Learning to Search Better Than Your Teacher

[Chang et al., 2015]

Leo Born

Hauptseminar: Imitation Learning
Artem Sokolov
Department of Compuational Linguistics
Ruprecht-Karls-Universität Heidelberg

18.10.2018

1. Motivation
2. Locally Optimal Learning to Search (LOLS)
3. Performance in NLP Contexts
   - POS tagging
   - Dependency parsing
   - Timeline summarization
4. Conclusion

Contents

Motivation

- Learning to Search (L2S) is nice when the reference policy is good

Motivation

- Learning to Search (L2S) is nice when the reference policy is good
- But what happens when the refernce policy is sub-optimal? Can't we improve upon it?

Motivation

- Learning to Search (L2S) is nice when the reference policy is good
- But what happens when the refernce policy is sub-optimal? Can't we improve upon it?
  - "Yes, we can" [Obama, 2008]

Motivation

- Learning to Search (L2S) is nice when the reference policy is good
- But what happens when the refernce policy is sub-optimal? Can't we improve upon it?
    - "Yes, we can" [Obama, 2008]
- Locally Optimal L2S as one approach to be better than the teacher

Contents

Locally Optimal Learning to Search (LOLS)

Learning to Search Better Than Your Teacher [Chang et al., 2015]

- Roll-in with learned policy, mixture policy for roll-out
- If in one-step deviation there is a better solution than the expert's, use this
- Guarantees locally optimal policy
- Outperforms SEARN [Daumé III et al., 2009] on POS tagging and dependency parsing tasks
- Useful for other NLP tasks as well

Algorithm

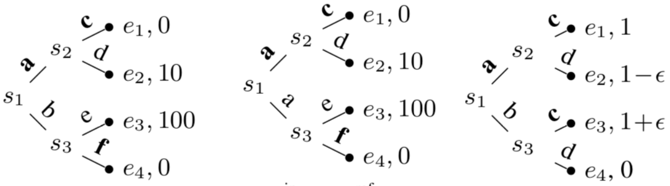
Algorithm 1 Locally Optimal Learning to Search (LOLS)

Require: Dataset $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{N}$ drawn from $\mathcal{D}$ and $\beta \geq 0$: a mixture parameter for roll-out.

Initialize a policy π_0.
for all $i \in \{1, 2, \ldots, N\}$ (loop over each instance) **do**
    Generate a reference policy π^{ref} based on $\boldsymbol{y}_i$.
    Initialize $\Gamma = \emptyset$.
    for all $t \in \{0, 1, 2, \ldots, T-1\}$ **do**
        Roll-in by executing $\pi_i^{\text{in}} = \hat{\pi}_i$ for t rounds and reach s_t.
        for all $a \in A(s_t)$ **do**
            Let $\pi_i^{\text{out}} = \pi^{\text{ref}}$ with probability β, otherwise $\hat{\pi}_i$.
            Evaluate cost $c_{i,t}(a)$ by rolling-out with π_i^{out} for $T - t - 1$ steps.
        end for
        Generate a feature vector $\Phi(\boldsymbol{x}_i, s_t)$.
        Set $\Gamma = \Gamma \cup \{\langle c_{i,t}, \Phi(\boldsymbol{x}_i, s_t)\rangle\}$.
    end for
    $\hat{\pi}_{i+1} \leftarrow \text{Train}(\hat{\pi}_i, \Gamma)$ (Update).
end for
Return the average policy across $\hat{\pi}_0, \hat{\pi}_1, \ldots \hat{\pi}_N$.

Analysis: What NOT to do

- Roll-in and roll-out with learned policy
- Roll-in with reference policy



(a) $\pi_i^{\text{in}} = \pi_i^{\text{out}} = \pi^{\text{ref}}$ (b) $\pi_i^{\text{in}} = \pi^{\text{ref}}$, representation constrained (c) $\pi_i^{\text{out}} = \pi^{\text{ref}}$

Figure: Effect of different roll-in/roll-out strategies [Chang et al., 2015].

Analysis: What to do instead

- Roll-in with learned policy, roll-out with mixture of learned and reference policy
- "a policy is locally optimal if changing any one decision it makes never improves its performance"

$$\delta_N = \frac{1}{NT}\sum_{i=1}^{N}\sum_{t=1}^{T}\mathbb{E}_{s\sim d^t_{\hat{\pi}_i}}\left[Q^{\pi_i^{\text{out}}}(s,\hat{\pi}_i) - \left(\beta \min_a Q^{\pi^{\text{ref}}}(s,a) + (1-\beta)\min_a Q^{\hat{\pi}_i}(s,a)\right)\right]$$

$$\beta(J(\bar{\pi}) - J(\pi^{ref})) + (1-\beta)\sum_{t=1}^{T}\left(J(\bar{\pi}) - \min_{\pi\in\Pi}\mathbb{E}_{s\sim d^t_{\bar{\pi}}}[Q^{\bar{\pi}}(s,\pi)]\right) \leq T\delta_N.$$

Analysis: What to do instead

- Minimizes a combination of regret to reference policy and regret to its own one-step deviations
- When reference is optimal, first term is non-negative; competes with one-step deviations in this case
- When reference is sub-optimal, first term can be negative -> learned policy has improved upon the reference policy

Analysis

roll-out → ↓ roll-in	**Reference**	**Mixture**	**Learned**
Reference	Inconsistent		
Learned	Not locally opt.	Good	RL

Figure: Effect of different roll-in/roll-out strategies [Chang et al., 2015].

Contents

POS tagging

- Train on 38k sentences, test on 11k from Penn Treebank
- Best SEARN performance: 94.88 accuracy
- Performance of LOLS:

roll-out → ↓ roll-in	**Reference**	**Mixture**	**Learned**
Reference is optimal			
Reference	95.58	94.12	94.10
Learned	**95.61**	94.13	94.10

Dependency parsing

- Shift-reduce parser with three actions
- Three reference policies
  - *Optimal*: "non-deterministic oracle" [Goldberg and Nivre, 2013]
  - *Sub-optimal*: action that leads to good end state when obvious (?), else arbitrary
  - *Bad*: Arbitrary action
- Performance of SEARN:
  - 84.0
  - 81.1
  - 63.4

Dependency parsing

- Performance of LOLS:

roll-out → ↓ roll-in	**Reference**	**Mixture**	**Learned**
Reference is optimal			
Reference	87.2	89.7	88.2
Learned	**90.7**	90.5	86.9
Reference is suboptimal			
Reference	83.3	87.2	81.6
Learned	87.1	**90.2**	86.8
Reference is bad			
Reference	68.7	65.4	66.7
Learned	75.8	**89.4**	87.5

Results

- Roll-in with reference is bad
- When reference is optimal, doing roll-outs with reference is a good idea
- When reference is sub-optimal or bad, mixture rollouts perform better
- LOLS also significantly outperforms SEARN on all tasks

Timeline summarization

Real-time web scale event summarization using sequential decision making [Kedzie et al., 2016]

- Implementation of a *streaming summarization system* on sentence level
- Uses LOLS
- Evaluation on TREC TS data with TREC metrics
- Best F1 score, particularly in combintation with sentence similarity filter

Timeline summarization

- Input: query (“eruption of Eyjafjallajökull’), category (“natural disaster”), sentence stream:
  - For every new sentence, decision whether to include it in summary or ignore it must be made
- Loss function is complement of Dice coefficient (F1 score)
- Greedy reference policy
- SGD classifier for updating $\tilde{\pi}$

Timeline summarization

- Data from TREC TS task [Aslam et al., 2013, Aslam et al., 2014]
  - Web-crawled between 10/2011 and 02/2013, 1.2 billion docs
  - Only considered news section: 7.6 million docs
  - 44 categorized events, 73.35 *nuggets* per event on average
- Stream size 100
- 5 events dev, 39 evaluated using leave-one-out

Timeline summarization

	unpenalized			latency-penalized			
	exp. gain	comp.	F_1	exp. gain	comp.	F_1	num. updates
APSAL	**0.119**^{c}	0.09	0.094	0.105	0.088	0.088	8.333
COS	0.075	0.176^{s}	0.099	0.095	0.236^{s}	0.128^{s}	145.615^{s,f}
LS	0.097	**0.207**^{s,f}	0.112	0.136^{c}	**0.306**^{s,c,f}	0.162^{s}	89.872^{s,f}
LSCOS	0.115^{c,l}	0.189^{s}	**0.127**^{s,c,l}	**0.162**^{s,c,l}	0.276^{s}	**0.184**^{s,c,l}	29.231^{s,c}

Figure: Average performance and number of updates (i.e. summary length) [Kedzie et al., 2016].

Contents

Conclusion

- SEARN: batch learning, LOLS: online learning
- During training, LOLS needs just last policy for roll-in
- **Distinguishing feature** is mixture roll-out policy
- Even when the reference is bad, LOLS improves upon it

Conclusion

- SEARN: batch learning, LOLS: online learning
- During training, LOLS needs just last policy for roll-in
- **Distinguishing feature** is mixture roll-out policy
- Even when the reference is bad, LOLS improves upon it
  - Or does it? Cf. [Sharaf and Daumé III, 2017]

Bibliography I

Aslam, J., Diaz, F., Ekstrand-Abueg, M., McCreadie, R., Pavlu, V., and Sakai, T. (2014).
TREC 2014 Temporal Summarization Track Overview.
In *Proceedings of the 23rd Text Retrieval Conference*, pages 1–15.

Aslam, J., Diaz, F., Ekstrand-Abueg, M., Pavlu, V., and Sakai, T. (2013).
TREC 2013 Temporal Summarization.
In *Proceedings of the 22nd Text Retrieval Conference*, pages 1–14.

Chang, K.-W., Krishnamurthy, A., Agarwal, A., Daumé III, H., and Langford, J. (2015).
Learning to Search Better Than Your Teacher.
In Blei, D. and Bach, F., editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pages 2058–2066, Lille, France.

Daumé III, H., Langford, J., and Marcu, D. (2009).
Search-based structured prediction.
Machine Learning, 75(3):297–325.

Bibliography II

Goldberg, Y. and Nivre, J. (2013).
Training Deterministic Parsers with Non-Deterministic Oracles.
Transactions of the Association for Computational Linguistics, 1:403–414.

Kedzie, C., Diaz, F., and McKeown, K. (2016).
Real-time web scale event summarization using sequential decision making.
In *IJCAI International Joint Conference on Artificial Intelligence*, pages 3754–3760.

Sharaf, A. and Daumé III, H. (2017).
Structured Prediction via Learning to Search under Bandit Feedback.
In *Proceedings ofthe 2nd Workshop on Structured Prediction for Natural Language Processing*, pages 17–26.