Applying Imitation Learning on Semantic Parsing

Semantic parsing

([Goodman et al. 2016 (http://aclweb.org/anthology/P16-1001)](http://aclweb.org/anthology/P16-1001))

Semantic parsers map natural language to meaning representations

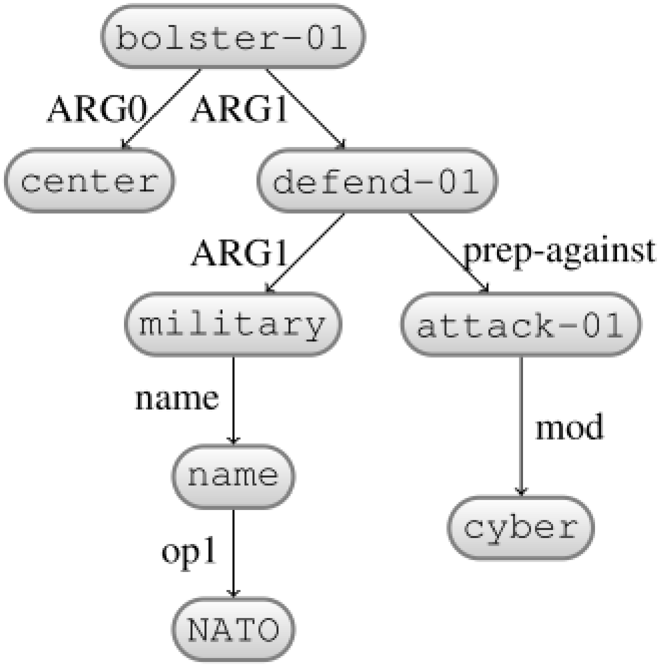
- Need to abstract over syntactic phenomena, resolve anaphora, eliminate ambiguousness in word senses
- Essentially the inverted task of natural language generation

Also known as:

- natural language understanding, natural language database interfaces, semantic role labeling, question answering on databases

Abstract meaning representation

([Banarescu et al. 2013 (http://www.aclweb.org/anthology/W13-2322)](http://www.aclweb.org/anthology/W13-2322))



A meaning representation formalism that utilizes a gr aph to represent relationships between concepts.

- Structure similar to dependency parses.
- But abstracts away from function words, and inflection details of words.
- Due to its structure, tr ansition-based approaches are common.

How can Imitation Learning help with that?

Similarly to dependency parsing, greedy encoding suffers from error propagation.

Imitation Learning addresses error propagation, by considering the interaction among the transition being considered and transitions to be predicted later in the sentence.

- Explores the search space, but avoids enumerating all possible outputs.
- Also learns how to recover from errors.

Transition system?

We consider a dependency graph (tree) as input.

- Dependency graphs are derived from the sentences.
- There is a lot more training data avalaible for dependecy parsing, than exists for AMR parsing.

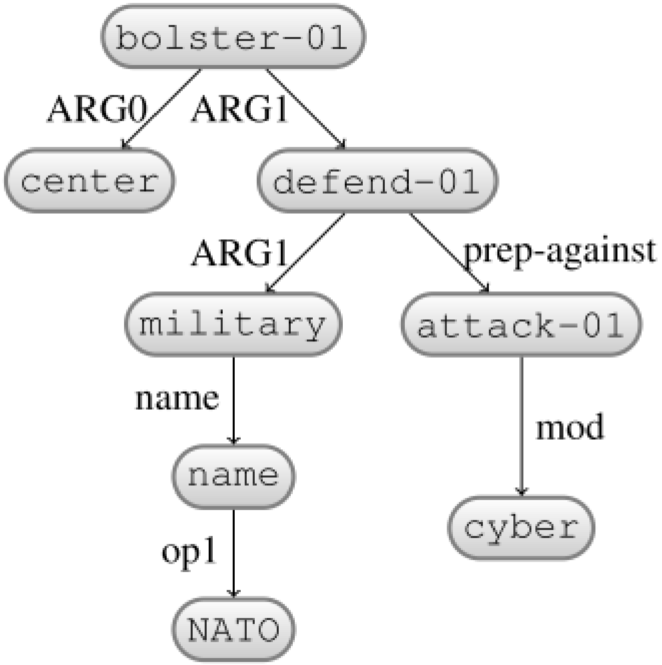
Transition actions transform the dependency graph into an AMR graph.

- In intermediate stages, some nodes are labeled with words from the sentence, and others with AMR concepts.

Actions

Action Name	Param.	Pre-conditions	Outcome of action
NextEdge	l_r	β non-empty	Set label of edge (σ_0, β_0) to l_r. Pop β_0.
NextNode	l_c	β empty	Set concept of node σ_0 to l_c. Pop σ_0, and initialise β.
Swap		β non-empty	Make β_0 parent of σ_0 (reverse edge) and its sub-graph. Pop β_0 and insert β_0 as σ_1.
ReplaceHead		β non-empty	Pop σ_0 and delete it from the graph. Parents of σ_0 become parents of β_0. Other children of σ_0 become children of β_0. Insert β_0 at the head of σ and re-initialise β.
Reattach	κ	β non-empty	Pop β_0 and delete edge (σ_0, β_0). Attach β_0 as a child of κ. If κ has already been popped from σ then re-insert it as σ_1.
DeleteNode		β empty; leaf σ_0	Pop σ_0 and delete it from the graph.
Insert	l_c		Insert a new node δ with AMR concept l_c as the parent of σ_0, and insert δ into σ.
InsertBelow			Insert a new node δ with AMR concept l_c as a child of σ_0.

Transition-based AMR parsing in action!



Task loss?

Smatch (Cai and Knight, 2013 (http://amr.isi.edu/smatch-13.pdf))

- F_1-Score between predicted and gold-target AMR gr aphs.
- Computationally expensive for every rollout.

Expert policy?

Best reachable state is explored via roll-outs, with naive Smatch used as a loss function.

- Skips combinatorial mapping of nodes between predicted and target gr aphs.
- Also, to encourage short trajectories, a length penalty is applied.

V-DAgger

Variant of DAgger proposed by Vlachos and Clark (2014) (http://www.aclweb.org/anthology/Q14-1042)

- Employs roll-outs, with the same policy used for both roll-ins and roll-outs.

Imitation Learning challenges

Incredible number of possible actions at each time-step.

- In the order of 10^3 to 10^4.
- Exploring all alternative actions at each time-step can be very time-consuming.

Incredible length of the action sequences.

- In the range of 50-200 actions.
- Especially challenging when combined with the large number of possible actions.

Targeted exploration

There is no reason to explore alternative actions when:

- Expert and learned policy agree on the correct action, **and**
- no alternative action is scored highly.

The algorithm limits the exploration to the expert action and learned policy actions whose scores is within a threashold τ from the best scored one.

- In first epoch, where there is no learned policy, we randomly explore a number of actions.

Other cases of partial exploration

SCB-LOLS and AggreVaTe both use partial exploration.

- They select which time-step they apply it at random.
- They select which actions they explore at random.

Targeted exploration focuses on the actions for which the leaned policy is least certain, or disagrees with the expert.

Issues of step-level stochasticity

v-DAgger and SEARN employ step-level stochasticity during their roll-outs.

- i.e. each step during roll-out can be performed by either the learned or expert policy.
- In other words, the same training example may have very different roll-outs when reexamined.
- This results in high variance in the reward signal, and hinders effective learning.

Noise reduction

a-bound by Khardon and Wachman (2007) (http://www.jmlr.org/papers/volume8/khardon07a/khardon07a.pdf)

- Exclude a training example from subsequent training if it has been already misclassified a times during training.

Alternatively, we could use LOLS

- Rollouts are performed consistently with the same policy .
- Can hurt training times when moving from exclusive expert to exclusive learned policy, due to large length of action sequences.

Focused costing

Introduced by [Vlachos and Craven (2011) (http://www.aclweb.org/anthology/W/W11/W11-0307.pdf)](http://www.aclweb.org/anthology/W/W11/W11-0307.pdf)

- Instead of using learned policy for β% of the rollout steps,
- use it for the first β steps and reverty to the expert policy for the rest.

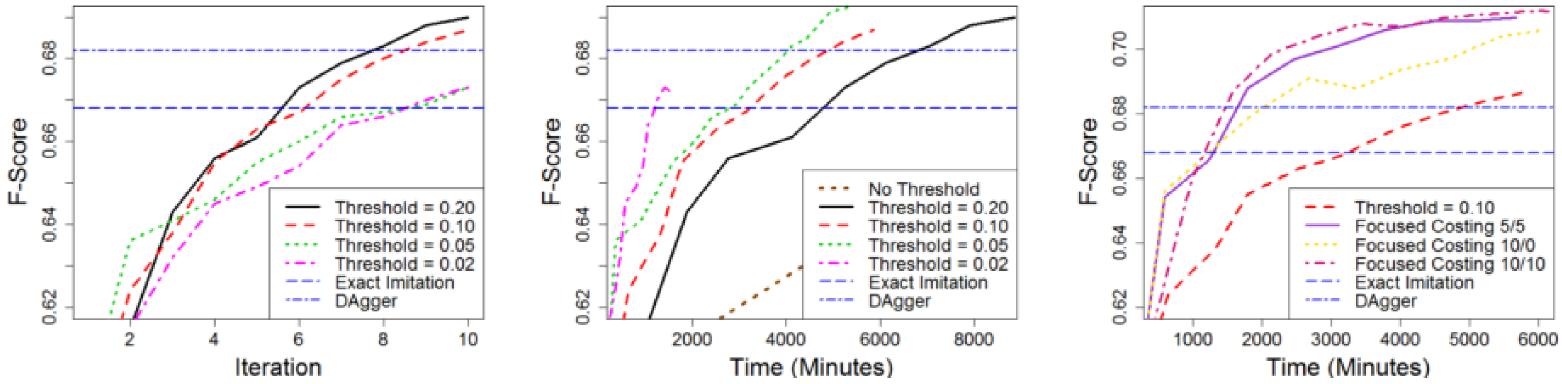
This keeps roughly the same computational cost, while focusing the effect of the explored action to the immediate actions that follow.

- Reduces noise, the mistakes the learned policy may make on distant actions are considered irrelevant.
- We can increase β with each epoch, to move away from the expert.

DAgger with a-bound

	Exact Imitation			Imitation Learning				
Experiment	No α	α=1	α-Gain	No α	α=1	IL Gain (α)	IL Gain (No α)	Total Gain
AROW, C=10	65.5	66.8	1.3	65.5	67.4	0.6	0.0	1.9
AROW, C=100	66.4	66.6	0.2	66.4	67.7	1.1	0.0	1.3
AROW, C=1000	66.4	67.0	0.6	66.5	68.2	1.2	0.1	1.8
PA, C=100	66.7	66.5	-0.2	67.2	68.7	2.2	0.5	2.0
Perceptron	65.5	65.3	-0.2	66.6	68.6	3.3	1.1	3.1

Targeted exploration and focused costing results



Comparison with previous work

Authors	Algorithmic Approach	R	P	F
Flanigan et al. (2014)	Concept identification with semi-markov model followed by optimisation of constrained graph that contains all of these.	0.52	0.66	0.58
Werling et al. (2015)	As Flanigan et al. (2014), with enhanced concept identification	0.59	0.66	0.62
Wang et al. (2015b)	Single stage using transition-based parsing algorithm	0.62	0.64	0.63
Pust et al. (2015)	Single stage System-Based Machine Translation	-	-	0.66
Peng et al. (2015)	Hyperedge replacement grammar	0.57	0.59	0.58
Artzi et al. (2015)	Combinatory Categorial Grammar induction	0.66	0.67	0.66
Wang et al. (2015a)	Extensions to action space and features in Wang et al. (2015b)	0.69	0.71	0.70
This work	Imitation Learning with transition-based parsing	0.68	0.73	0.70

Comparison with previous work

	Validation F-Score			Test F-Score	
Dataset	EI	D	V-D	V-D	Rao et al
proxy	0.670	0.686	0.704	0.70	0.61
dfa	0.495	0.532	0.546	0.50	0.44
bolt	0.456	0.468	0.524	0.52	0.46
xinhua	0.598	0.623	0.683	0.62	0.52
lpp	0.540	0.546	0.564	0.55	0.52