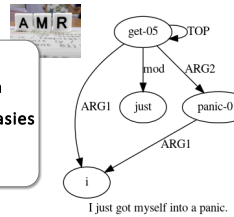
Improving Transition-based Semantic Parsing of AMR

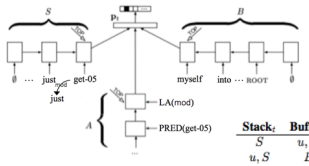
Austin Blodgett* PhD Student, in Collaboration with Miguel Bellesteros, Ramon Astudillo, Tahira Naseem, Young-Suk Lee
*Georgetown University

Transition Based AMR Parsing

Abstract Meaning Representation (AMR)

- AMR: "who is doing what to whom"
- sentence-level meaning representation
- abstract away from syntactic idiosyncrasies
- has been used to help NMT and QA
  (Song et. Al, 2019; Mitra et. al, 2016)





AMR parsing using Stack-LSTMs (Ballesteros and Al-Onaizan 2017)

Stack_t	Buffer_t	Action	Stack_{t+1}	Buffer_{t+1}	Graph
S	u, B	SHIFT	u, S	B	—
u, S	B	PRED(X)	n, S	B	—
u, S	B	REDUCE	S	B	—
u, v, S	B	MERGE	$(u, v), S$	B	—
u, S	B	ENTITY(l)	$(u : l), S$	B	—
u, S	B	DEPENDENT(r, d)	u, S	B	$u \xrightarrow{r} d$
u, v, S	B	RA(r)	u, v, S	B	$u \xrightarrow{r} v$
u, v, S	B	LA(r)	u, v, S	B	$u \xleftarrow{r} v$
u, v, S	B	UNSHIFT	u, S	v, B	—

Introduction

Transition-based methods were originally designed to parse syntax as dependency trees.

What makes parsing AMRs more difficult?

- **Many to many** relationship between nodes/tokens

- **Unaligned nodes** that don't correspond to any token

- **Re-entrencie**s: AMR nodes can have multiple parents

- **No Gold Actions**: We have to use an oracle to guess the correct action based on gold AMRs

Experiments

Experiment 1: add stop word information to prediction of relations

- Currently the parser calls REDUCE on unaligned words ('the', 'of', 'at', 'on', etc.)
- Some of that information would be useful for predicting edges ('in' marks :location, 'at' marks :time, etc.)
- Proposal
  - Add a stack-LSTM stopwords
  - When reduce is called on a token with no edges, add it to stopwords
  - use the output of stopwords to predict edge labels

Experiment 2: add DUPLICATE action, inspired by (Zhang et al. 2019)

- re-entrencies are a difficult part of AMR parsing (no parallel in dependency parsing)
- curently, we rely on UNSHIFT which relies on knowledge of the entire stack, not individual tokens
- (Zhang et al. 2019) handles re-entrencies by duplicating tokens and then predicting edges
- Proposal:
  - Add a Stack-LSTM "Latent" for storing nodes that didn't exist in the original token sequence (including diplicates)
  - The purpose of Latent is that we can introduce nodes independently of the state of the buffer.
  - Add actions DUPLICATE and INTRODUCE to add/shift a node from Latent.

Experiment 3: Augment tokens with Unaligned nodes

- many nodes in the gold AMRs are unaligned
- currently our parser can't predict (avg SMATCH cost -6.2)
- Proposal:
  - Augment tokens with tags <extra-node>
  - align <extra-node> with an unaligned node in gold AMR
  - at training time, parser should learn to add predicates and edges to <extra-node> as if it was aligned

Experiment 4: add composition function to encode dependents

- Currently the parser uses composition functions for each action to encode partial graph representations of the AMR during parsing.
- LA and RA encode the *head* of each added edge with info about it's dependent
- We can also encode the *dependent* with info about it's head
- Possibly useful for predicting re-entrencies which a difficulty for AMR parsing.

Preliminary Results

epoch	base	stopwords	duplicate	unaligned	dep embeddings
0	0.651	**0.667**	0.643	0.651	**0.657**
1	0.673	**0.676**	0.663	0.661	0.669
2	0.673	**0.678**	0.673	0.651	0.672
3	0.674	**0.678**	0.661	0.671	**0.676**
4	0.675	**0.682**	0.670	0.671	**0.679**
5	0.677	**0.683**	0.671	0.674	0.675
10	0.680	**0.683**	0.668		
15	0.683		0.672		

Discussion

Experiment 1 (Saving Stopwords) gets the best results revealing that stop word carry important information for AMR semantic roles.

Experiments 2 and 3 reduce performance by a small fraction. We will run error analysis to see why this is and if it can be improved.

For Experiment 4 results are mixed and may improve with more training.

References

Ballesteros, Miguel, and Yaser Al-Onaizan. "**Amr parsing using stack-lstms**." (2017).

Naseem, Tahira, Abhishek Shah, Hui Wan, Radu Florian, Salim Roukos, and Miguel Ballesteros. "**Rewarding Smatch: Transition-Based AMR Parsing with Reinforcement Learning**." (2019).

Zhang, Sheng, Xutai Ma, Kevin Duh, and Benjamin Van Durme. "**AMR Parsing as Sequence-to-Graph Transduction**." (2019).

email@university.edu