict node labels (framework specific)
- Build edges (framework specific)

Action	n	Stack
		[a, full, color, page, **in**]
RESOLVE	5	[**page**]

$$\textbf{page} \xrightarrow{\text{BV}} \text{a}, \textbf{page} \xrightarrow{\text{ARG1}} \text{full}$$

$$\textbf{page} \xrightarrow{\text{compound}} \text{color}, \textbf{page} \xrightarrow{\text{ARG1}} \text{in}$$

Initial tokenized nodes queue: [A, full, ,, four-color, page, in, Newsweek, *will*, cost, $, 100,980]

Result

Submission\F1	tops	labels	properties	anchors	edges	attributes	all
TUPA(multi)	0.616	0.457	0.327	0.626	0.347	0.037	0.453
RESOLVER	0.502	0.365	0.317	0.568	0.095	0.00	0.378

- Cross-framework variable-arity actions are **hard to learn**
- **Information loss** happens when converting graphs to tree structures.
- **Model design** can still be improved.

Acknowledgement

- Organizers (especially Stephan Oepen)
- Anonymous Reviewers
- Teammates (especially Aaron Lo)
- Other participating teams (especially Longxu Dou)

We welcome any comments and suggestions!

slai@cse.cuhk.edu.hk

Hitachi at MRP 2019: Unified Encoder-to-Biaffine Network for Cross-Framework Meaning Representation Parsing

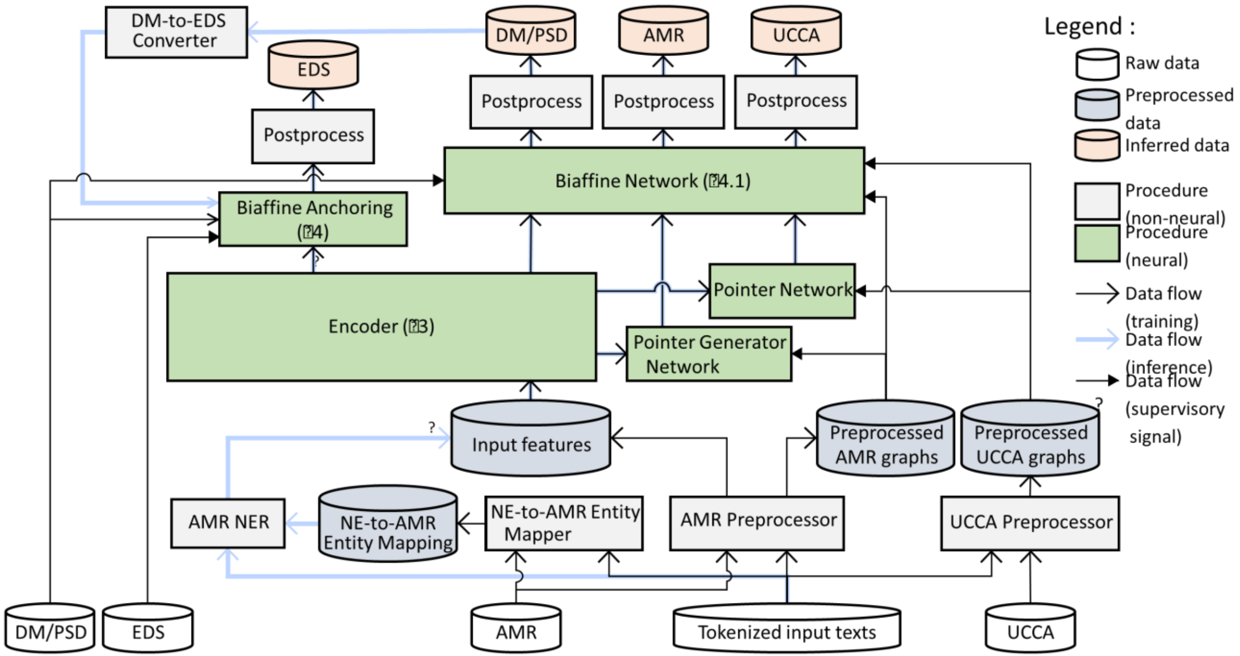
HITACHI Inspire the Next

Yuta Koreeda*, Gaku Morio*, Terufumi Morishita*, Hiroaki Ozaki*, Kohsuke Yanai (*equal contribution)

Our Approach

- Unify graph predictions with a single encoder-to-biaffine network
- Multi-task variant of the unified system (in post evaluation)

1. Extract task-independent contextualized token representations with shared encoder
2. Complement missing nodes
3. Predict edges and their labels with biaffine networks [Dozat+18]



Hitachi at MRP 2019: Unified Encoder-to-Biaffine Network for Cross-Framework Meaning Representation Parsing

HITACHI
Inspire the Next

Frame Specific Approaches

Framework	Biaffine Like Net.	Rule	Linear model	Generator
DM and PSD	Edge + Frame	Properties	-	-
EDS	Node anchor	Node & Edge gen.	Node & Edge gen.	-
UCCA	Edge	-	-	Non-terminal node: pointer network
AMR	Edge	Preprocess + serialize	-	Node: pointer-generator network

Results

Variant	Average	DM	PSD	EDS	UCCA	AMR
SFL	0.7575/5	0.9071/9	0.9064/3	0.8339/7	0.7014/6	0.4386/8
SFL(ensemble)*	0.7604/5	0.9102/9	0.9121/2	0.8374/7	0.7036/6	0.4386/8
BERT+SFL(NT)	0.7450/6	0.9038/9	0.9069/3	0.8301/7	0.6945/6	0.3896/8
BERT+MTL(NT)	0.7144/6	0.8726/9	0.8791/7	0.7987/7	0.6422/6	0.3794/9
BERT+MTL+FT(NT)	0.7507/5	0.9045/9	0.9054/4	0.8304/7	0.7126/6	0.4008/8

ÚFAL MRPipe at MRP2019: UDPipe Goes Semantic in the Meaning Representation Parsing Shared Task

Milan Straka, Jana Straková

November 3, 2019

Charles University in Prague
Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics

MRPipe Design

- start completely from scratch

MRPipe Design

- start completely from scratch
- **uniform** architecture for **simple directed graph** parsing
  - i.e., we consider at most one edge for all pairs of nodes in both directions
  - therefore, we can model trees, DAGs, even cycles
  - we could model **hypergraphs** (i.e., parallel edges) easily, but we did not yet evaluated it (~0.4% parallel edges in AMR, ~1.25% in UCCA)

MRPipe Design

- start completely from scratch
- **uniform** architecture for **simple directed graph** parsing
  - i.e., we consider at most one edge for all pairs of nodes in both directions
  - therefore, we can model trees, DAGs, even cycles
  - we could model **hypergraphs** (i.e., parallel edges) easily, but we did not yet evaluated it (~0.4% parallel edges in AMR, ~1.25% in UCCA)
- **no** linguistic information, structural constraints, dicts, ...

MRPipe Design

- start completely from scratch
- **uniform** architecture for **simple directed graph** parsing
  - i.e., we consider at most one edge for all pairs of nodes in both directions
  - therefore, we can model trees, DAGs, even cycles
  - we could model **hypergraphs** (i.e., parallel edges) easily, but we did not yet evaluated it (~0.4% parallel edges in AMR, ~1.25% in UCCA)
- **no** linguistic information, structural constraints, dicts, …
- rich pretrained embeddings – frozen **BERT** embeddings

MRPipe Parsing Algorithm

- consider **tokens** as **nodes**, **anchors** as **edges** to them

MRPipe Parsing Algorithm

- consider **tokens** as **nodes**, **anchors** as **edges** to them
- construct the graph layerwise by interleaving following two operations:

MRPipe Parsing Algorithm

- consider **tokens** as **nodes**, **anchors** as **edges** to them
- construct the graph layerwise by interleaving following two operations:
  - **AddNodes**: for every node, independently consider creating its new child or parent, with all its properties

MRPipe Parsing Algorithm

- consider **tokens** as **nodes**, **anchors** as **edges** to them
- construct the graph layerwise by interleaving following two operations:
  - **AddNodes**: for every node, independently consider creating its new child or parent, with all its properties
  - **AddEdges**: for every created node, independently consider connecting it to every other node (existing or new), generating all attributes if required

MRPipe Node Representation

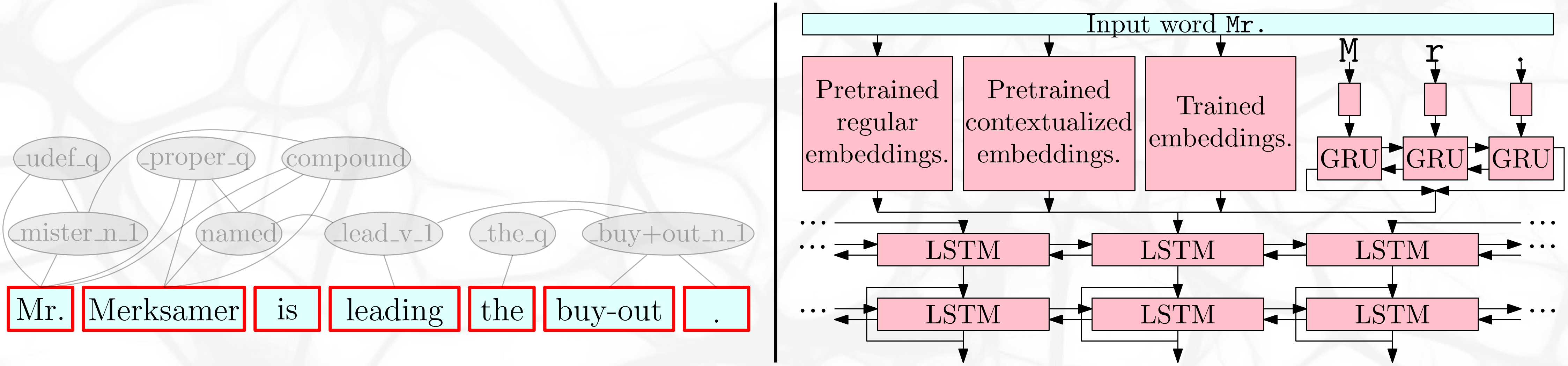
- each node is represented as
  - the **underlying token** which generated it (recursively)
  - embeddings of all node **properties**
  - embeddings of all **adjacent** edges **attributes**

MRPipe Node Representation

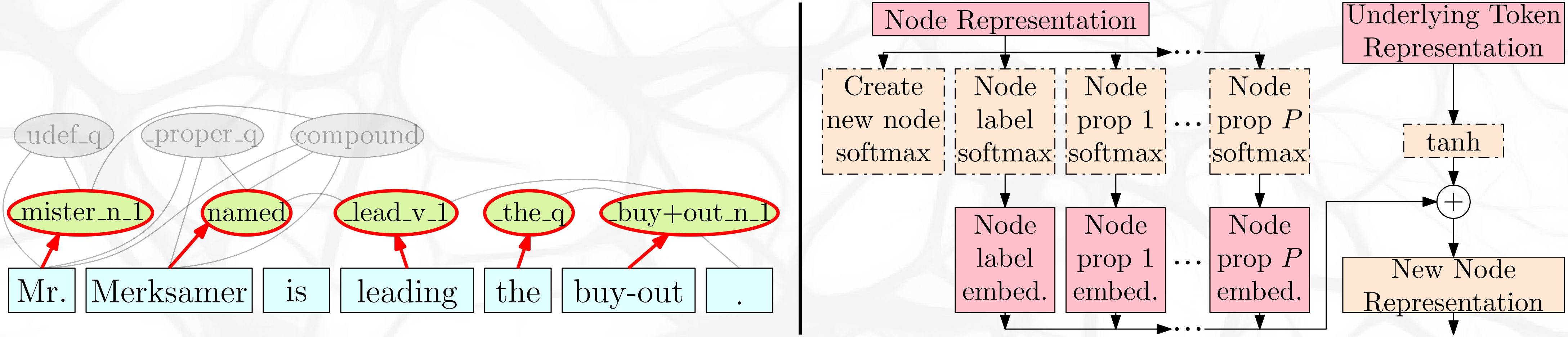
- each node is represented as
  - the **underlying token** which generated it (recursively)
  - embeddings of all node **properties**
  - embeddings of all **adjacent** edges **attributes**
- node properties can be encoded **relatively**
  - with respect to the **anchored tokens**

MRPipe Node Representation

- each node is represented as
  - the **underlying token** which generated it (recursively)
  - embeddings of all node **properties**
  - embeddings of all **adjacent** edges **attributes**
- node properties can be encoded **relatively**
  - with respect to the **anchored tokens**
  - automatically choosing **absolute** (e.g., POS, frames) or **relative** encoding (e.g., labels, cargs, op[1-9], ARG[1-9])

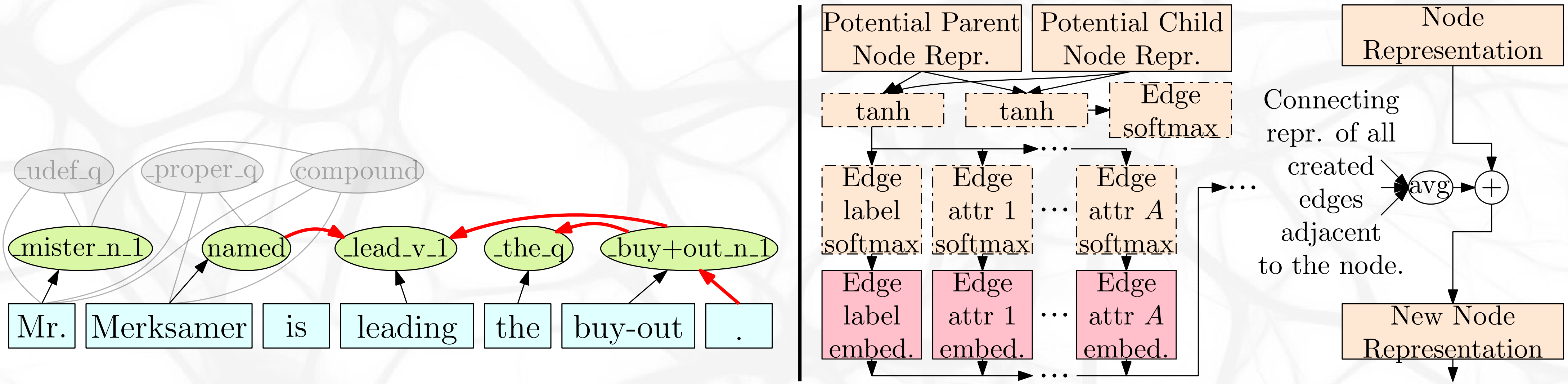


(a) Left: Initial configuration with tokens only. Right: Token representation encoder architecture.

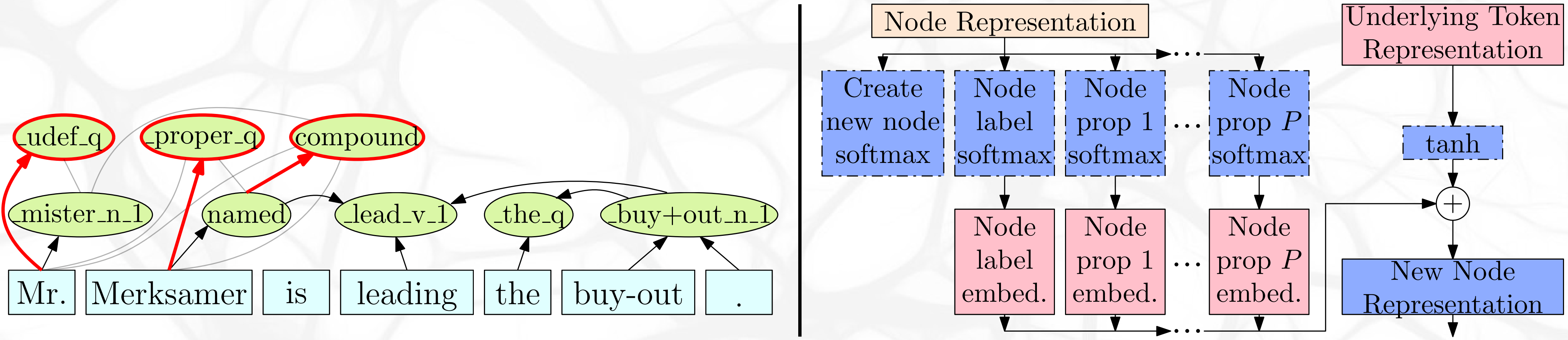


(b) Left: First **AddNodes** operation. Right: Architecture of the new node classifier and representation encoder.

MRPipe Example

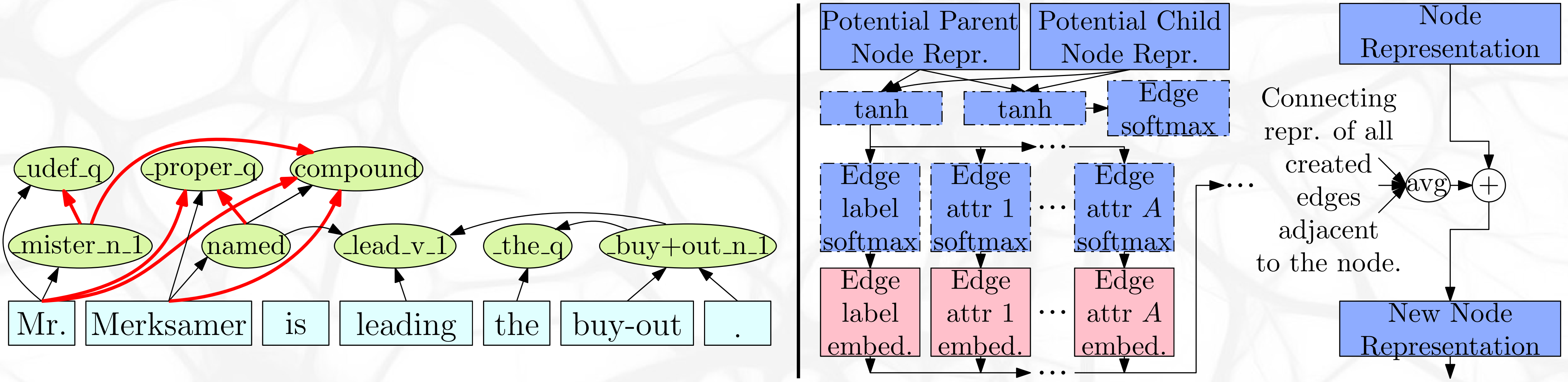


(c) Left: First **AddEdges** operation. Right: Architecture of the edge classifier and updated node representation encoder.



(d) Left: Second **AddNodes** operation. Right: Architecture of the new node classifier and representation encoder.

MRPipe Example



(e) Left: Second **AddEdges** operation. Right: Architecture of the edge classifier and updated node representation encoder.

MRPipe Results

- we utilized incorrect companion test data for three treebanks (the ones without anchors)
- our fixed submission ranked on a shared **3rd** place
- best overall **labels** and **properties** scores, worse **edges**

System	Tops		Labels		Properties		Anchors		Edges		Attributes		All	
Original ST submission	75.12%	6	63.99%	7	56.53%	6	69.53%	6	62.17%	7	7.85%	4	74.74%	6
Bugfix ST submission	81.47%	6	**73.06%**	**1**	**69.95%**	**1**	77.23%	3	73.89%	5	7.87%	4	83.96%	3
99% training data	80.59%	6	**73.06%**	**1**	**70.18%**	**1**	77.35%	3	74.27%	5	7.96%	4	84.14%	3
No BERT embeddings	70.50%	8	70.71%	4	67.01%	4	76.02%	4	65.02%	6	5.30%	6	78.99%	5
Ensemble	81.13%	6	**73.39%**	**1**	**70.82%**	**1**	77.57%	3	75.85%	4	8.28%	3	85.05%	3
HIT-SCIR (Che et al., 2019)	*90.41%*	*2*	*70.85%*	*3*	***69.86%***	***1***	*77.61%*	*2*	***79.37%***	***1***	***12.40%***	***1***	***86.20%***	***1***
SJTU–NICT (Li et al., 2019)	***91.50%***	***1***	*71.24%*	*2*	*68.73%*	*2*	***77.62%***	***1***	*77.74%*	*2*	*9.40%*	*2*	*85.27%*	*2*
SUDA–Alibaba (Zhang et al., 2019b)	*86.01%*	*5*	*69.50%*	*4*	*68.24%*	*3*	*77.11%*	*3*	*76.85%*	*3*	*8.16%*	*3*	*83.96%*	*3*
Saarland (Donatelli et al., 2019)	*86.70%*	*4*	***71.33%***	***1***	*61.11%*	*5*	*75.08%*	*5*	*75.01%*	*4*	—		*81.87%*	*4*

MRPipe Future Work

- allow anchoring to **sub-token** by addint attributes with character indices
- generate nodes **one-by-one** so that they are conditioned on already generated ones (important for constituency structure)
- better node representation
- better architecture of edge generation (not an independent decision for every edge)

Amazon at MRP 2019: Parsing Meaning Representations with Lexical and Phrasal Anchoring

Jie Cao[†]*, **Yi Zhang**[‡], **Adel Youssef**[‡], **Vivek Srikumar**[†]
[†]School of Computing, University of Utah
[‡]AWS AI, Amazon
{jcao, svivek}@cs.utah.edu, {yizhngn, adel}@amazon.com

Abstract

This paper describes the system submission of our team *Amazon* to the shared task on Cross Framework Meaning Representation Parsing (MRP) at the 2019 Conference for Computational Language Learning (CoNLL). Via extensive analysis of implicit alignments in AMR, we recategorize five meaning representations (MRs) into two classes: Lexical-Anchoring and Phrasal-Anchoring. Then we propose a unified graph-based parsing framework for the lexical-anchoring MRs, and a phrase-structure parsing for one of the phrasal-anchoring MRs, UCCA. Our system submis-

sometimes also with assumptions on underlying syntactic structures.

Anchoring is crucial in graph-based meaning representation parsing. Training a statistical parser typically starts with a conjectured alignment between tokens/spans and the semantic graph nodes to help to factorize the supervision of graph structure into nodes and edges. In our paper, with evidence from previous research on AMR alignments (Pourdamghani et al., 2014; Flanigan et al., 2014; Wang and Xue, 2017; Chen and Palmer, 2017; Szubert et al., 2018; Lyu and Titov, 2018), we propose a uniform handling of three meaning

SUDA–Alibaba at MRP 2019: Graph-Based Models with BERT

Yue Zhang[1], Wei Jiang[2], Qingrong Xia[2],
Junjie Cao[1], Rui Wang[1], Zhenghua Li[2*], Min Zhang[2]

[1] Alibaba Group, China

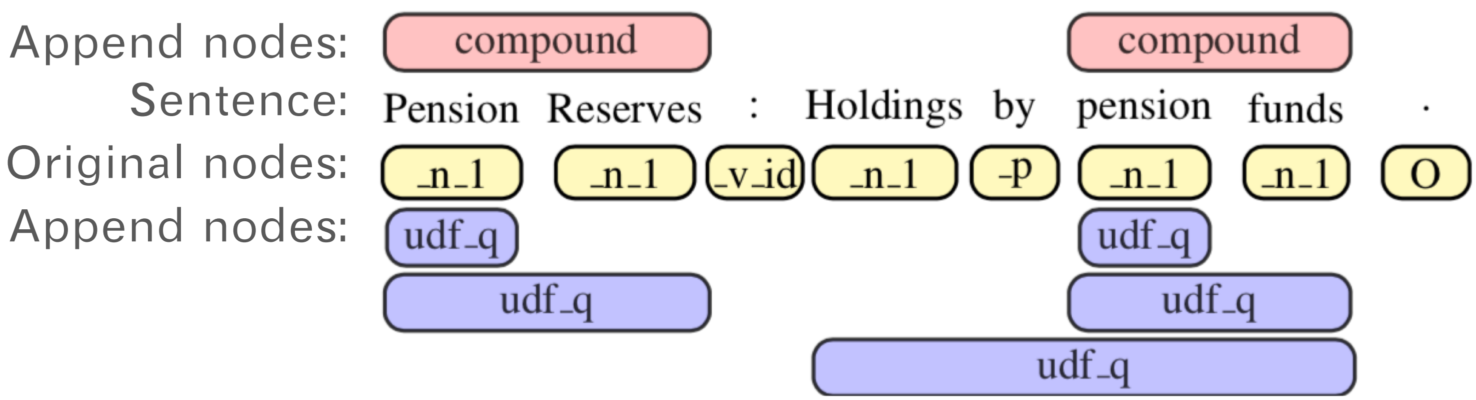
[2] School of Computer Science and Technology, Soochow University, China

poster id #14

EDS

Rank 1st

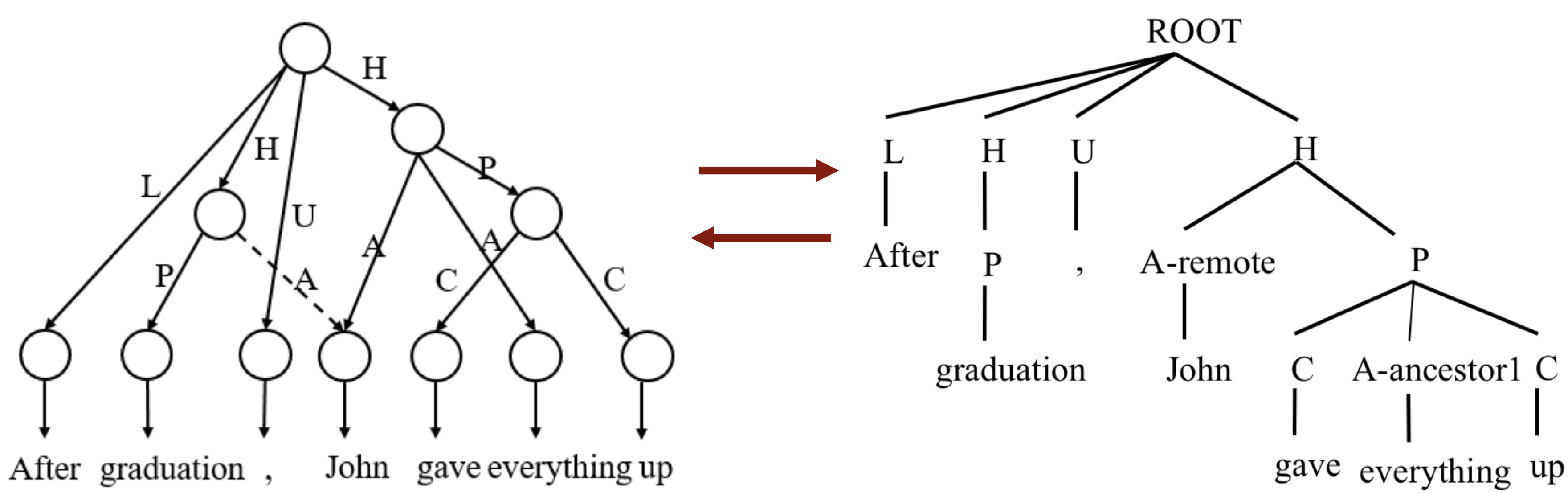
- **Node prediction**: original node & append node
  - Original nodes: take coarse POS tag and sense as joint label
  - Append nodes: predict begin/end index (anchor) and type for each node
- **Edge prediction**:
  - Compute the score of each edge relation between two nodes by the biaffine scorer



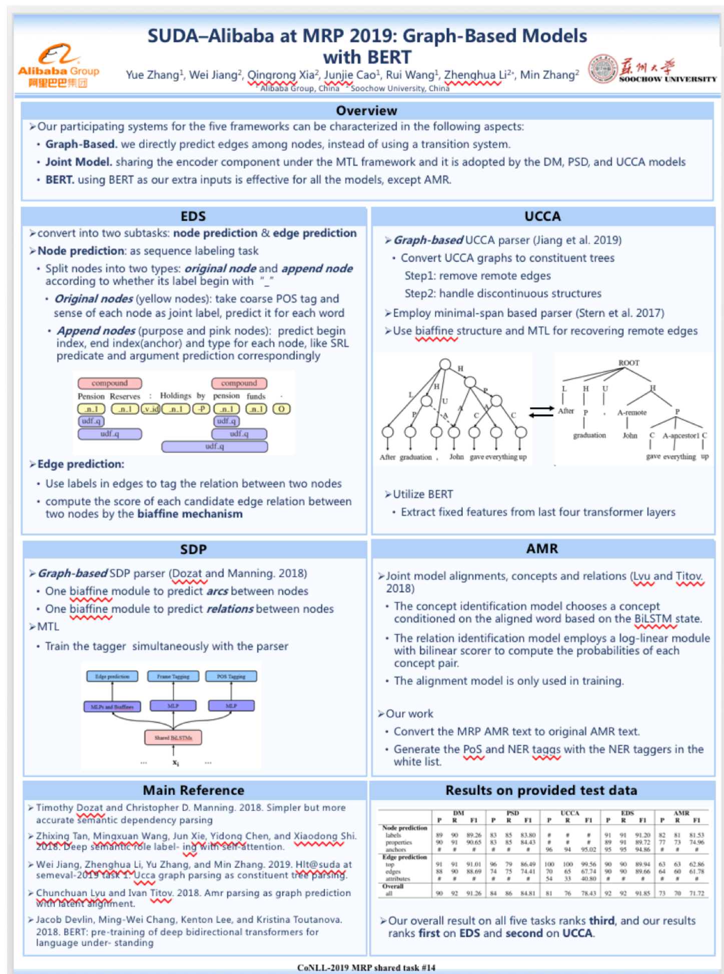
UCCA

Rank 2nd

- Convert UCCA to **constituent tree** (Jiang et al. 2019)
  - Remove remote edges
  - Handle discontinuous nodes
- Utilize minimal **span-based parser** (Stern et al. 2017)
- Remote recovery as a new task (multi-task learning)



Welcome to our poster for more details about our work!



poster id #14

ÚFAL–Oslo at MRP 2019: Garage Sale Semantic Parsing

Kira Droganova,† Andrey Kutuzov,‡
Nikita Mediankin† and Daniel Zeman†

†Charles University, Faculty of Mathematics and Physics, ÚFAL
‡University of Oslo, Faculty of Mathematics and Natural Sciences, Language Technology Group

†{droganova|mediankin|zeman}@ufal.mff.cuni.cz
‡andreku@ifi.uio.no

Garage Sale Semantic Parsing

Hao Peng, Sam Thomson, and Noah A. Smith. 2017. Deep multitask learning for semantic dependency parsing. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 2037–2048, Vancouver, Canada. Association for Computational Linguistics.

Jeffrey Flanigan, Dyer Chris, Noah A. Smith, and Jaime Carbonell. CMU at SemEval-2016 task 8: Graph-based AMR parsing with infinite ramp loss. In Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016), pp. 1202-1206. 2016.

Garage Sale Semantic Parsing

- create forward conversion scripts;
- create training/development splits;
- create/download all accompanying files;
- convert the data and train a model;
- create backwards conversion scripts.

Garage Sale Semantic Parsing

Peking at MRP 2019: Composition- and Factorization-Based Parsing for Elementary Dependency Structures

Yufei Chen, Yajie Ye and Weiwei Sun

Wangxuan Institute of Computer Technology
Peking University

November 3, 2019

北京大學
PEKING UNIVERSITY

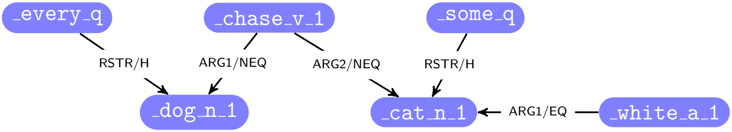
English Resource Semantics

(1) a. *Every dog chases some white cats.*

b. $\texttt{some}(\texttt{y}, \texttt{white}(\texttt{y}) \wedge \texttt{cat}(\texttt{y}), \texttt{every}(\texttt{x}, \texttt{dog}(\texttt{x}), \texttt{chase}(\texttt{e}, \texttt{x}, \texttt{y})))$

c. $\texttt{every}(\texttt{x}, \texttt{dog}(\texttt{x}), \texttt{some}(\texttt{y}, \texttt{white}(\texttt{y}) \wedge \texttt{cat}(\texttt{y}), \texttt{chase}(\texttt{e}, \texttt{x}, \texttt{y})))$

A compact graph-based representation of the two readings



Variables (e, x and y) are implicitly patched to the predicates that treat them as intrinsic variables (chase, dog and cat)

Elementary Dependency Structures Removing H, EQ, NEQ, etc.

北京大學 PEKING UNIVERSITY

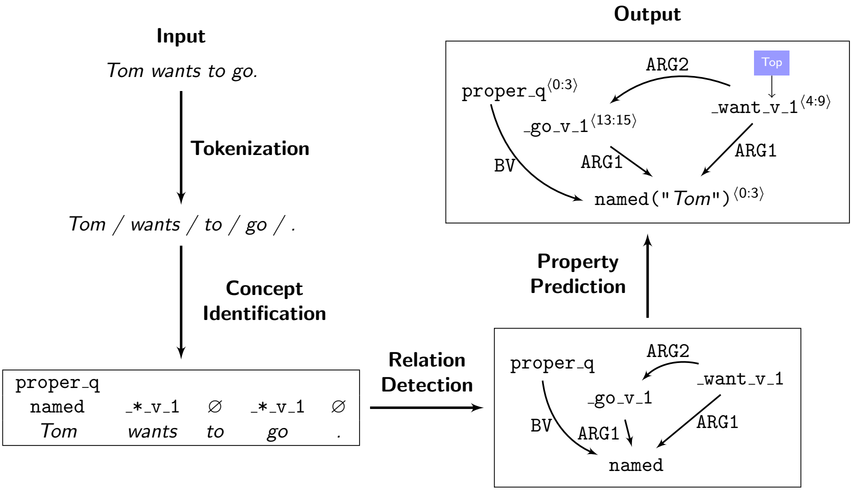
String-to-graph parsing approaches

- Factorization-based approach
- Composition-based approach
- Transition-based approach
- Translation-based approach

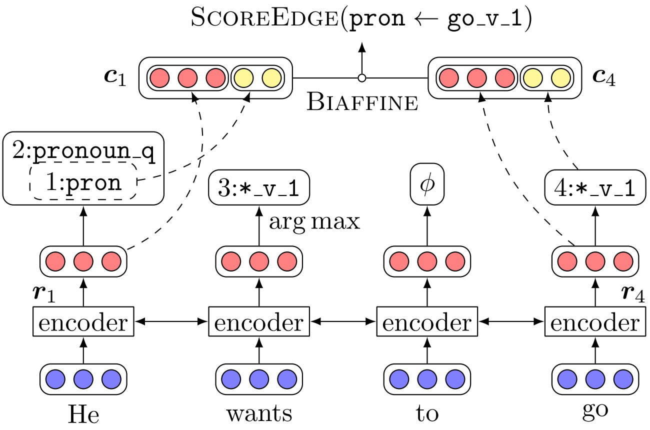
String-to-graph parsing approaches

- Factorization-based approach
- Composition-based approach
- Transition-based approach
- Translation-based approach

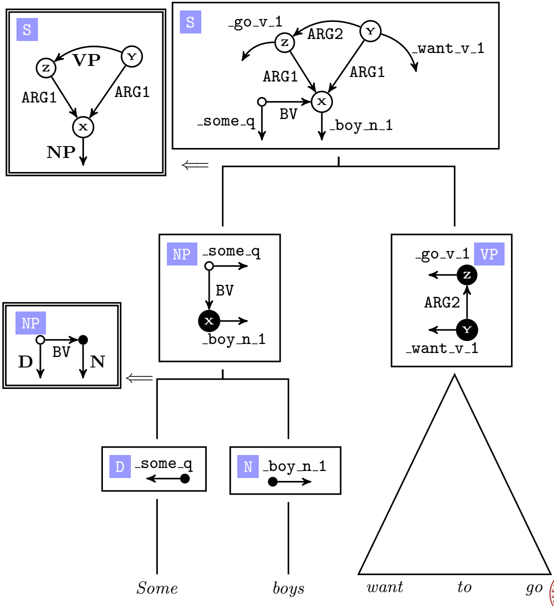
Factorization-based approach



Neural models



Composition-based approach



Thanks for your attention!

Meaning Representation Parsing Shared Task

Discussion

Discussion about the MRP task 2019→2020

- Some possible discussion points
  - Evaluation metric(s) – what to avoid? Improvements?
    - One main metric (even if approximate)? Several "equal" metrics (~ several "winners")
  - Extending the task
    - More languages (within the same frameworks)?
    - Additional frameworks?
    - Same text across frameworks ([mostly] evaluation only)?
  - Time schedule
    - How much time needed for "ingesting" whitelisted resources?
      - Or limit them to basics, like embeddings? Or not allow them at all?
      - Any tools to whitelist/blacklist?
  - Any general remarks?