CU-NLP at SemEval-2016 Task 8: AMR Parsing using LSTM-based Recurrent Neural Networks

William R. Foland Jr.
OK Robot Go, Ltd.
5345 Dunraven Circle
Golden, Co, 80403, USA
bill.foland@okrobotgo.com

James H. Martin
Department of Computer Science and
Institute of Cognitive Science
University of Colorado
Boulder, CO 80309
James.Martin@colorado.edu

Abstract

We describe the system used in our participation in the AMR Parsing task for SemEval-2016. Our parser does not rely on a syntactic pre-parse, or heavily engineered features, and uses five recurrent neural networks as the key architectural components for estimating AMR graph structure.

1 Introduction

Abstract Meaning Representation, or AMR (Banarescu et al., 2012) is a graph-based representation of the meaning of sentences which incorporates linguistic phenomena such as semantic roles, coreference, negation, and more.[1]

The process of creating AMR's for sentences is called AMR Parsing. We used an early version of the system described in this paper to generate our submission to the Semeval-2016 Meaning Representation Parsing Task.[2]

The details of our system will be explained using this example sentence: ***France plans further nuclear cooperation with numerous countries .*** A graphical depiction is shown in Figure 1.

The system extracts features from the sentence which are processed by a form of recurrent neural network called BDLSTM to create a set of AMR concepts. Features from these concepts are processed by a pair of BDLSTM networks to compute relation probabilities. All concepts are then connected using an iterative, greedy algorithm to compute the set of relations in the AMR. Another two BDLSTM networks compute attribute and name categories to complete the estimation of AMR element probabilities.

[1]http://amr.isi.edu/language.html

[2]http://alt.qcri.org/semeval2016/task8/#

2 Related Work

Most current AMR parsers assume input that has undergone varying degrees of syntactic analysis, ranging from simple part-of-speech tagging to more complex dependency or phrase-structure analysis. (Wang et al., 2015; Vanderwende et al., 2015; Peng et al., 2015; Pust et al., 2015; Artzi et al., 2015; Flanigan et al., 2014; Werling et al., 2015). In contrast, we follow the spirit of minimal feature extraction using pre-trained word embeddings, as in (Collobert et al., 2011) and a recurrent network architecture similar to that described in (Zhou and Xu, 2015).

3 System Architecture

3.1 Feature Extraction

In our system, all features are represented by embedding vectors, trained and stored in lookup tables. Word feature embeddings are mapped from the words in the sentence, and are trained with back propagation just like other parameters in the network. They are initialized with vectors which are pre-trained on large corpora of english text, we use the word embeddings from (Collobert et al., 2011).

The only explicit features not derived from the raw input are features based on named entity recognition (NER). We first use the Univ. of Illinois Wikifier to find and classify named entities and then encode these features as embeddings.

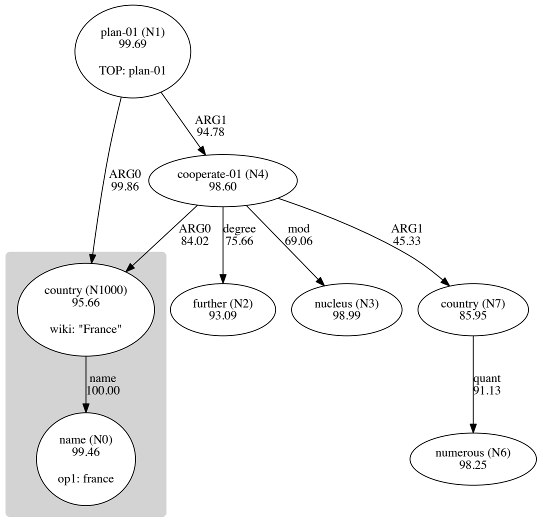

Figure 1: Graphical Representation of the Reference AMR (annotated with probabilities expressed as percentage.)

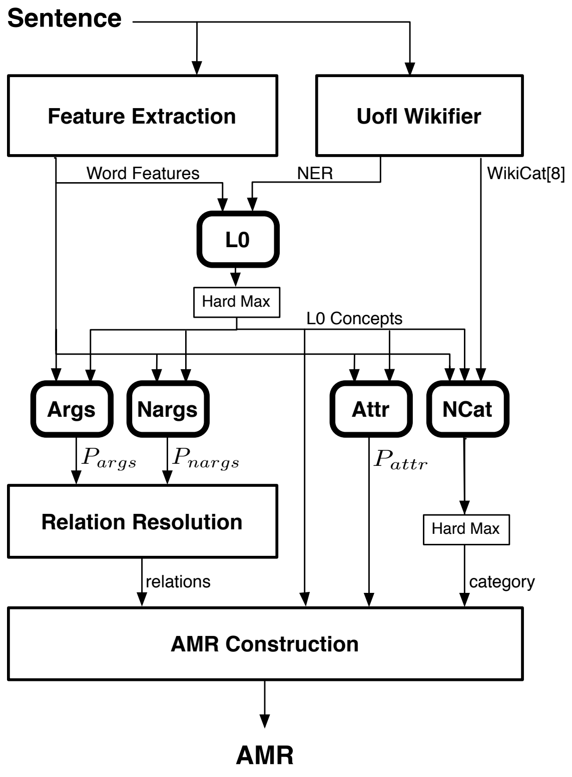

Figure 2: AMR Parser Architecture, BDLSTM networks in bold.

3.2 Neural Networks

Unlike relatively simple sequence processing tasks like part-of-speech tagging and NER, semantic analysis requires the ability to keep track of relevant information that may be arbitrarily far away from the words currently under consideration. Fortunately, recurrent neural networks (RNNs) are a class of neural architecture that use a form of short-term memory in order to solve this semantic distance problem. Basic RNN systems have been enhanced with the use of special memory cell units, referred to as Long Short-Term Memory neural networks, or LSTM's (Hochreiter and Schmidhuber, 1997). Such systems can effectively process information dispersed over hundreds of words (Schmidhuber et al., 2002; Gers et al., 2001).

Bidirectional LSTMs (BDLSTM) networks are LSTMs that are connected so that both future and past words in the sentence can be examined. We use the LSTM cell as described in (Graves et al., 2013), Figure 3, configured in a Bi-directional structure, called BDLSTM (Zhou and Xu, 2015), shown in Figure 4 as the core network in our system. Five BDLSTM Neural Networks comprise our parser.

3.2.1 Level 0 Concepts BDLSTM Network (L0)

The first step in our process is to create the set of concepts (nodes) that form the basis for any AMR representation; we call these Level 0, or L0, concepts. For the most part in current AMR training data, these concepts are in a direct relationship to words and sequences of words in a sentence. The task of the L0 network is, therefore, to take the input sequence of words and produce an output sequence of IOB tags that identify and classify the concepts in the AMR output.

For training, AMR concepts are first aligned to words using an AMR-to-word alignment algorithm. We used the alignment provided in the SemEval dataset. In cases where multiple concepts are associated with the same word, we use only the lower level concept and ignore upper level concept(s).

The system classifies each L0 concept as predicate or non-predicate, and predicts the PropBank sense for the predicates. AMR concepts are either

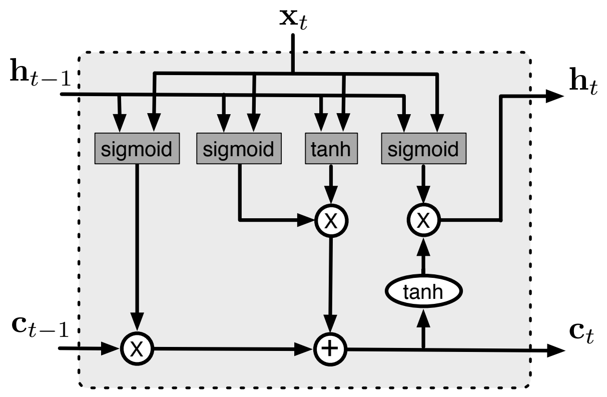

Figure 3: LSTM Cell.
An "unrolled" representation of an LSTM Cell. Rectangles represent linear layers followed by the labelled non-linearity. Each cell learns how to weigh, or gate, the input, previous cell memory, and output.

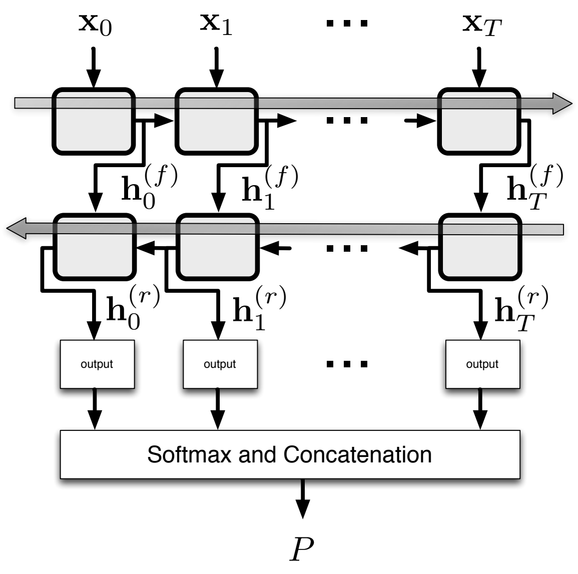

Figure 4: Bi-Directional LSTM.
A general diagram of a BDLSTM network, showing the feature input vectors x_i, the forward layer (f) and the reverse layer (r). The network generates vectors of log likelihoods which are converted to an array of probabilities.

English words (boy), PropBank framesets (plan-01), or special keywords. A translation table was created from training data by calculating the most probable AMR concept, given the sentence word and the general concept identifier.

The most common multilevel cases, a subgraph composed of a named entity, its related category, and a wiki link when available, are identified as exceptions and tagged as name concepts, which will be expanded.[3]

The features used in the L0 nework are:

- word: 130Kx50, the word embedding
- suffix: 430x5, embedding based on the final two letters of each word.
- caps: 5x5, embedding based on the capitalization pattern of the word.
- NER: 5x5, indexed by NER from the Wikifier, 'O', 'LOC', 'ORG', 'PER' or 'MISC'.

The L0 Network produces probabilities for 19 BIOES tagged concept types, and the highest probability tag is chosen for each word, as shown for the example sentence in Table 1.

3.2.2 Predicate Argument Relations BDLSTM Network (Args)

The Args Network is run once for each predicate concept, and produces a matrix P_{args} which defines the probability of a type of predicate argument relation from an L0 predicate concept to any other L0 concept. (See, for example, ARG0 and ARG1 relations in Figure 1.), prior to the identification of any relations.[4] The matrix has dimensions a by c, where a is the number of non-arg relations to be identified, and c is the total number of concepts.

The Args features, calculated for each source predicate concept, are:

[3]For example *France* in the shaded section of Figure 1.

[4]relation probabilities change as hard decisions are made, see section 3.4

words	BIOES	Prob	kind
France	S_Named	0.995	txNamed
plans	S_Pred-01	0.997	plan-01
further	S_NonPred	0.931	further
nuclear	S_NonPred	0.990	nucleus
cooperation	S_Pred-01	0.986	cooperate-01
with	O	1.000	O
numerous	S_NonPred	0.982	numerous
countries	S_NonPred	0.860	country
.	O	0.999	O

Table 1: L0 Network Example Output

- Word, Suffix and Caps as in the L0 network.
- L0: 19x5, indexed by the L0 network identified concept.
- PredWords[5], 130Kx50: The word embeddings of the word and surrounding 2 words associated with the source predicate concept.
- PredL0[5], 19x10: The L0 embedding of the word and surrounding 2 words associated with the source predicate concept.
- regionMark: 21x5, indexed by the distance in words between the word and the word associated with the source predicate concept.

3.2.3 Non-Predicate Relations BDLSTM Network (Nargs)

The Nargs Network uses features similar to the Args network, is run once for each concept, and produces a matrix P_{nargs} which defines the probability of a type of relation from an L0 concept to any other L0 concept, prior to the identification of any relations.[5] The matrix has dimensions a_n by c, where a_n is the number of non-arg relations to be identified, and c is the total number of concepts.

3.2.4 Attributes BDLSTM Network (Attr)

The Attr Network determines a primary attribute for each concept, if any.[6] The attributes (op words) associated with named entities are determined directly during L0 concept identification. This network is simplified to detect only one attribute (there could be many) per concept, and only computes probabilities for the most common attributes: TOP, polarity, and quant.

3.2.5 Named Category BDLSTM Network (NCat)

The NCat Network uses features similar to the L0 Network, along with the suggested categories (up to eight) from the Wikifier, and produces probabilities for each of 68 :instance roles, or categories, for named entities identified in the training set AMR's.

- Word, Suffix and Caps as in the L0 network.
- WikiCat[8]: 108 x 5, indexed by suggested categories from the Wikifier.

[5]Degree, mod, or quant are examples of narg relations in Figure 1.

[6](TOP: plan-01) and (op1: france) are attribute examples shown in Figure 1.

Semeval Task 8 Dataset	Smatch F1
Test	66.1%
Evaluation	56.0%

Table 2: Smatch F1 results for Test and Eval Datasets

3.3 Wikifier

Named entities in AMR are annotated with a canonical form, using Wikipedia as the standard (see *France* in Figure 1). A :wiki role, or link, should be provided if an appropriate wikipedia page exists, root category (or top-level :instance role) should also be provided. To determine these fields, prior to running the L0 network, we run the sentences through the University of Illinois Wikifier (Ratinov et al., 2011; Cheng and Roth, 2013) which provides a wiki link and a list of possible categories. We insert the link directly as a :wiki role, and use the possible categories as feature inputs to the NCat Network.

3.4 Relation Resolution

The generated P_{args} and P_{nargs} for each L0 identified concept are processed to determine the most likely relation connections, using the constraints:

1. AMR's are single component graphs without cycles.
2. AMR's are simple directed graphs, a max of one relation between concepts is allowed.
3. Outgoing predicate relations are limited to one of each kind (i.e. can't have two ARG0's)

We apply a greedy algorithm which repeatedly selects the most probable edge from P_{args} and P_{nargs}, then adjusts P_{args} and P_{nargs} based on the constraints (hard decisions change the probabilities), until all edge probabilities are below a threshold. From then on, only the most probable edges which span subgraphs are chosen, until the graph contains a single component.

4 Results

Semeval task 8 provides aligned, split datasets. Our Smatch F1 result for the test dataset was 66.1%, and 56.0% for the eval dataset (Table 2). Reportedly, the eval dataset is more challenging than the provided test dataset. The mean of all task 8 results for the eval dataset is 55% with a standard deviation of 6%, more detail is not yet available.

5 Conclusion

In this paper, we have described our submission to the AMR Parsing task for SemEval-2016. Our parser does not make use of a syntactic pre-parse, and avoids the use of heavily engineered features. Future work will include expanding the identification of concepts and exploring the use of more sophisticated alignments and word embeddings.

References

Yoav Artzi, Kenton Lee, and Luke Zettlemoyer. 2015. Broad-coverage ccg semantic parsing with amr. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štepánek, Pavel Stranák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang*.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2012. Abstract meaning representation (amr) 1.0 specification. In *Parsing on Freebase from Question-Answer Pairs. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. Seattle: ACL*, pages 1533–1544.

X. Cheng and D. Roth. 2013. Relational inference for wikification. In *EMNLP*.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537.

Jeffrey Flanigan, Sam Thomson, Jaime G Carbonell, Chris Dyer, and Noah A Smith. 2014. A discriminative graph-based parser for the abstract meaning representation.

Felix A Gers, Douglas Eck, and Jürgen Schmidhuber. 2001. Applying lstm to time series predictable through time-window approaches. In *Artificial Neural NetworksICANN 2001*, pages 669–676. Springer.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey E. Hinton. 2013. Speech recognition with deep recurrent neural networks. *CoRR*, abs/1303.5778.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Xiaochang Peng, Linfeng Song, and Daniel Gildea. 2015. A synchronous hyperedge replacement grammar based approach for amr parsing. *CoNLL 2015*, page 32.

Michael Pust, Ulf Hermjakob, Kevin Knight, Daniel Marcu, and Jonathan May. 2015. Parsing english into abstract meaning representation using syntax-based machine translation. *Training*, 10:218–021.

L. Ratinov, D. Roth, D. Downey, and M. Anderson. 2011. Local and global algorithms for disambiguation to wikipedia. In *ACL*.

Jürgen Schmidhuber, F Gers, and Douglas Eck. 2002. Learning nonregular languages: A comparison of simple recurrent networks and lstm. *Neural Computation*, 14(9):2039–2041.

Lucy Vanderwende, Arul Menezes, and Chris Quirk. 2015. An amr parser for english, french, german, spanish and japanese and a new amr-annotated corpus. In *Proceedings of NAACL-HLT*, pages 26–30.

Chuan Wang, Nianwen Xue, Sameer Pradhan, and Sameer Pradhan. 2015. A transition-based algorithm for amr parsing. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 366–375.

Keenon Werling, Gabor Angeli, and Christopher Manning. 2015. Robust subgraph generation improves abstract meaning representation parsing. *arXiv preprint arXiv:1506.03139*.

Jie Zhou and Wei Xu. 2015. End-to-end learning of semantic role labeling using recurrent neural networks. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.