Tree-structured decoding with doubly-recurrent neural networks

David Alvarez-Melis & Tommi S. Jaakkola
MIT CSAIL
{davidam,tommi}@csail.mit.edu

Summary

We propose a novel neural network architecture specifically tailored to tree-structured decoding, which:

- **maintains separate depth and width recurrent states** and combines them to obtain hidden states for every node in the tree.
- has a **mechanism to predict tree topology** *explicitly* (as opposed to *implicitly* by adding nodes with special tokens).

Our experiments show that this architecture

- is capable of recovering trees from encoded representations
- achieves state-of-the-art performance in a task consisting of mapping sentences to simple functional programs
- exhibits desirable invariance properties over sequential architectures

Background and Motivation

Why tree-structured?

- RNNs are a natural model for sequential data
- But many types of data are non-sequential, e.g.
  - natural language sentences or associated parse trees
  - programs, executable queries, etc
- Even sentences, which can be modeled as if they were linear sequences, have an underlying compositional process.

Previous work

Current neural architectures for non-sequential data usually assume:

a) the full tree structure is given (e.g. [5, 6]), or
b) at least the nodes are known (e.g. [1, 3])

In case (a), the network aggregates the node information in a manner that is coherent with a given tree structure. In case (b), generation is reduced to an attachment problem, i.e., sequentially deciding which pairs of nodes to join with an edge until a tree is formed.

Full ***decoding with structure*** is much less explored. Models so far remained relatively close to their sequential counterparts, e.g. using alternating RNNs coupled with external classifiers to predict branching [7] and introducing special tokens [2] to signal stopping.

Two downsides to using special tokens to control topology are:

(i) tree growth (up to $O(n)$ padding nodes in an n-node tree)
(ii) single stopping token selected competitively with other tokens

Challenges of tree-structured decoding

As opposed to seq-to-seq, **encoding and decoding are intrinsically asymmetrical**. Decoding requires multiple design choices:

- In which order should the tree be generated?
- What information should each node receive? Parent, sibling(s), etc.
- How to terminate generation?

Our approach

Grow tree root-to-leaves, encode parent-to-child and sibling-to-sibling information in separate recurrent states and model topological (stopping) decisions explicitly with a dedicated module

Doubly-Recurrent Neural Networks

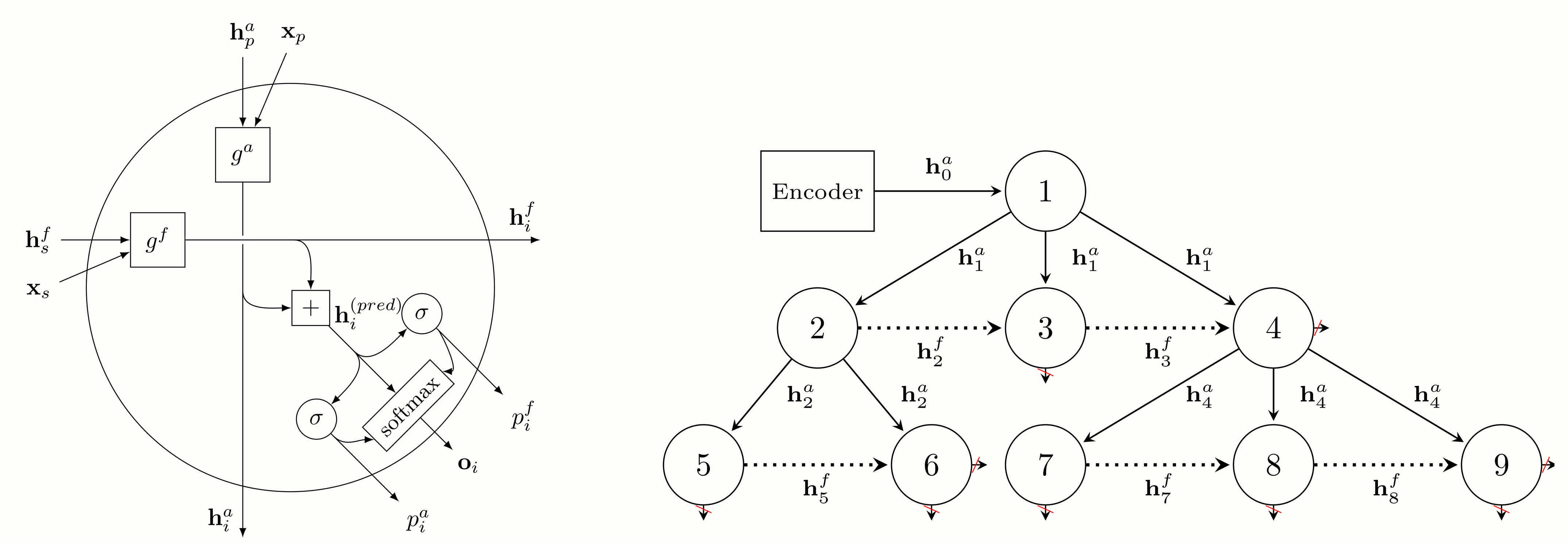

Figure 1: **Left**: A cell in the DRNN corresponding to node i with parent p and sibling s. **Right**: Structure-unrolled DRNN network in an encoder-decoder setting. Solid (dashed) lines indicate ancestral (fraternal) connections. Crossed arrows indicate production halted by the topology modules.

Cell recurrent states

$$\mathbf{h}_i^a = g^a(\mathbf{h}^a_{p(i)}, \mathbf{x}_{p(i)}) \quad (\textit{ancestral}\text{, depth state})$$
$$\mathbf{h}_i^f = g^f(\mathbf{h}^f_{s(i)}, \mathbf{x}_{s(i)}) \quad (\textit{fraternal}\text{, width state})$$

These are combined to obtain a *predictive hidden state*:

$$\mathbf{h}_i^{(pred)} = \tanh\left(\mathbf{U}^f\mathbf{h}_i^f + \mathbf{U}^a\mathbf{h}_i^a\right)$$

Training DRNNs

- With (reverse) **back-propagation through structure** (BPTS)
- Forward pass: top-down, on the structure-unrolled network
- Backward pass: bottom-up, feeding into every node gradients from children and sibling, computing internally gradients with respect to both topology and label prediction.
- Two loss terms: label and topology prediction

Topological Prediction

Instead of using stopping tokens, our model makes **topological decisions explicitly**, by computing:

$$p_i^a = \sigma(\mathbf{u}^a \cdot \mathbf{h}_i^{(pred)})$$

where $p_i^a \in [0, 1]$ is interpreted as the probability that node i has children. Analogously, the probability of stopping *fraternal* growth:

$$p_i^f = \sigma(\mathbf{u}^f \cdot \mathbf{h}_i^{(pred)})$$

Topological decisions $\alpha_i, \varphi_i \in \{0, 1\}$ are included for label prediction:

$$\mathbf{o}_i = \text{softmax}(\mathbf{W}\mathbf{h}_i^{(pred)} + \alpha_i\mathbf{v}^a + \varphi_i\mathbf{v}^f)$$

In practice, during training, we perform **teacher forcing**, replacing topological predictions p_i^a, p_i^f for true values (α_i, φ_i) after computing loss and before computing $\mathbf{o}_i$.

Experiments

Synthetic tree recovery

Task: Recovering tree structure from flattened (string) representations
Dataset: 5000 trees labeled with letters A-Z. We generate trees in a top-down fashion, conditioning every node's label and topology on the state of its ancestors and siblings.
Model: A DRNN as decoder, paired with a (sequential) RNN as encoder.
Evaluation: To give partial credit to correct substructures, we use an IR approach to evaluation, measuring F1-score of node and edge recovery.

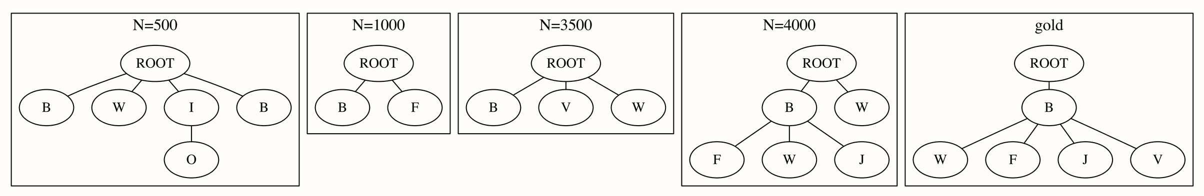

Figure 2: Trees generated from input string "ROOT B W F J V".

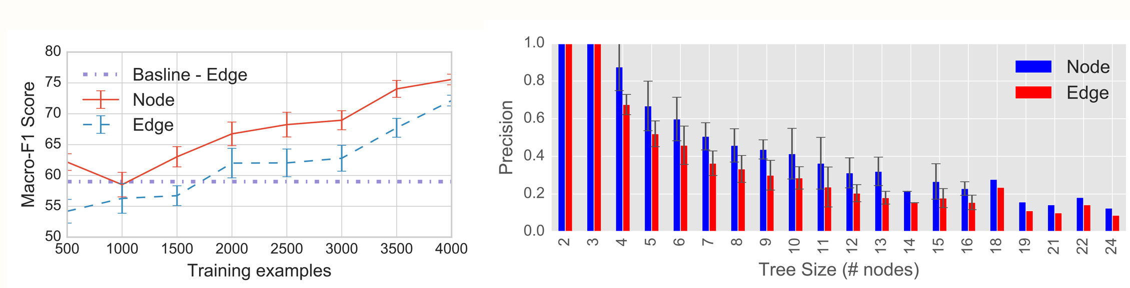

Figure 3: **Left**: Av. F1-Score vs. training data. **Right**: Node/edge precision vs. tree size.

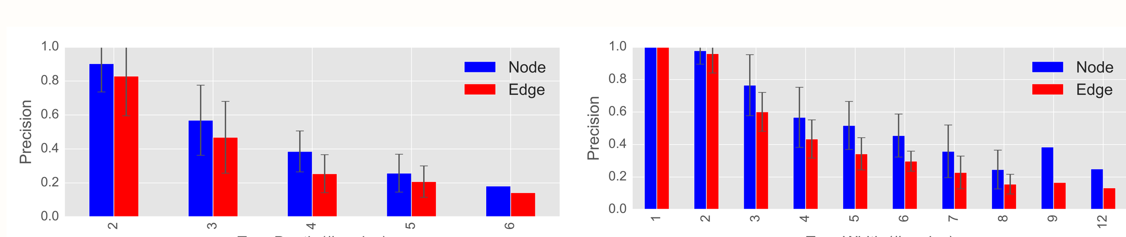

Figure 4: Node/edge precision vs. tree depth (left figure) and width (right).

Experiments (contd.)

Mapping sentences to functional programs

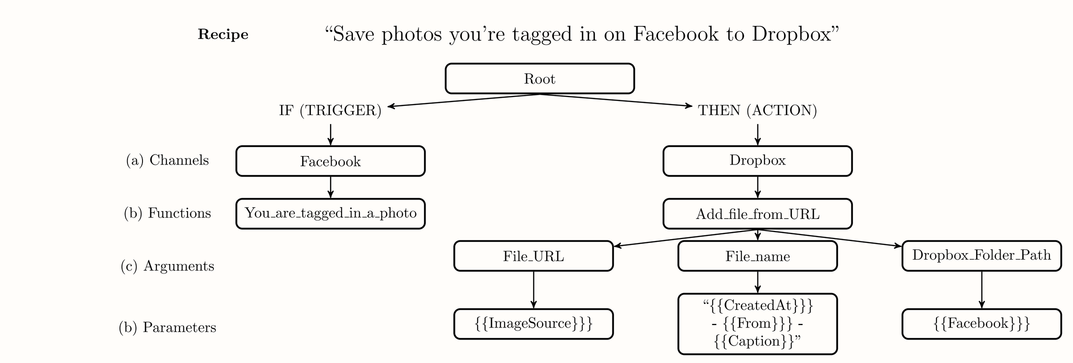

Figure 5: Example from the IFTTT dataset: description and program.

Dataset: IFTTT [4], consisting of simple programs *(recipes)* paired with descriptions of their purpose. User-generated and extremely noisy.
Model: RNN encoder and a DRNN decoder.
Evaluation: Accuracy in channel & function + F1-score of pred. tree

Method	Channel	+Func	F1
retrieval	36.8	25.4	49.0
classifier	64.8	47.2	56.5
posclass	67.2	50.4	57.7
Seq2Seq	68.8	50.5	60.3
Seq2Tree	69.6	51.4	60.4
Gru-Drnn	70.1	51.2	62.7
Lstm-Drnn	**74.9**	**54.3**	**65.2**

Method	Channel	+Func	F1
retrieval	43.3	32.3	56.2
classifier	79.3	66.2	65.0
posclass	81.4	71.0	66.5
Seq2Seq	87.8	75.2	73.7
Seq2Tree	89.7	**78.4**	74.2
Gru-Drnn	89.9	77.6	74.1
Lstm-Drnn	**90.1**	78.2	**77.4**

Table 1: Results on the IFTTT task. **Left**: non-English/unintelligible removed, **Right**: at least 3+ humans agree with gold (758 recipes).

Machine Translation

Can decoding with structure bring benefits to a task traditionally approached as a sequence-to-sequence problem, such as MT?
Training data: 50K En↔Fr sentences from the WMT14 dataset.
Models:

- DRNN: L/R children distinction, paired w/ LSTM encoder
- SEQ2SEQ: LSTM units, roughly same # of params as DRNN

Evaluation:

i) Invariance to structural perturbations in output, measuring Δ in LL
ii) Quality of translations at different *resolutions* (max target "size")

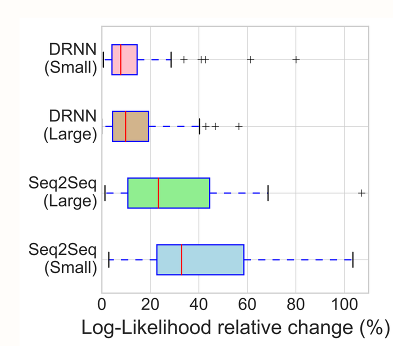

Figure 6: LLH change w/ target perturbation.

Source	" produit différentes réponses qui changent avec le temps selon nos expériences et nos relations "
Seq2Seq:	
$l = 1$	a
$l = 4$	with the different actions
$l = 8$	with the different actions who change with
Drnn:	
$d = 1$	answers
$d = 2$	different answers change
$d = 3$	product the different answers change .

Table 2: Translations at different resolutions.

Selected References

[1] D. Chen and C. D. Manning. A Fast and Accurate Dependency Parser using Neural Networks. *Proc. 2014 Conf. Empir. Methods Nat. Lang. Process.*, (i):740–750, 2014.
[2] L. Dong and M. Lapata. Language to Logical Form with Neural Attention. In *ACL*, pages 33–43, 2016.
[3] E. Kiperwasser and Y. Goldberg. Easy-First Dependency Parsing with Hierarchical Tree LSTMs. *TACL*, 2016.
[4] C. Quirk, R. Mooney, and M. Galley. Language to Code: Learning Semantic Parsers for If-This-Then-That Recipes. *ACL-IJCNLP*, (July):878–888, 2015.
[5] R. Socher, B. Huval, C. D. Manning, and A. Y. Ng. Semantic Compositionality through Recursive Matrix-Vector Spaces. In *EMNLP*, number Mv, pages 1201–1211, 2012.
[6] K. S. Tai, R. Socher, and C. D. Manning. Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks. In *ACL-IJCNLP*, pages 1556–1566, 2015.
[7] X. Zhang, L. Lu, and M. Lapata. Top-down Tree Long Short-Term Memory Networks. In *NAACL-HLT-2016*, pages 310–320, 2016.