SEARNN: Training RNNs with Global-Local Losses

Rémi Leblond*, Jean-Baptiste Alayrac*, Anton Osokin, Simon Lacoste-Julien

INRIA/ENS, Paris, France — MILA/DIRO UdeM

* equal contribution

Summary

Some known problems when training RNNs:

RNNs are usually trained using Maximum Likelihood Estimation (MLE) i.e. Teacher Forcing. Issues with this surrogate:

- Different from the test loss
- All-or-Nothing loss
- Teacher forcing ⟶ Exposure bias

Can we build on existing approaches?

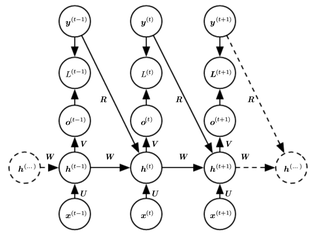
- Existing approaches use ideas from Reinforcement Learning to tackle these problems.
- Instead we use ideas from Structured Prediction: we revisit the Learning to Search literature, in particular the SEARN (or LOLS) approach [2, 7].

Contributions:

- Link between RNNs and Learning To Search
- Introduction of a new algorithm: SEARNN [3]
- Experiments on real structured prediction tasks with substantial improvement over MLE

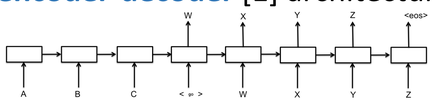
RNNs

Recurrent Neural Network:



$$h_t = f(h_{t-1}, y_{t-1})$$
$$s_t = \mathrm{proj}(h_t)$$
$$o_t = \mathrm{softmax}(s_t)$$

This work: **encoder-decoder** [1] architecture



Probabilistic interpretation:

$$o_t = P(Y_t|X, Y_1, ..., Y_{t-1}) \quad \prod_{t=1}^{T} o_t = P(Y_1, ..., Y_T|X)$$

⟶ **MLE:** $\max_\theta \sum_{i=1}^{n} \log(P_\theta(Y = Y_X|X))$

Known problems:

- $\neq$ **test loss** (edit distance, bleu score)
- **0/1 flavour** — Probability mass aggregated on one sample, bad for structured prediction
- **exposure bias** — Training scenario is different from what happens at test time

Learning to Search

Structured prediction

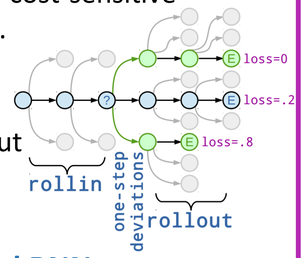
Learn a mapping f between inputs X and structured outputs Y made of interrelated parts often subject to constraints.

Learning To Search (L2S)

Reduces the structured problem down to cost-sensitive classification with theoretical guarantees.

How does it work?

A unique shared classifier makes predictions **one by one**, conditioned on the input and the previous tokens. This classifier is trained on an intermediate dataset.



Links between Learning to Search and RNNs

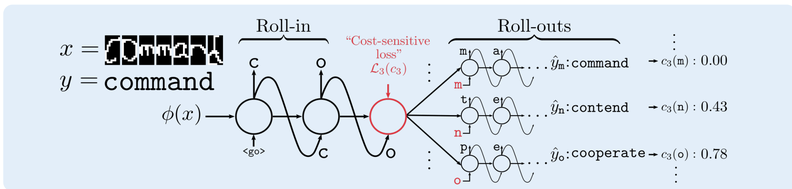
- Decomposition of structured tasks in sequential predictions conditioned on the past
- Unique shared classifier for all decisions using predecessors

SEARNN [3]

Overview:

Integrate roll-outs in the decoder to compute the cost of every possible action at every step.

Leverage these costs to enable better training losses.



Algorithm

1. Compute **costs** with **roll-in/outs**
2. Derive a **loss** from the **costs**
3. Use the loss to take a **gradient step**
4. Rinse and **repeat**

The devil in the details [7]

roll-out → / ↓ roll-in	Reference	Mixed	Learned
Reference	MLE (with TL)	Inconsistent	
Learned	Not locally opt.	Good	RL

Which cost-sensitive loss?
How can we scale?

Why is it better than MLE?

Makes direct use of the evaluation metric, and leverages **structured information** by comparing costs, contrary to MLE. Global-local losses, with **global** information at each **local** cell, vs. MLE (local information) and RL approaches (global loss).

Experiments

Full algorithm:

Dataset		A	T	Cost	MLE	roll-in / roll-out	LL learned mixed	LL reference learned	LL learned learned	LLCAS learned mixed	LLCAS reference learned	LLCAS learned learned
OCR		26	15	Hamming	2.8		1.9	2.5	**1.8**	1.9	2.4	1.9
CoNLL		22	70	norm. Hamming	4.2		**3.7**	6.1	5.6	5.8	5.3	5.1
Spelling	0.3	43	10	edit	19.6		17.8	19.5	17.9	**17.7**	19.6	**17.7**
	0.5				43.0		37.3	43.3	37.5	**37.1**	43.3	38.2

Scaling approach:

Dataset		MLE	LL uni.	LL stat.	LL pol.	LL bias.	LL top-k	sLL uni.	sLL stat.	sLL pol.	sLL bias.	sLL top-k	sLLCAS uni.	sLLCAS stat.	sLLCAS pol.	sLLCAS bias.	sLLCAS top-k
OCR		2.84	1.94	**1.50**	1.96	1.84	2.13	1.82	1.91	1.86	2.25	2.69	2.03	2.33	**1.50**	2.37	1.94
Spelling	0.3	19.6	17.7	17.7	17.7	17.7	17.8	18.8	20.5	17.5	17.7	18.4	18.8	20.2	17.7	17.7	18.2
	0.5	43.0	37.0	36.7	37.1	36.6	36.6	37.4	41.4	37.7	36.9	41.7	37.6	41.7	37.0	37.8	40.5

Takeaways:

- Significant improvements over MLE on **all 4 tasks**.
- The **harder** the task, the **bigger** the improvement.
- **Learned/mixed** is the best strategy for roll-in/out.
- The best performing losses (for now) are those **structurally close** to MLE.
- SeaRNN does not require **warm start**.
- The proposed sampling strategy works, maintaining improvements at a **fraction of the cost**.

Machine Translation (in progress): ISWLT 14 Ger/Eng

Depth	MLE[4]	MIXER[4]	MLE[5]	BSO[5]	MLE	SeaRNN
1	17.7	20.7	22.5	23.8	22.7	24.5

Depth	Actor-critic[6]*	MLE	SeaRNN (ref/mixed)
2	27.5	27.4	28.0

References

[1] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *NIPS*, 2014

[2] H. Daumé, III, J. Langford, and D. Marcu. Search-based structured prediction. *Machine Learning*, 2009

[3] R. Leblond, JB. Alayrac, A. Osokin and S. Lacoste-Julien. SEARNN: Training RNNs with global-local losses. *ArXiv preprint: 1706.04499*

[4] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba. Sequence level training with recurrent neural networks. In *ICLR*, 2016.

[5] S. Wiseman and A. M. Rush. Sequence-to-sequence learning as beam-search optimization. In *EMNLP*, 2016.

[6] D. Bahdanau, P. Brakel, K. Xu, A. Goyal, R. Lowe, J. Pineau, A. Courville, and Y. Bengio. An actor-critic algorithm for sequence prediction. In *ICLR*, 2017.

[7] K.-W. Chang, A. Krishnamurthy, A. Agarwal, H. Daumé, III, and J. Langford. Learning to search better than your teacher. In *ICML*, 2015.

Check out our project webpage for code/data!