Learning from text: A personal experience

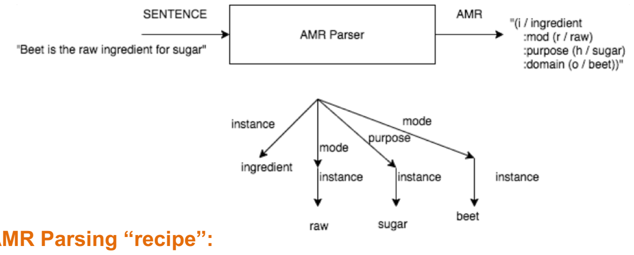
Camelia LEMNARU
Knowledge Engineering Group
Department of Computer Science, *Technical University of Cluj-Napoca*, Romania
Camelia.Lemnaru@cs.utcluj.ro

Motivation

Semantic parsing is one of the most difficult language understanding tasks, but important in many real-world applications (conversational systems, translation systems, etc). **AMR parsing** is a kind of *deep semantic parsing*, which attempts to capture the meaning of a sentence and transfer it to a formal representation language (AMR) [1]. It can be jointly interpreted as a sequence learning problem and a structured learning problem. Considering the recent advancements brought about by recurrent models in these problems, and drawing inspirations from existing solutions to dependency parsing, we explore a **transition based approach using a Stack Long-Short Term Memory** model for solving this problem, in [2] and [5].

AMR Parsing – From text to meaning

An AMR is a DAG: nodes are concepts (entities or actions), edges are the relations between concepts.
An AMR graph captures the meaning along the line **WHO** did **WHAT** to **WHOM**?



AMR Parsing "recipe":

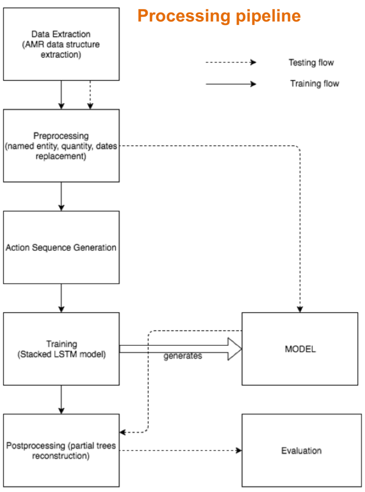
- **Identify concepts** → beet, raw, sugar, ingredient
- **Identify relation types** → :mod, :purpose, :domain
- **Predict structure** → AMR graph

Approach – Transition based LSTM parser [2]

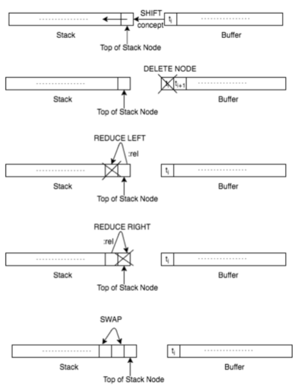
Processing pipeline: inspired from **transition-based dependency parsing**
We attempt to learn the action sequence that needs to be applied on the sentence in order to obtain the AMR graph; additional pre-processing and post-processing, to handle: **named** and **date entities**, **quantities.**

Action set: extension of the arc-standard dependency parser [3], with several actions being altered to support action labels - necessary in the AMR graph, and new actions added.

Action sequence (oracle) generation: deterministic algorithm



Action set



"Beet is the raw ingredient for sugar."
e.0 e.1 e.2 e.3 e.4 e.5 e.6

Shift_beet, **DeleteNode**, **DeleteNode**, **Shift**_raw, **Shift**_ingredient, **ReduceLeft**_:mod, **ReduceLeft**_:domain, **DeleteNode**, **Shift**_sugar, **ReduceRight**_:purpose

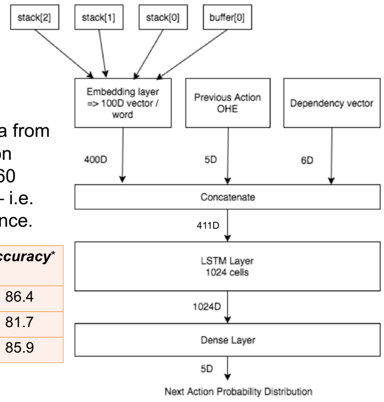
(i / ingredient~e.4
:mod (r / raw~e.3)
:purpose~e.5 (h / sugar~e.6)
:domain~e.1 (o / beet~e.0))

Action set (definitions)

Action(transition)	Current state	Result state
$Reduce - left_{l_r}$	$([\sigma\|x,y],[\beta],A)$	$([\sigma\|y],[\beta],A\cup(y,x,l_r))$
$Reduce - right_{l_r}$	$([\sigma\|x,y],[\beta],A)$	$([\sigma\|x],[\beta],A\cup(x,y,l_r))$
$Shift_{l_c}$	$([\sigma],[i\|\beta],A)$	$([\sigma\|x],[\beta],A)$
$Delete$	$([\sigma],[i\|\beta],A)$	$([\sigma],[\beta],A)$
$Swap$	$([\sigma\|x,y,z],[\beta],A)$	$([\sigma\|y,x,z],[\beta],A)$

Action sequence prediction model [2]

Approach We draw inspiration from the **Stack Long Short Term Memory** model proposed in [4] for dependency parsing.



Action prediction results: data from Task 9 @ SemEval 2017 (discussion forums, formal reports). Out of 39260 instances, 8856 **trainable** (~24%) – i.e. we could generate an action sequence.

Dataset	Smatch F-Score	Smatch Mean	Accuracy*
Reports	57.8	63.9	86.4
Online Forums	55.4	62.2	81.7
Full set	58.2	66.4	85.9

*Accuracy for predicting the correct action

AMR Parsing – Addressing challenges [5]

Improving action sequence generation: better handle non-projectivity.

- **enlarge action set:**

Proposed additional actions (definitions)

Action(transition)	Current state	Result state
$Swap_2$	$([\sigma\|x,y,z,t],B,A)$	$([\sigma\|z,y,x,t],B,A)$
$Swap_3$	$([\sigma\|x,y,z,t,v],B,A)$	$([\sigma\|t,y,z,x,v],B,A)$
$Rotate$	$([x\|\sigma\|y,z],B,A)$	$([y\|\sigma\|x,t],B,A)$
$Break - token_{l_{c_1}l_{c_2}}$	$([\sigma],[i\|B],A)$	$([y\|\sigma\|x,y],B,A)$

- **alter oracle generation algorithm:** explore **local backtracking** searches, as well as an "**informed swap**" strategy (choose *Swap* whenever a reduce operation becomes possible due to the swap)

Re-entrance (co-reference): initial parsing approach processes only trees.

implicit	*explicit*
"The boy wants to go."	"Bob likes himself."
(w / want-01	(l / like-01
:ARG0 (**b** / boy)	:ARG0 (**b** person :name „Bob")
:ARG1 (g / go-01)	:ARG1 (**b**))
:ARG0 (**b**))	

- **detect, remove and restore** (*under development*)
  (+) Small modifications to current processing flow
  (-) Implicit co-reference poorly detected by current detection systems

Conclusions

Despite being relatively new, AMR parsing has drawn the attention of important research groups in NLP. The great majority of solutions utilize transition based strategies, which makes them sensitive to issues such as **non-projectivity** and **co-reference**.

Our proposed solution exploits the principles of transition based dependency parsing, and we are currently focusing on improving the action set and the oracle generation strategy, as well as handling AMR reentrant concepts (co-reference).

References

[1] L. Banarescu, C. Bonial, S. Cai, M. Georgescu, K. Grif- fitt, U. Hermjakob, K. Knight, M. Palmer, and N. Schnei- der, "Abstract meaning representation for sembanking," in *the 7th Linguistic Annotation Workshop and Interop- erability with Discourse*, 2013, pp. 178–186.

[2] S. Cimpian, A. Lazar, F. Macicasan, C. Lemnaru, "A transition-based approach for AMR parsing using LSTM networks", *2017 IEEE 12th International Conference on Intelligent Computer Communication and Processing (ICCP)*, Cluj-Napoca, pp. 103-110, ISBN: 978-1-5386-3369-4, 2017

[3] Nivre, J. "An efficient algorithm for projective dependency parsing", In Proceedings of the 8th International Workshop on Parsing Technologies (IWPT), 2003

[4] C. Dyer, M. Ballesteros, W. Ling, A. Matthews, & N. A. Smith, "Transition-based dependency parsing with stack long short-term memory", *In Proc. ACL*, (2015)

[5] R. Pop, A. Dregan, F. Macicasan, C. Lemnaru and R. Potolea, " Enhancements on a Transition-based Approach for AMR Parsing using LSTM Networks", under review at *2018 IEEE 13th International Conference on Intelligent Computer Communication and Processing (ICCP)*

Knowledge Engineering Group
https://keg.utcluj.ro

The code is available at:
https://github.com/silvianacmp/AMR_lic

TECHNICAL UNIVERSITY OF CLUJ-NAPOCA ROMANIA