Transformer; Syntax in SMT

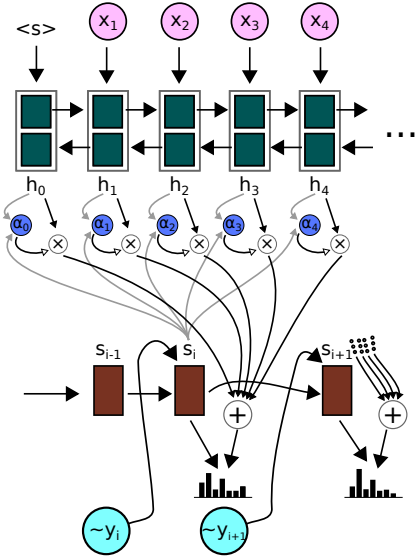
Ondřej Bojar

April 25, 2019

EUROPEAN UNION
European Structural and Investment Fund
Operational Programme Research,
Development and Education

Charles University
Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics

Reminder: Seq2seq with Attention



Attention – Formal Notation

Inputs:

decoder state s_i

encoder states $h_j = \left[\overrightarrow{h_j}; \overleftarrow{h_j}\right] \quad \forall i = 1 \dots T_x$

where $\overrightarrow{h_j} = \mathrm{RNN}_{\mathrm{enc}}(h_{j-1}, x_j) = \tanh(U_e \overrightarrow{h_{j-1}} + W_e E_e x_j + b_e)$

Attention energies: $e_{ij} = v_a^\top \tanh\left(W_a s_{i-1} + U_a h_j + b_a\right)$

Attention distribution: $\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}$

Context vector: $c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j$

Attention Mechanism in Equations (2)

Decoder state:

$$s_i = \tanh(U_d s_{i-1} + W_d E_d \hat{y}_{i-1} + C c_i + b_d)$$

Output projection:

$$t_i = \tanh\left(U_o s_i + W_o E_d \hat{y}_{i-1} + C_o c_i + b_o\right)$$

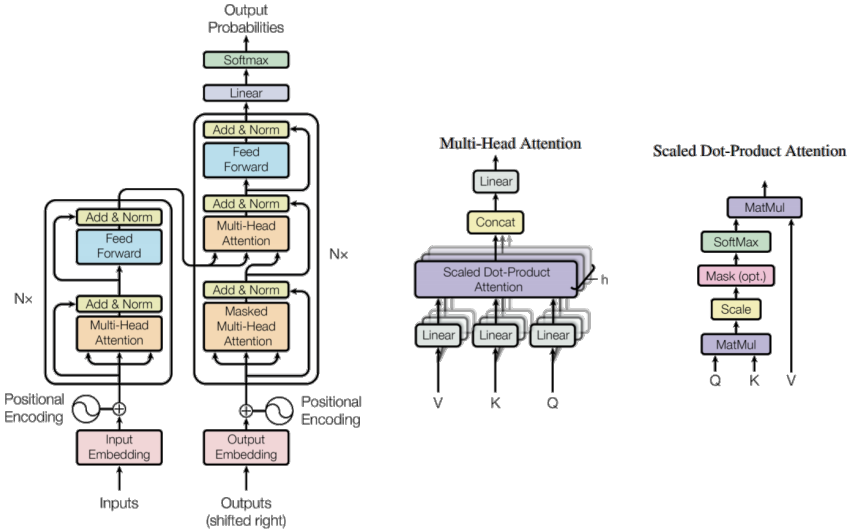
...context vector is mixed with the hidden state

Output distribution:

$$p\left(y_i = k \mid s_i, y_{i-1}, c_i\right) \propto \exp\left(W_o t_i\right)_k + b_k$$

Transformer

Attention is All You Need (Vaswani et al., 2017)



Transformer Detailed Walkthroughs

Transformer Illustrated:

- http://jalammar.github.io/illustrated-transformer/

Paper annotated with PyTorch code:

- http://nlp.seas.harvard.edu/2018/04/03/attention.html

Summary at Medium:

- https://medium.com/@adityathiruvengadam/transformer-architecture-attention-is-all-you-need-aecc

Self-Attention

See slides 46–53 (pages 53–60) by Jindřich Libovický.

Three uses of multi-head attention in Transformer

- Encoder-Decoder Attention:
  - Q: previous decoder layers; K = V: outputs of encoder

  ⇒ Decoder positions attend to all positions of the input.
- Encoder Self-Attention:
  - Q = K = V: outputs of the previous layer of the encoder

  ⇒ Encoder positions attend to all positions of previous layer.
- Decoder Self-Attention:
  - Q = K = V: outputs of the previous decoder layer.
  - Masking used to prevent depending on future outputs.

  ⇒ Decoder attends to all its previous outputs.

Syntax in NMT

Ways of Adding Syntax to NMT

- Source Syntax Attached to Tokens.
  - Factored NMT (+standard 'flat' word embeddings).
  - Graph-Convolutional Networks (word embs. reflect structure).
- Source Syntax Reflected in Encoder.
  - … Syntax reflected in network structure.
  - ! We are skipping TreeLSTM.
- Target Syntax through:
  - Interleave.
  - Multi-Decoder (sharing ends after encoder or attention or decoder).
- Somewhat suspicious results...

Why Syntax in NMT?

Motivation same as in SMT: long-distance dependencies.

- What city is the Taj Mahal in?
- Where is the Taj Mahal $\emptyset$?

The handling of the *in* depends on the *What*/*Where*.

- CCG tags for *is* differ $\Rightarrow$ dependency highlighted.

- $\text{What}_{(S[wq]/(S[q]/NP))/N}$ city $\text{is}_{(S[q]/P\ P)/NP}$ the Taj Mahal in?
- $\text{Where}_{S[wq]/(S[q]/NP)}$ $\text{is}_{(S[q]/NP)/NP}$ the Taj Mahal?

Linguistic Features in NMT in General

- Source word factors easy to incorporate:
  - Concatenate embeddings of the various factors.
  - POS tags, morph. features, source dependency labels help en↔de and en→ro (Sennrich and Haddow, 2016).
- Target word factors:
  - Interleave for morphology: (Tamchyna et al., 2017)

Src	there are a million different kinds of pizza .
Baseline (BPE)	existují miliony druhů piz@@ zy .
Interleave	VB3P existovat NNIP1 milion NNIP2 druh NNFS2 pizza Z: .

  - Interleave for syntax: (Nadejde et al., 2017)

Src BPE	Obama receives Net+ an+ yahu in the capital of USA
Tgt	NP Obama ((S[dcl]\NP)/PP)/NP receives NP Net+ an+ yahu PP/NP in NP

Nadejde et al. (2017)

Source-side

BPE:	Obama	receives	Net+	an+	yahu	in	the	capital	of	USA
IOB:	O	O	B	I	E	O	O	O	O	O
CCG:	NP	((S[dcl]\NP)/PP)/NP	NP	NP	NP	PP/NP	NP/N	N	(NP\NP)/NP	NP

Target-side

NP Obama ((S[dcl]\NP)/PP)/NP receives NP Net+ an+ yahu PP/NP in NP/N the N capital (NP\NP)/NP of NP USA

- Source side: Enrich tokens.
  - Concatenate the embeddings of the individual vocabularies.
  - Derive word embeddings from syntax.
- Target side:
  - Predict interleaved
    - Tamchyna et al. (2017) did this for morphology.
  - Have two decoders
    - Worked worse since the outputs differed in length.

Graph-Convolutional Networks

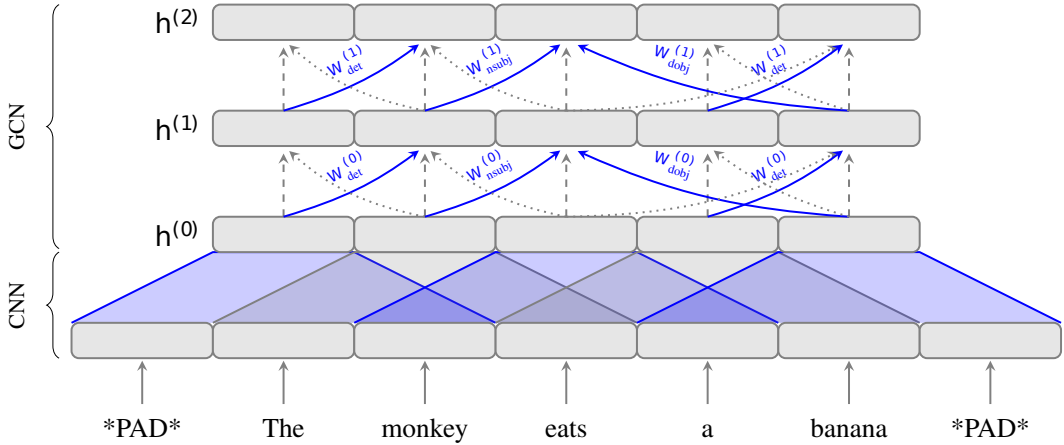
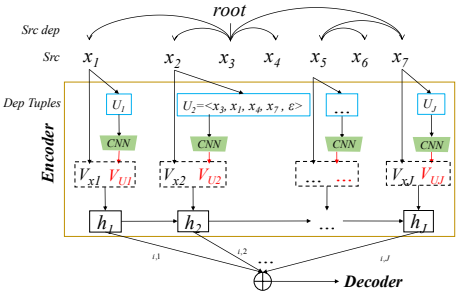


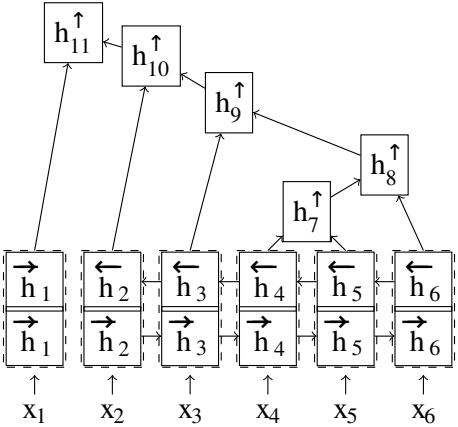
Figure 2: A 2-layer syntactic GCN on top of a convolutional encoder. Loop connections are depicted with dashed edges, syntactic ones with solid (dependents to heads) and dotted (heads to dependents) edges. Gates and some labels are omitted for clarity.

Chen et al. (2017b)



- CNN-derived embeddings of nodes' syntactic neighbourhood included: (parent, siblings).
- Two mechanisms:
  - Concatenated to standard embeddings.
  - Separate attention over these word-level annotiations
- +1.6 BLEU on Chinese-to-English.

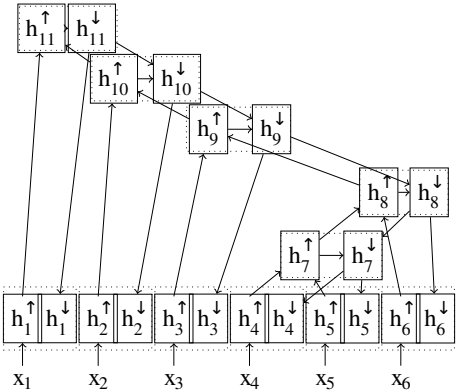
Chen et al. (2017a) (1/4)



Tree-GRU Encoder:

- Constituency syntax of the tree provides additional states.

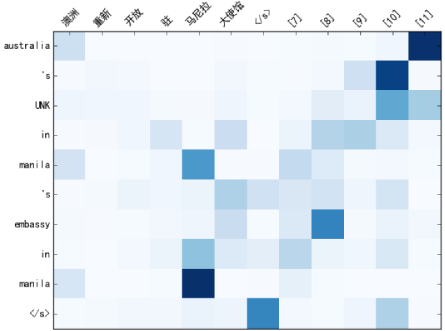
Chen et al. (2017a) (1/4)

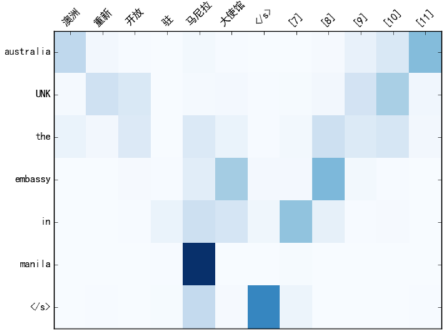


Bidirectional tree encoder.

- Can be seen as many RNNs running from each word up to the root and back to the word.

Chen et al. (2017a) (3/4)





Tree coverage model:

- Attention coverage depends on source syntax.
- Without it (left), output is repeated.

(Tree) Coverage in Attention

$$C_{j,i} = \mathrm{GRU}(C_{j-1,i}, \alpha_{j,i}, d_{j-1}, h_i) \tag{1}$$

$$e_{j,i} = v_a^T \tanh(W_a d_{j-1} + U_a h_i + V_a C_{j-1,i}) \tag{2}$$

$C_{j,i}$ coverage of src word i when producing tgt word j,
h_i encoder state at word i, d_j decoder state at word j,
$\alpha_{j,i}$ attention weight of src i when producing word j,
$e_{j,i}$ "energy" for tgt word j from src word i (i.e. α before softmax)

$$\begin{aligned} C_{j,i} = \mathrm{GRU}(&C_{j-1,i}, \alpha_{j,i}, d_{j-1}, h_i, \\ &C_{j-1,L(i)}, \alpha_{j,L(i)} \\ &C_{j-1,R(i)}, \alpha_{j,R(i)}) \end{aligned} \tag{3}$$

$L(i)$ and $R(i)$: left and right childs of (binarized) src tree for non-leaf nodes.

Suspicious Results on Multi-Tasking

Promoting Source Syntax in Transformer

See the poster of CICLING 2019.
Dominik Macháček:

Predicting Target Syntax

My students Dan Kondratyuk and Ronald Cardenas retried Nadejde et al. (2017) with:

- sequence-to-sequence model,
- Transformer model.

Predicting target syntax using:

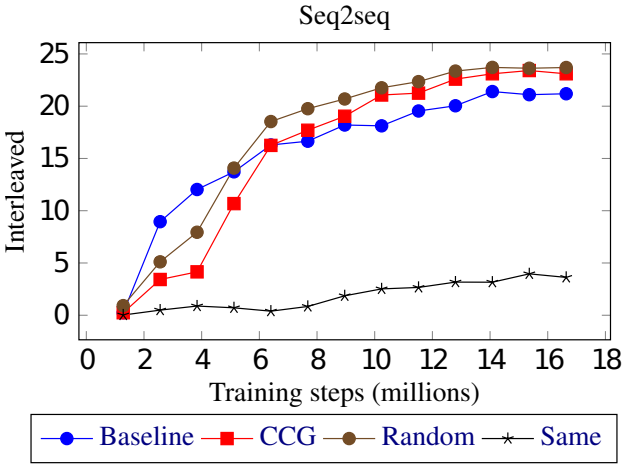
- a secondary decoder

  (The sequence of CCG tags may not match the translated sentence.)
- interleaving.

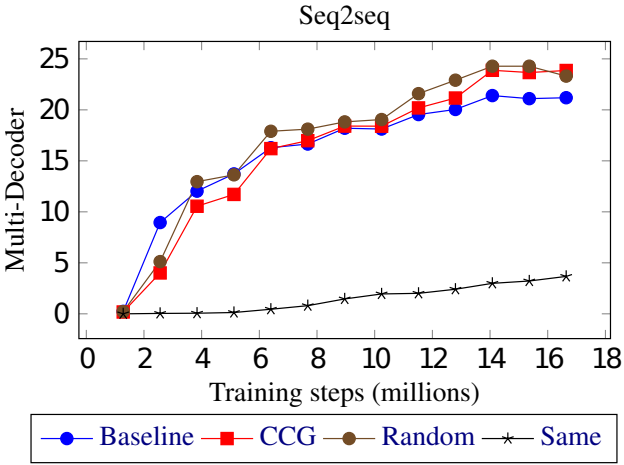
As tags, they used:

- correct CCG tags, • random tags, • a single dummy tag.

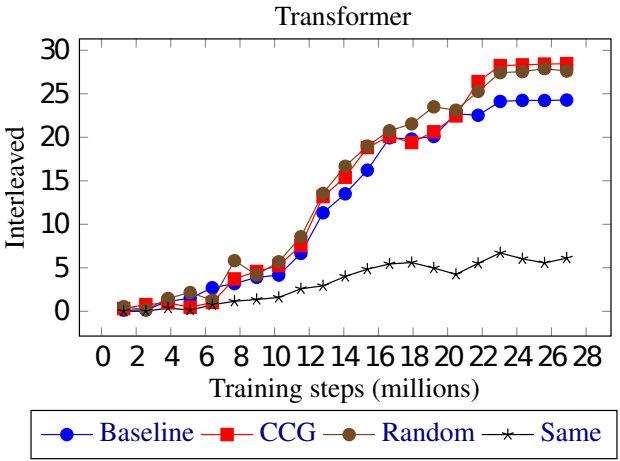
Predicting Target Syntax



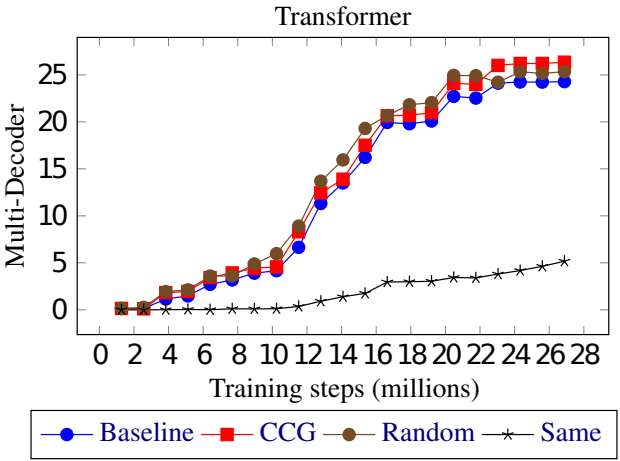
Predicting Target Syntax



Predicting Target Syntax



Predicting Target Syntax




BACKPROPAGANDA

Summary

- Transformer is a great replacement for RNN:
- Explicit syntax can be useful.
  - Very many options possible.
  - All so far tested on a handful of languages and in incomparable settings.
  - Some gains hard to reproduce.
  - The best setup very unclear at the moment.

... Stay tuned or bring your own option.

References

Joost Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Simaan. 2017. Graph convolutional encoders for syntax-aware neural machine translation. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 1947–1957. Association for Computational Linguistics.

Huadong Chen, Shujian Huang, David Chiang, and Jiajun Chen. 2017a. Improved Neural Machine Translation with a Syntax-Aware Encoder and Decoder. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1936–1945, Vancouver, Canada, July. Association for Computational Linguistics.

Kehai Chen, Rui Wang, Masao Utiyama, Lemao Liu, Akihiro Tamura, Eiichiro Sumita, and Tiejun Zhao. 2017b. Neural Machine Translation with Source Dependency Representation. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 2846–2852, Copenhagen, Denmark, September. Association for Computational Linguistics.

Maria Nadejde, Siva Reddy, Rico Sennrich, Tomasz Dwojak, Marcin Junczys-Dowmunt, Philipp Koehn, and Alexandra Birch. 2017. Predicting target language ccg supertags improves neural machine translation. In Proceedings of the Second Conference on Machine Translation, Volume 1: Research Paper, pages 68–79, Copenhagen, Denmark, September. Association for Computational Linguistics.

Rico Sennrich and Barry Haddow. 2016. Linguistic input features improve neural machine translation. In Proceedings of the First Conference on Machine Translation, pages 83–91, Berlin, Germany, August. Association for Computational Linguistics.

Aleš Tamchyna, Marion Weller-Di Marco, and Alexander Fraser. 2017. Modeling target-side inflection in neural machine translation. In Proceedings of the Second Conference on Machine Translation, Volume 1: Research Paper, pages 32–42, Copenhagen, Denmark, September. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus,