Pervasive Attention
2D Convolutional Neural Networks for Sequence-to-Sequence Prediction

Maha Elbayad†‡ Laurent Besacier‡ Jakob Verbeek†
†firstname.lastname@inria.fr
‡firstname.lastname@univ-grenoble-alpes.fr

Code available at: github.com/**elbayadm/attn2d**

Overview

In state-of-the-art encoder-decoder models, the source and target sequences are processed **separately**. The decoder, equipped with an **attention mechanism**, focuses on different parts of the source at each decoding step. However, the attention is limited to assigning weights to the **once and for all** computed encoder states.

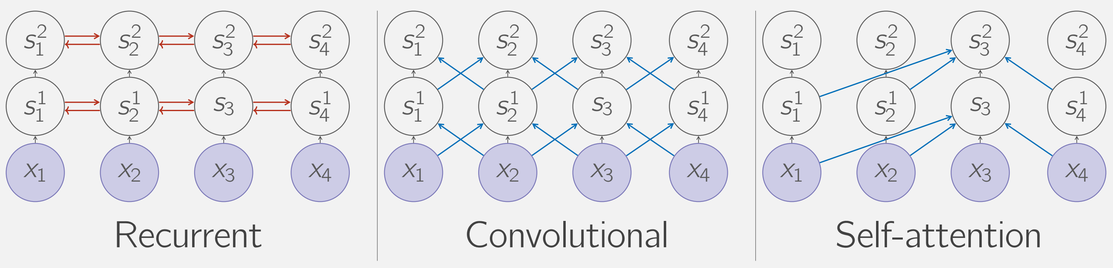
Contributions: build an architecture from the get go around attention by **jointly** encoding the source and target sequences and allowing for different source representations for every target position.

Encoder-Decoders

Encoder

Inputs: source sequence $\mathbf{x} = (x_1, x_2, \ldots, x_{|\mathbf{x}|})$.

Depending on the chosen architecture, the encoder computes the source representations.



Decoder

Inputs: source codes $(s_1, \ldots, s_{|\mathbf{x}|})$ and target sequence $\mathbf{y} = (y_1, y_2, \ldots, y_{|\mathbf{y}|})$.

At every step t:

- Under the architecture, compute the hidden state h_t causally.
- Given the new state, the attention mechanism yields a context c_t.
- $h_t := \text{combine}(h_t, c_t)$.

Pervasive attention: the input

	x_1	x_2	x_3	x_4	x_5
y_1	h^0_{11}	h^0_{12}	h^0_{13}	h^0_{14}	h^0_{15}
y_2	h^0_{21}	h^0_{22}	h^0_{23}	h^0_{24}	h^0_{25}
y_3	h^0_{31}	h^0_{32}	h^0_{33}	h^0_{34}	h^0_{35}
y_4	h^0_{41}	h^0_{42}	h^0_{43}	h^0_{44}	h^0_{45}
y_5	h^0_{51}	h^0_{52}	h^0_{53}	h^0_{54}	h^0_{55}

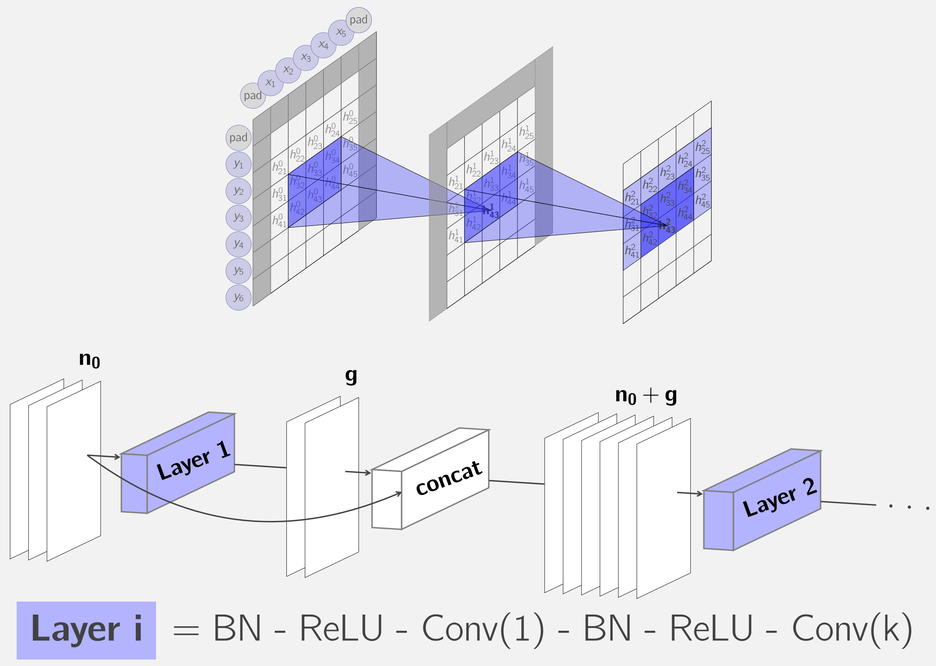
The initial 2D grid:

$\forall i, j : \{1, \ldots |y|\} \times \{1, \ldots |x|\}$

$$\begin{cases} u_i & = U_{y_i} \text{ (target embedding)} \\ v_j & = V_{x_j} \text{ (source embedding)} \\ h_{ij} & = concat(u_i, v_j) \end{cases}$$

Pervasive attention: the convolutional network

- **Causality:** with masked filters in the target direction.
- **Context:** grown with stacked convolutions.
- **Padding:** throughout the network to maintain source/target resolution.



Layer i = BN - ReLU - Conv(1) - BN - ReLU - Conv(k)

- Using a single-block DenseNet (Huang et al., 2017) with **L** layers.
- Each layer grows its input channels by **g**.

References

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In *ICLR*.
Deng, Y., Kim, Y., Chiu, J., Guo, D., and Rush, A. (2018). Latent alignment and variational attention. *arXiv preprint arXiv:1807.03756*.
Edunov, S., Ott, M., Auli, M., Grangier, D., and Ranzato, M. (2018). Classical structured prediction losses for sequence to sequence learning. In *NAACL*.
Gehring, J., Auli, M., Grangier, D., Yarats, D., and Dauphin, Y. (2017). Convolutional sequence to sequence learning. In *ICML*.
Huang, G., Liu, Z., van der Maaten, L., and Weinberger, K. (2017). Densely connected convolutional networks. In *CVPR*.
Sennrich, R., Haddow, B., and Birch, A. (2016). Neural machine translation of rare words with subword units. In *ACL*.
Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In *NIPS*.

Pervasive attention: the aggregation

	x_1	x_2	x_3	x_4	x_5
y_1	h^L_{11}	h^L_{12}	h^L_{13}	h^L_{14}	h^L_{15}
y_2	h^L_{21}	h^L_{22}	h^L_{23}	h^L_{24}	h^L_{25}
y_3	h^L_{31}	h^L_{32}	h^L_{33}	h^L_{34}	h^L_{35}
y_4	h^L_{41}	h^L_{42}	h^L_{43}	h^L_{44}	h^L_{45}
y_5	h^L_{51}	h^L_{52}	h^L_{53}	h^L_{54}	h^L_{55}

To aggregate activations across source positions e.g. with $H_3 = [h^L_{31}, ..., h^L_{3|x|}] \in \mathbb{R}^{d \times |x|}$, we can use:

- Max/average pooling.
- Self-attention:
  $\rho = softmax(H_3^T W + b)$.
  $h_3 = H_3 \rho$.
- A combination of the above.

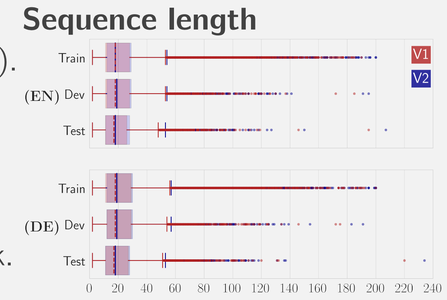
Experimental results

Benchmark: IWSLT'14 German↔English translation.

Pre-processing:
- Tokenization (Moses).
- Lower-casing.
- Length $\leq$ 175 words
- Lengths ratio $\leq$ 1.5.
- Train, Dev, Test : 160k, 7.2k, 6.7k

Sub-word segmentation
- BPE (Sennrich et al., 2016).
- 14k merge operations on EN+DE (V1) / on each separately (V2).
- V1(EN,DE) = 8.8k, 12k, V2(EN,DE) = 13.3k, 13.8k.

Sequence length

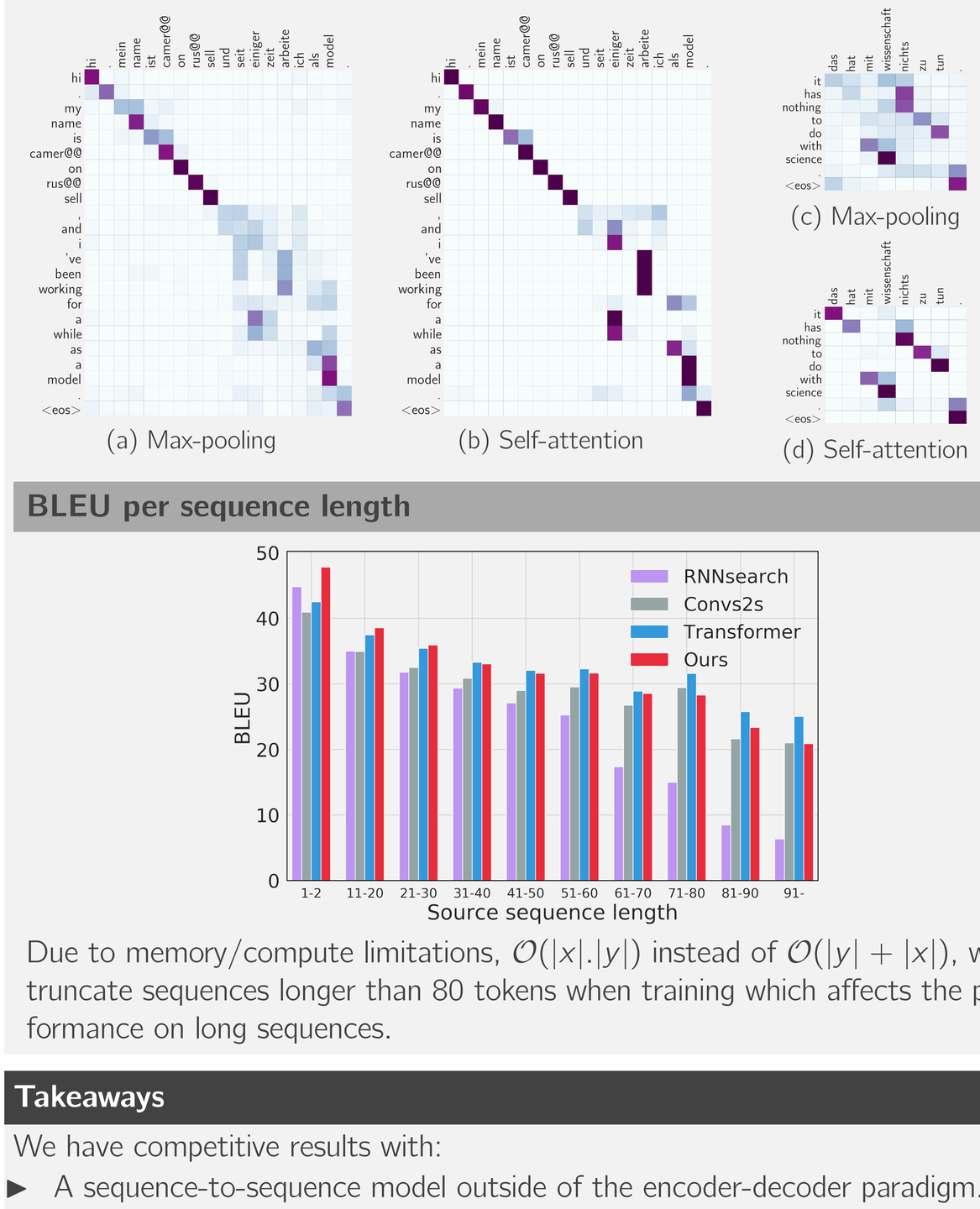


Comparison to the stat-of-the-art

	De→En	#prms	En→De	#prms
■RNNsearch (Bahdanau et al., 2015)	29.98	13M	25.04	15M
Varational attention (Deng et al., 2018)	33.10	-	-	-
■■ConvS2S (MLE) (Gehring et al., 2017)	31.59	21M	27.18	22M
ConvS2S (MLE+SLE) (Edunov et al., 2018)	32.84	-	-	-
■■Transformer (Vaswani et al., 2017), V1	34.42	46M	**28.23**	48M
■■Transformer (Vaswani et al., 2017), V2	**34.44**	52M	28.07	52M
Pervasive attention (ours), V1	33.86	**11M**	27.21	**11M**
Pervasive attention (ours), V2	34.05	22M	27.97	22M

■ models we trained using either our implementation or fairseq. ■ 10 averaged checkpoints.

Alignment visualization



(a) Max-pooling (b) Self-attention (c) Max-pooling (d) Self-attention

BLEU per sequence length

Due to memory/compute limitations, $\mathcal{O}(|x|.|y|)$ instead of $\mathcal{O}(|y| + |x|)$, we truncate sequences longer than 80 tokens when training which affects the performance on long sequences.

Takeaways

We have competitive results with:

- A sequence-to-sequence model outside of the encoder-decoder paradigm.
- A convolutional architecture, proving they work well for NLP problems.
- An implicit attention via re-encoding the source sequence then simply max-pooling the representations.
- Less parameters (at least 1/2 compared to transformer).