ICL-HD at SemEval-2016 Task 8: Meaning Representation Parsing - Augmenting AMR Parsing with a Preposition Semantic Role Labeling Neural Network

Lauritz Brandt and **David Grimm** and **Mengfei Zhou** and **Yannick Versley**

Institute for Computational Linguistics
Ruprecht-Karls-Universität
Heidelberg

{brandt, grimm, zhou, versley}@cl.uni-heidelberg.de

Abstract

We describe our submission system to the SemEval-2016 Task 8 on Abstract Meaning Representation (AMR) Parsing. We attempt to improve AMR parsing by exploiting preposition semantic role labeling information retrieved from a multi-layer feed-forward neural network. Prepositional semantics is included as features into the transition-based AMR parsing system CAMR (Wang, Xue, and S. Pradhan 2015a). The inclusion of the features modifies the behavior of CAMR when creating meaning representations triggered by prepositional semantics. Despite the usefulness of preposition semantic role labeling information for AMR parsing, it does not have an impact to the parsing F-score of CAMR, but reduces the parsing recall by 1%.

1 Introduction

Progress in Natural Language Processing has led to a multitude of well-motivated tasks that each represent part of a sentence's meaning but result in a meaning description spread over separate, unconnected descriptions. These separate levels of semantic annotation, like co-reference or named entities, and the lack of simple human-readable corpora where whole sentence meanings are encoded led to the Abstract Meaning Representation (AMR) formalism (Banarescu et al. 2013). AMR structures capture sentence meanings with rooted, directed and labeled graphs where sentences with the same meaning receive the same AMR. These graphs are encoded in a bracketed format and can be visually represented in a human-understandable way (see Figure 1). AMR structures are organized with nodes representing concepts and the semantic relationships that hold between these concepts[1]. Hence, AMRs can be useful for every NLP component that relies on or exploits semantic meaning resources. Particular application areas are, among others, entity linking (Pan et al. 2015), event detection (Li et al. 2015) and machine translation. An example for a AMR graph is given in Figure 1: there is a concept RECOMMEND-01 which is the root of the graph and there is a concept OFFER-01 that stands in semantic relationship to RECOMMEND-01 with the edge ARG1.

We augment the existing AMR parser CAMR (Wang, Xue, and S. Pradhan 2015a) with a preposition semantic role labeling (prepSRL) neural network with the intention to improve the AMR graph creation accuracy. Prepositions in conjunction with their arguments make a crucial contribution to the meaning of sentences and are therefore a very intuitive supplement to AMR parsing. For example, see how in Figure 2 the meaning of the preposition *in* is involved in the creation of the AMR edge :LOCATION. *in* semantically expresses the *agency*'s spatial location and therefore triggers the identically named AMR edge :LOCATION. Prepositional semantics is a knowledge resource that has not yet been exploited for the domain of AMR parsing. Moreover, CAMR has problems in correctly creating AMR edges triggered by prepositional relations.

[1]Concepts can be PropBank framesets, English words or special words standing for quantities, entity types or logical expressions.

Proceedings of SemEval-2016, pages 1160–1166,
San Diego, California, June 16-17, 2016.

We should offer earthquake workers our full understanding.

```
(R / RECOMMEND-01
    :ARG1 (O / OFFER-01
        :ARG0 (W / WE)
        :ARG1 (U / UNDERSTAND-01
            :ARG0 W
            :MOD (F / FULL))
        :ARG3 (P / PERSON
            :ARG0-OF (W2 / WORK-01
            :MOD (E / EARTHQUAKE)))))
```

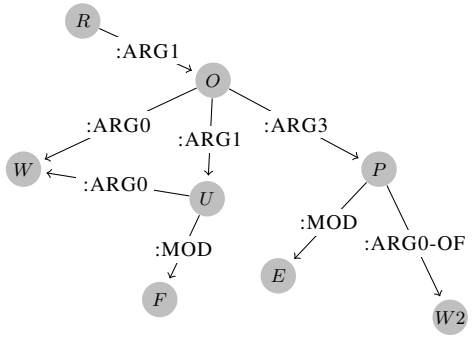

Figure 1: Example of an AMR visualization

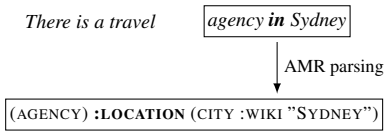

Figure 2: AMR edge activated by prepositional semantics

2 Related Work

The first attempt to automatically generate AMR structures from sentences was the work of Flanigan et al. (2014). They used a graph-based structured prediction algorithm with two stages: the first stage is a semi-Markov model concerned with identification of concepts, the second stage connects these concepts by finding the maximum spanning connected subgraph from a graph where all possible relations between concepts are realized. They achieve an F-score of 0.58 on the LDC2013E117 corpus. Werling, Angeli, and Manning (2015) improve the AMR parsing concept of Flanigan et al. (2014) by supporting the critical task of concept identification with a predefined set of actions for concept subgraph generation that are evoked after a statistical classification procedure. Besides graph-based approaches, there exist also other strategies on AMR parsing: Peng, Song, and Gildea (2015) learn synchronous hyperedge replacement grammar rules from string-graph pairs. An Earley algorithm with cube-pruning then performs string-to-AMR parsing with these rules. Pust et al. (2015) treat English and AMR as a language pair and use a machine translation approach to parse AMRs from sentences. They convert AMRs into into a grammar of string-to-tree rules that can be handled by syntax-based machine translation formalisms and use these rules with a bottom-up chart decoder to parse AMRs with given local features and a language model. Wang, Xue, and S. Pradhan (2015a) use a transition-based system that transforms dependency graphs into AMR structures by evoking specific actions at each reached state while traversing the dependency tree. As can be seen, there are many different point of views on AMR parsing.

There is nothing sad about old shells.

(a) Gold parse

```
(S3 / SAD
    :DOMAIN (N2 / NOTHING)
    :TOPIC (S2 / SHELL
        :MOD (O / OLD)))
```

(b) CAMR parse

```
(X3 / NOTHING
    :MOD (X4 / SAD
        :COMPARED-TO (X7 / SHELL
            :MOD (X6 / OLD))))
```

Figure 3: Examples of where CAMR's edge construction for prepositional phrases fails.

2.1 Motivation

The motivation for our system design comes from the error analysis of the transition-based AMR parser CAMR of Wang, Xue, and S. Pradhan (2015a). It turns out that the parser has difficulties on correctly identifying AMR relations which involve prepositional semantics. Therefore, we have chosen to aid CAMR with preposition semantic role labeling (prepSRL) in order to improve AMR parsing results.

2.2 Error Analysis of CAMR

Figure 3 shows a CAMR parse error: (b) should indicate a :TOPIC edge label for the edge between the concepts SAD and SHELL. This relation is semantically expressed by the preposition *about*. As can

be seen in Table 1, parsing precision and parsing recall is low for certain relations that can be evoked by prepositional semantics. Where the :LOCATION edge shows an arguably good CAMR performance, other relations like :TOPIC and :DESTINATION fail to be correctly parsed at all. We aim to improve CAMR's action selection for prepositions by introducing features representing prepositional semantics.

Relation	Gold	Parser	Corr.	Prec.	Rec.
:LOCATION	394	371	131	35.31%	33.25%
:PURPOSE	130	118	14	11.86%	10.77%
:TOPIC	70	44	6	13.64%	8.57%
:SOURCE	55	27	2	7.41%	3.64%
:DESTINATION	11	7	0	0.0%	0.0%
:INSTRUMENT	10	9	0	0.0%	0.0%

Table 1: Error analysis of CAMR based on parses of sentences taken from the *DEFT* corpus for 6 common AMR edges. **Gold**, **Parser**: # of relations in CAMR parse and Gold standard. **Corr.**: # of correctly parsed CAMR relations. **Prec./Rec.**: Precision/Recall.

3 System Description

3.1 Baseline System

We used the AMR parser CAMR of Wang, Xue, and S. Pradhan (2015a) as a starting point for our idea of supporting AMR parsing with prepSRL. It converts dependency trees into AMR graphs with a transition-based technique by evoking certain tree transforming actions at reached transition states. In the training procedure, the tokens of the input sentence are first aligned with the nodes of its gold AMR graph using the JAMR aligner (Flanigan et al. 2014). Such aligned AMR graphs are represented as span graphs storing token spans for AMR concept nodes. With these span graphs, a greedy transition-based mechanism learns to rewrite the dependency trees into AMR graphs. In order to learn these transformations, a transition system processes the nodes of the input dependency graphs in a bottom-up left-to-right fashion. It decides at each reached configurations which action to perform next in transforming the dependency graph into an AMR span graph. Configurations are defined as a tuple of buffers holding unprocessed nodes and unprocessed edges and the partial span graph parses for the current input sentence. While traversing the dependency tree, an

- **NEXT-EDGE-l_r:** assigns relation label to current edge and steps on to the next edge
- **SWAP-l_r:** swaps dependency relation between nodes (head transforms to dependent and vice versa)
- **REATTACH$_k$-l_r:** removes an arc, reattaches the former dependent to another node and assigns a label to the new arc
- **REPLACE HEAD:** replaces a head with its dependent
- **REENTRANCE$_k$-l_r:** links a node to another node in the subgraph and therefore has the ability to convert trees into graphs
- **MERGE:** merges two nodes into one node
- **NEXT-NODE-l_c:** assigns a concept label to current node and proceeds to next element in buffer
- **DELETE-NODE:** deletes a node and all its connections
- **INFER-l_c:** inserts a concept node between current node and parent

Figure 4: Possible actions for the transition system

averaged perceptron algorithm decides the actions to take by computing scores for all possible actions given specific features[2] and a weight vector. During test time, always the highest scoring action is chosen before moving on to the next state. During training, the algorithm will update the weight vector if it has chosen the wrong action and proceeds parsing with the correct one[3]. The core of the system are the set of actions that can be taken at states by the algorithm. Figure 4 shows an overview and a short description of the eight possible action types[4]. Actions alter the dependency tree by deleting or inserting nodes, merging two nodes into one, assigning relation labels and creating or modifying arcs. Therefore, the averaged perceptron can learn to do the right transformations to end up with a AMR span graph.

3.2 Semantic Role Labeling Features

Wang, Xue, and S. Pradhan (2015b) successfully improve their base system described in chapter 3.1

[2]Feature contexts vary from action type to action type and can include lemmas, words, named entities, POS tags, dependency labels and node span lengths. For a full list of features see (Wang, Xue, and S. Pradhan 2015a).

[3]I.e. the action that is necessary to build the gold span graph.

[4]For more detailed information about the action types see (Wang, Xue, and S. Pradhan 2015a).

by adding SRL features to their model[5]. We also included these features and added our prepSRL information in a similar way. The first SRL feature encodes an action's compatibility with the predicted frameset from the SRL system. For each action that predicts a concept label (*NEXT-NODE-*l_c), the predicted SRL frameset is compared to the candidate concept labels. If both match, the value of the feature is set to *true*. Therefore, the system will bias towards choosing the predicted SRL frameset as concept label[6]. The second feature encodes predicted SRL argumenthood for an action's current edge. For each action that predicts edge labels, the parser has access to the information whether current action's dependent is predicted by the semantic role labeler. Hence, the system will favor edges that are congruent with the semantic role labeler's edges.

3.3 Preposition Semantic Role Labels

The prepSRL information consists of an attachment according to a dependency parse and a semantic role label for the prepositional phrase head predicted by a neural network. The neural network is trained on data annotated with semantic role labels. A simple feed-forward neural network with one hidden layer is trained in a softmax regression framework on the role labels of Penn Treebank (PTB), the SemEval 2007 Task 6 corpus and the DEFT corpus[7][8]. Additionally a 'multi-task' neural network was trained on all three corpora simultaneously. The network architecture is sketched in Figure 5. All three prediction models share the same two hidden layers. As labels and number of labels vary, softmax regression is performed for each corpus separately.

The neural networks are fed a combination of word embedding[9] and a subset of the hand-crafted features proposed by (D. Hovy, Tratz, and E. H. Hovy 2010). These features include a binary indicator for token capitalization and a binary vector representation of both, the token's POS-tag and the supersense label according to WordNet[10]. For each sample, these features are extracted for the following tokens: the *preposition token*, the *previous token*, the *preceding verb*, the *preceding verb/adjective/noun*, the *dependency head*, the *dependency child* and a *heuristic child*, for which we choose the ensuing token. All corpora are parsed with the ClearNLP parser[11] to obtain POS-Tags and tokenization. For reasons of compatibility with the AMR-Parsing task, the dependency trees were extracted from parses by the BLLIP parser[12]. Corpus sizes and accuracy measures for the simple neural network can be seen in Table 2.

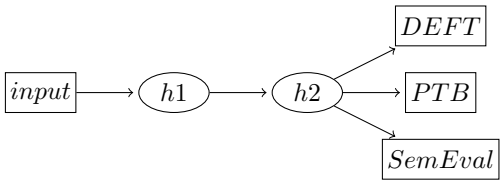

Figure 5: 'Multi-Task' neural network architecture. Each arrow represents a fully connected link to the next layer. Ellipsoid nodes are subject to non-linear activation, square nodes are subject to softmax regression. The three targets are trained alternating on mini-batches of 20 samples.

The neural network predicts a semantic role label for every prepositional phrase head. Comparing the different models by accuracy on these labels is difficult, as the target spaces and number of samples available differ. A list of valid target labels for each corpus can be seen in Table 3. The SemEval corpus comes with a total of 155 labels, where each preposition has a number of senses. These sense labels are reduced in number to create more meaningful target labels for the prepSRL task. The mapping scheme of Srikumar and Roth (2013) was implemented for this reduction. The DEFT corpus comes with a total

[5]They used the ASSERT SRL system described in (S. S. Pradhan et al. 2004).

[6]Remember that concept labels in AMR can be PropBank framesets.

[7]Used version: LDC2015E86: DEFT Phase 2 AMR Annotation R1

[8]All neural networks are trained with 200 hidden nodes per layer, a learning rate of 0.01 in a gradient descent batch learning environment. The weights are randomly initialized in $\left[\pm\sqrt{6\,/\,(input_{dim}\,+\,output_{dim})}\right]$. *Tanh* is used as non-linear activation function.

[9]The word embeddings are taken from (Pennington, Socher, and Manning 2014). For unknown words, a null vector is used instead.

[10]For unavailable information, a null vector is used instead for POS-tags, whereas for supersenses a 'unknown' supersense was introduced.

[11]For detailed information, see `https://github.com/clir/clearnlp`.

[12]For detailed information see (McClosky, Charniak, and Johnson 2006) and (Charniak 2000).

	PTB	SemEval	DEFT
Corpus Size	35,298	16,534	8,023
simple accuracy	91.69%	82.47%	72.69%
simple* accuracy	90.67%	80.12%	72.95%
multi-task accuracy	92.19%	72.28%	72.18%

Table 2: Preposition Semantic Role Labeling results. *Corpus size* is given in samples. Accuracy is measured on a test set from a random train/dev/test-split containing 90%/10%/10% of the samples respectively, where *simple accuracy* uses the simple neural network, *simple* accuracy* uses the simple neural network with an additional hidden layer and *multi-task accuracy* uses the 'multi-task' neural network, with the respective corpus' label set as output space and the corpus' test set.

They are creating traffic congestion in new places.

(C / CREATE-01
:ARG0 (T / THEY)
:ARG1 (C2 / CONGEST-01
:ARG2 (T2 / TRAFFIC)
:ARG3 (P / PLACE
:MOD (N / NEW)))))

Figure 6: Example of how an edge could be wrongly created.

PTB	LOC TMP DIR MNR PRP EXT BNF
DEFT	location manner mod name op1 op2 part-of poss purpose quant source time topic
SemEval	Activity Agent Attribute Beneficiary Cause Co-Participants Destination Direction EndState Experiencer Instrument Location Manner MediumOfCommunication Numeric ObjectOfVerb Opponent/Contrast Other Participant/Accompanier PartWhole PhysicalSupport Possessor ProfessionalAspect Recipient Separation Purpose Species Source StartState Temporal Topic Via

Table 3: Valid *prepSRL* target labels for each corpus. In total there are 7/13/32 labels for the PTB/DEFT/SemEval corpus respectively.

of 82 labels. We remove all labels with less than 200 samples from the corpus to ensure training quality.

Given this prepSRL system, the AMR parsing results are expected to be improved in the following way. In the AMR of Figure 6 currently the concept PLACE is the ARG3 of CREATE-01 but it should be the :LOCATION of CONGEST-01. Because the prepSRL feature has the ability to influence the edge creation and labeling actions of the transition system, the AMR parser can decide for the correct actions to take.

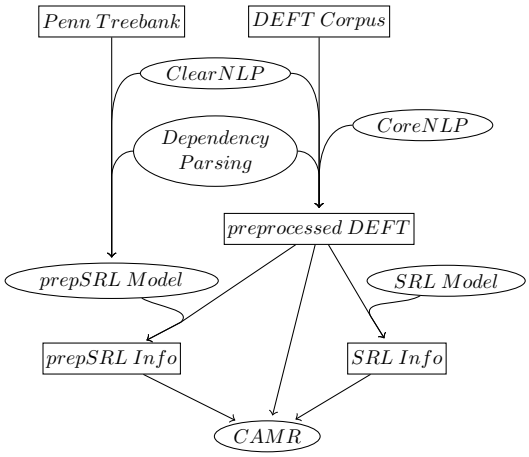

Figure 7: Preprocessing pipeline for our system. Round nodes depict models, rectangular nodes depict data.

4 Experiments

4.1 Experimental Setup

We first preprocessed the 16,831 sentences of the DEFT corpus training section (Knight et al. 2014) that we used for training CAMR with the prepSRL features. Preprocessing information for the AMR parser includes lemmas, POS tags, named entities and dependency parses[13]. In addition, we preprocessed the training sentences with the ClearNLP toolkit for the training of the neural network. We used the tokenization and POS tag components of ClearNLP and replaced the generated dependencies for compatibility reasons with the dependencies generated by the BLLIP parser.

After preprocessing, the alignments between the AMR graphs and their sentences were created with the JAMR aligner. ASSERT-generated SRL files were provided to us by Sameer Pradhan for the training and test inputs, enabling us to run CAMR with the SRL features. Separately, the neural network for prepSRL is trained with PTB-style preposition labels. We parsed our training corpora with the resulting model and generated the feature files for the prepSRL information. We trained CAMR in four different feature settings that are shown in Figure 8. The generated models were tested on the DEFT corpus test set that contains 1371 sentences.

[13]Lemmas, POS tags and named entities are generated by the CoreNLP toolkit (Manning et al. 2014). The dependency parses were generated with the BLLIP parser ((McClosky, Charniak,

(a)	CAMR
(b)	(a) + SRL features
(c)	(a) + prepSRL features
(d)	(b) + prepSRL features

Figure 8: Overview of the used training settings

Model	Prec.	Rec.	F-Score
(a): CAMR	67%	58%	62
(b): (a) +SRL	68%	58%	62
(c): (a) +prepSRL	67%	57%	62
(d): (b) +prepSRL	68%	57%	62

Table 4: Evaluation results

4.2 Results

We evaluated our approach of augmenting AMR parsing with a prepSRL system by using the standard evaluation measure for AMR parsing which is the Smatch evaluation metric to date[14]. Smatch uses semantic overlap between AMR parses to measure parsing accuracy. Results of the evaluation are given in Table 4. They reveal that the prepSRL features have a slightly negative influence on the parsing accuracy of CAMR. The Smatch F-score remains the same over all trained models, but the recall is reduced by 1% when adding the prepSRL features. The model with prepSRL achieves a Smatch score of 0.60 on the SemEval-2016 Task 8 test data. One possible explanation for the prepSRL results could be the ambiguity concerned with prepositions:

(1) *Establishing Models **in** Industrial Innovation.*

(2) *There is a travel agency **in** Sydney.*

In (1), *in* does not indicate an AMR :LOCATION relation, in contrast to its occurrence in (2). At the moment, our system cannot disambiguate between the two appearances of *in* according to the features used.

4.3 Error Analysis

A quantitative error analysis of our parser's output is shown in Table 5. If compared with the previous results in Table 1, the :LOCATION relation shows a minor improvement of precision and recall, where all other relations either show no difference or are parsed worse than before.

and Johnson 2006) and (Charniak 2000))

[14] see (Cai and Knight 2012)

Relation	Gold	Parser	Corr.	Prec.	Rec.
:LOCATION	394	374	135	36.10%	34.26%
:PURPOSE	130	109	9	8.26%	6.92%
:TOPIC	70	46	4	8.70%	5.71%
:SOURCE	55	39	2	5.12%	3.63%
:DESTINATION	11	4	0	0.0%	0.0%
:INSTRUMENT	10	9	0	0.0%	0.0%

Table 5: Error analysis of our parser (CAMR + SRL + prepSRL features) based on parses of sentences taken from the DEFT corpus test set. **Gold**, **Parser**: # of relations in CAMR parse and Gold standard. **Corr.**: # of correctly parsed CAMR relations. **Prec./Rec.**: Precision/Recall.

5 Conclusion

We extended the AMR parser CAMR (Wang, Xue, and S. Pradhan 2015a) with a neural network for prepSRL but did not reach improved AMR results using this method. In fact, the combination with prepSRL slightly reduced the recall of the system. This could be due to the fact that our prepSRL neural network generates parses for all preposition occurrences without disambiguating ambiguous prepositions. Future work has to find a better way to integrate prepSRL information into the architecture of CAMR. One possibility of this could be the refinement of the neural network where only prepositions receive a SRL parse that are likely to produce an AMR relation. Despite our results, we nevertheless think that the inclusion of prepositional semantics could improve AMR parsing results if used in an appropriate way.

Acknowledgments

We would like to thank Sameer Pradhan for providing us with SRL parses of all task data. Furthermore we would like to thank Chuan Wang for assisting us with detailed questions about CAMR code and parsing pipelines.

References

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

Cai, Shu and Kevin Knight (2012). *Smatch: an evaluation metric for semantic feature structures. submitted*.

Charniak, Eugene (2000). "A Maximum-entropy-inspired Parser". In: *Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference*. NAACL 2000. Seattle, Washington, pp. 132–139.

Flanigan, Jeffrey, Sam Thomson, Jaime G. Carbonell, Chris Dyer, and Noah A. Smith (2014). "A Discriminative Graph-Based Parser for the Abstract Meaning Representation". In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1426–1436.

Hovy, Dirk, Stephen Tratz, and Eduard H. Hovy (2010). "What's in a Preposition? Dimensions of Sense Disambiguation for an Interesting Word Class". In: *COLING 2010, 23rd International Conference on Computational Linguistics, Posters Volume, 23-27 August 2010, Beijing, China*, pp. 454–462.

Knight, Kevin, Laura Baranescu, Claire Bonial, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Daniel Marcu, Martha Palmer, and Nathan Schneider (2014). *Abstract Meaning Representation (AMR) Annotation Release 1.0*.

Li, Xiang, Thien Huu Nguyen, Kai Cao, and Ralph Grishman (2015). "Improving Event Detection with Abstract Meaning Representation". In: *Proceedings of the First Workshop on Computing News Storylines*. Beijing, China: Association for Computational Linguistics, pp. 11–15.

Manning, Christopher D., Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky (2014). "The Stanford CoreNLP Natural Language Processing Toolkit". In: *Association for Computational Linguistics (ACL) System Demonstrations*, pp. 55–60.

McClosky, David, Eugene Charniak, and Mark Johnson (2006). "Effective Self-training for Parsing". In: *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*. HLT-NAACL '06. New York, New York, pp. 152–159.

Pan, Xiaoman, Taylor Cassidy, Ulf Hermjakob, Heng Ji, and Kevin Knight (2015). *Unsupervised Entity Linking with Abstract Meaning Representation*.

Peng, Xiaochang, Linfeng Song, and Daniel Gildea (2015). *A Synchronous Hyperedge Replacement Grammar based approach for AMR parsing*.

Pennington, Jeffrey, Richard Socher, and Christopher D. Manning (2014). "GloVe: Global Vectors for Word Representation". In: *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543.

Pradhan, Sameer S, Wayne H Ward, Kadri Hacioglu, James H Martin, and Dan Jurafsky (2004). "Shallow Semantic Parsing using Support Vector Machines". In: *HLT-NAACL 2004: Main Proceedings*. Ed. by Daniel Marcu Susan Dumais and Salim Roukos. Boston, Massachusetts, USA: Association for Computational Linguistics, pp. 233–240.

Pust, Michael, Ulf Hermjakob, Kevin Knight, Daniel Marcu, and Jonathan May (2015). "Parsing English into Abstract Meaning Representation Using Syntax-Based Machine Translation". In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, pp. 1143–1154.

Srikumar, Vivek and Dan Roth (2013). "Modeling Semantic Relations Expressed by Prepositions". In: 1, pp. 231–242.

Wang, Chuan, Nianwen Xue, and Sameer Pradhan (2015a). "A Transition-based Algorithm for AMR Parsing". In: *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Denver, Colorado: Association for Computational Linguistics, pp. 366–375.

– (2015b). "Boosting Transition-based AMR Parsing with Refined Actions and Auxiliary Analyzers". In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Beijing, China: Association for Computational Linguistics, pp. 857–862.

Werling, Keenon, Gabor Angeli, and Christopher D. Manning (2015). "Robust Subgraph Generation Improves Abstract Meaning Representation Parsing". In: *CoRR* abs/1506.03139.