Attention is all you need

Christoph Schneider

Universität Heidelberg

25.07.2019

Overview

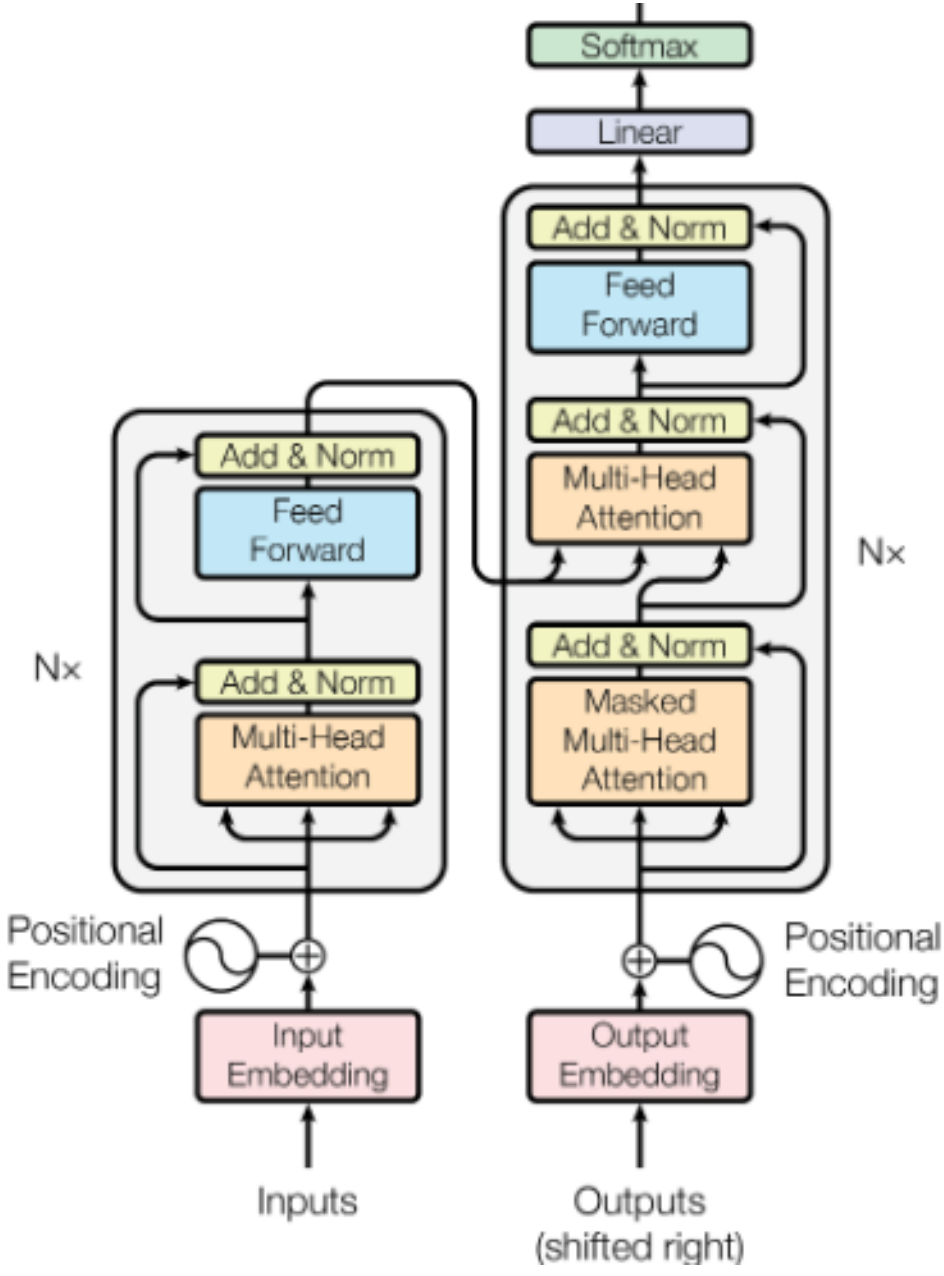
General Info

General Info

- RNNs process one input after the other $\rightarrow$ not parallelisable
- CNNs solve that, but require O(n) operations to relate two input words that are n positions apart.
- The transformer is parallelizable and relates all inputs with O(1) operations.
- $\rightarrow$ Fast training
- $\rightarrow$ Improved handling of LDDs.
- $\rightarrow$ Improved translation quality.

Architecture

Architecture



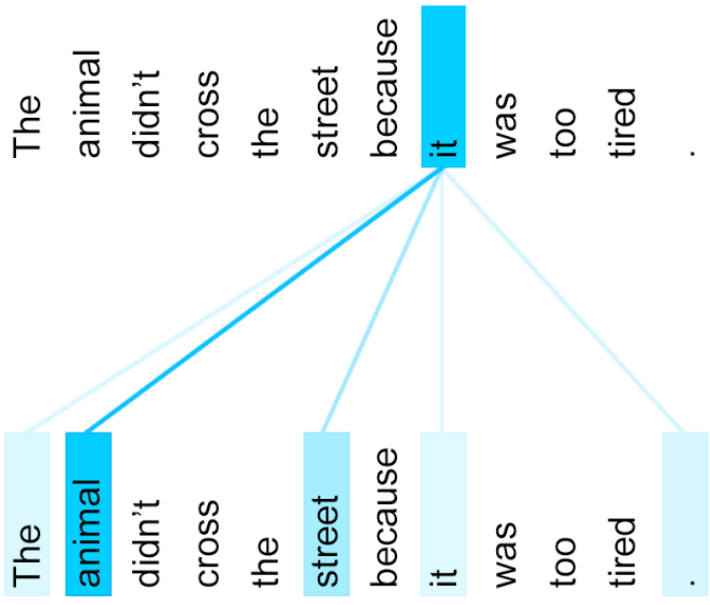
Positional Encodings

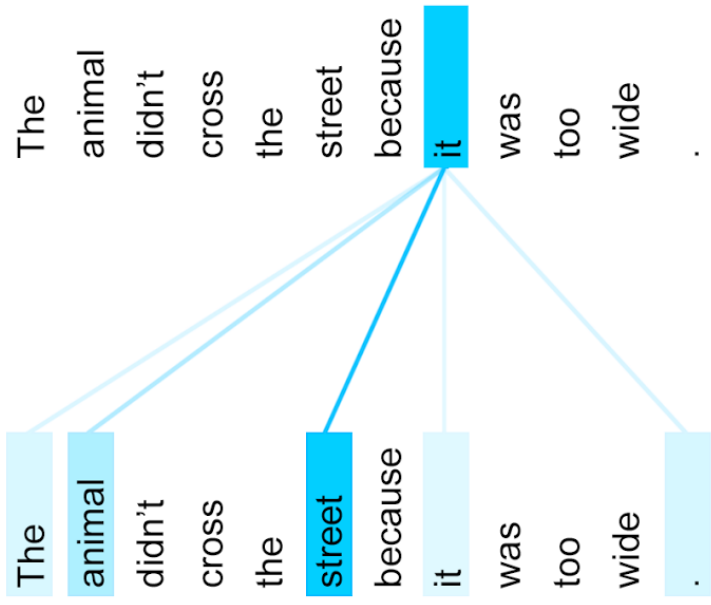
- ConvSeq2Seq used learned positional embeddings
- $\rightarrow$ only sequences of length $<=$ the sequences seen during training get meaningful positional embeddings
- Positional encodings can be used for sequences of any length
- Encoding values are calculated with:

$$PE_{(pos,2i)} = sin(\frac{pos}{10000^{\frac{2i}{d_{model}}}})$$

$$PE_{(pos,2i+1)} = cos(\frac{pos}{10000^{\frac{2i}{d_{model}}}})$$

Self-Attention



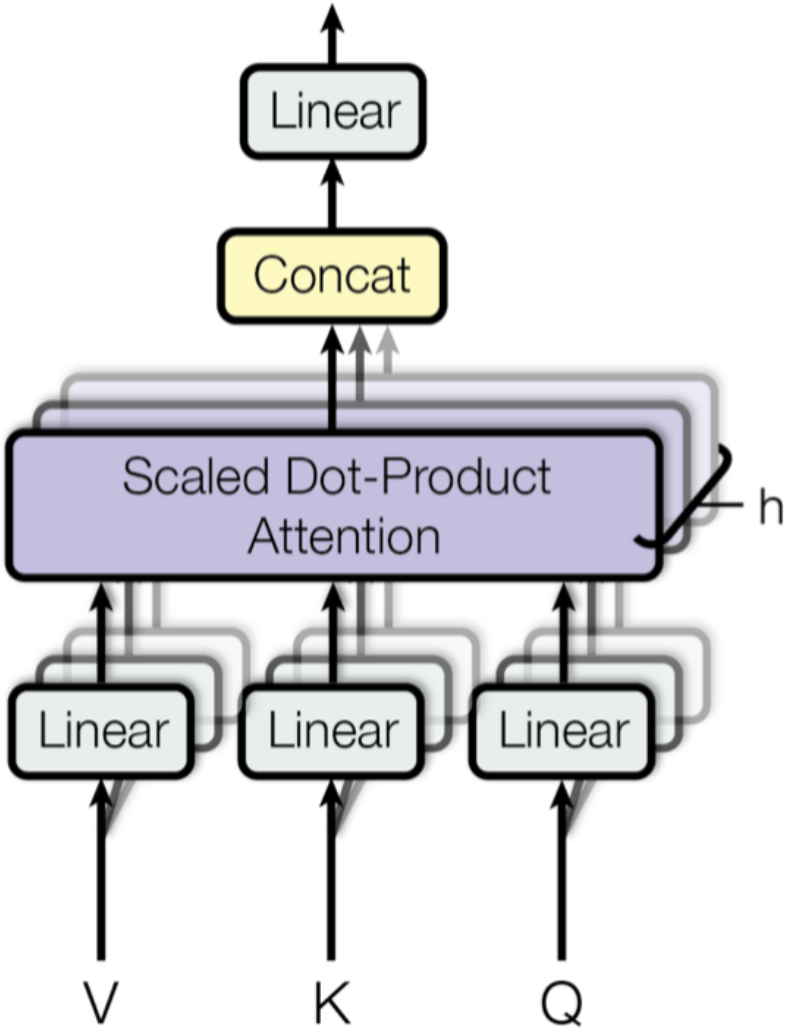


Self-Attention

Self Attention in Matrix Notation

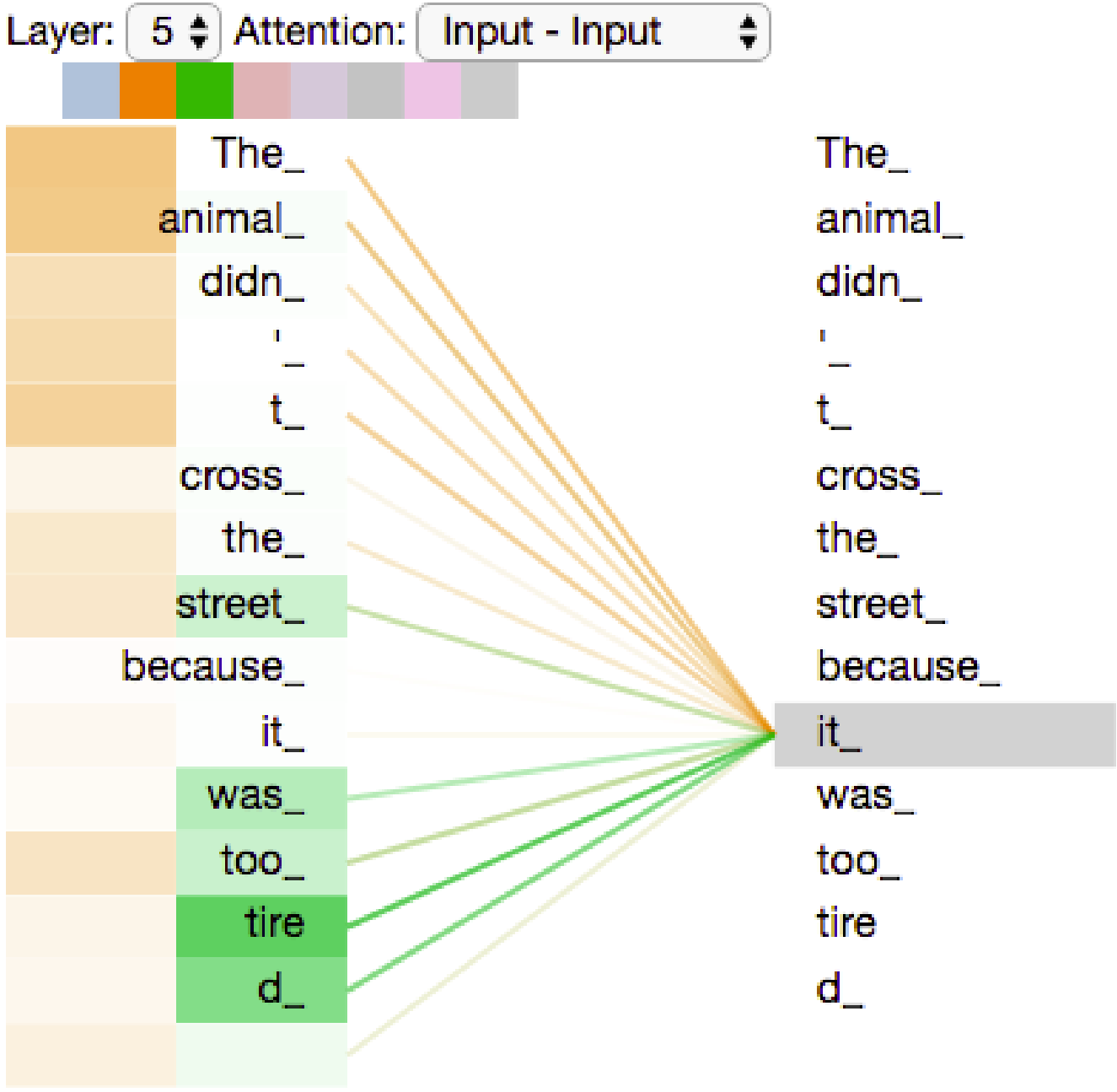
- $Attention(Q, K, V) = softmax\left(\frac{Q*K^T}{\sqrt{d_k}}\right) * V$
- for large embedding sizes d $Q * K^T$ will contain large values. This pushes the softmax into regions with a small gradient. That's why we divide by $\sqrt{d}$

Multi-Head Attention



- Calculate h (Q,K,V) triples independently
- Q,K,V $\in R^{n \times \frac{d}{h}}$
- Calculate attention over each of them
- Concatenate the outputs
- And apply a linear layer

Multi-Head Attention



Complexity

Table 1: Maximum path lengths, per-layer complexity and minimum number of sequential operations for different layer types. n is the sequence length, d is the representation dimension, k is the kernel size of convolutions and r the size of the neighborhood in restricted self-attention.

Layer Type	Complexity per Layer	Sequential Operations	Maximum Path Length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(log_k(n))$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

Teacher Forcing

- During training the decoder doesn't rely on it's own output as an input for the next time step.
- instead we feed the correct output sentence to the decoder.
- This allows the decoder to attend to words in future time steps.
- Solution: Mask out future words, by setting their attention weights to $1e^{-9}$

Experiments

Results

Table 2: The Transformer achieves better BLEU scores than previous state-of-the-art models on the English-to-German and English-to-French newstest2014 tests at a fraction of the training cost.

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [18]	23.75			
Deep-Att + PosUnk [39]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [38]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [32]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [39]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [38]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [9]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	**$3.3 \cdot 10^{18}$**	
Transformer (big)	**28.4**	**41.8**	$2.3 \cdot 10^{19}$	

Results

Table 4: The Transformer generalizes well to English constituency parsing (Results are on Section 23 of WSJ)

Parser	Training	WSJ 23 F1
Vinyals & Kaiser el al. (2014) [37]	WSJ only, discriminative	88.3
Petrov et al. (2006) [29]	WSJ only, discriminative	90.4
Zhu et al. (2013) [40]	WSJ only, discriminative	90.4
Dyer et al. (2016) [8]	WSJ only, discriminative	91.7
Transformer (4 layers)	WSJ only, discriminative	91.3
Zhu et al. (2013) [40]	semi-supervised	91.3
Huang & Harper (2009) [14]	semi-supervised	91.3
McClosky et al. (2006) [26]	semi-supervised	92.1
Vinyals & Kaiser el al. (2014) [37]	semi-supervised	92.1
Transformer (4 layers)	semi-supervised	92.7
Luong et al. (2015) [23]	multi-task	93.0
Dyer et al. (2016) [8]	generative	93.3

Discussion

Questions

- What are some disadvantages of using a Transformer instead of RNNs?
- How is the applied number of heads determined?

Questions

- I'm kind of thinking about the trade-off between training time and results, especially after seeing XLNET's training cost without fine-tuning... Lies the computational benefit of Transformer architectures only in being parallelizable, or is the computational cost generally lower compared to the one of Recurrent Architectures? Maybe you find some comparison charts of training time with results, it would much be appreciated

Questions

- Where does 'query', 'key' and 'values' come from? What can we imagine under that? How are they created?

- About section 3.5: Why are two functions formulated, one with sin and one with cos? Where does i come from and why does it differ from d_{model}?

Questions

- - The more heads are used the more the embedding size is reduced (multihead attention). This is done to keep a small training time, but wouldn't more attention heads be more useful if the same or bigger embedding size were used? Or more accurately, how do we know the reduction of the embedding size doesn't hurt the overall model performance to the point that the improvement due to using more attention layers is lost?

Questions

- The way I interpret the paper, the actual advantages of the transformer architecture is the applicability to different tasks with at most minimal tuning and the reduced training time, because the increase in BLEU is rather small: 0.51 more BLEU points for the EN-FR task as shown in table 2. But as far as I know the transformer has become the de facto go to architecture in seq2seq learning. Is this due to any further improvements in the architecture itself, meaning that one such improved transformer far outperforms the other models in the table, or due to the possibility to build far bigger models because of the reduced training time, meaning "we use 24 transformer blocks and improve upon the state of the art (which uses only 12 transformer blocks)"?

Questions

- Positional Embeddings The authors have this number 10000 in their positional embeddings formula. This seems to be chosen so that even for long sentences, half of the word embeddings stay untouched for sine and are just offset by 1 for cosine. My question is: Do we think the model can still learn something if this number is set to 100 or even 10? Or is all the word embedding information lost then? See attachments for plots and a plotting script.

- Word Order in the Encoder Word order is not encoded at all in the encoder's structure. Can we still assume that the output at "timestep" i corresponds in some way to token i in the source sentence? Do the residual connections enforce this in a way?

Questions

- how does the transformer architecture cope with varying length sequences?
- Why is masking out "illegal" connections from previous output necessary? What would be the harm if the decoder had access to the masked out values?

Questions

- Why is the layer normalization so important for the model?
- Transformers are notorious for being much more difficult to train/fine-tune compared to recurrent models. Why might that be? Is it just that they're very large models or is there more to it?

Questions

- Is there some intuition behind, of the augmentation of the perplexity with the increasing of the label smoothing value?. Table 3, (D)

- On table 3 (A), setting a fix number of parameters $65x10^{65}$. The best BELU score, with a lower perplexity (PPL) is the one using h $=$ 16. Why is not considered like the base model for the comparisons?

Questions

- How does the sinusoidal function help translate long sentences?
- How relevant are each of three listed ways that multi-head attention is used in the work? Is it possible that some affect the performance more than others? (3.2.3)

Questions

- How does the sinusoidal function help translate long sentences?
- How relevant are each of three listed ways that multi-head attention is used in the work? Is it possible that some affect the performance more than others? (3.2.3)

Questions

- What does this network want to figure out or the intuition of the structure?
- It is applied to each position", how is the input sequence segmented into different positions and how is the output of each network aggregated for obtaining the sequence representation?

Questions

- Could you please clarify in your presentation why there should be multiple attention heads in a layer?
- Is it possible to get reliable alignment informations out of Transformer translation models like out of SMTs/RNNs?
- How does the performance of the Transformer system scale with the number of used GPUs?

Questions

- I wonder why this masking is even neccessary? Wouldn't the right-shift prevent the model from seeing future outputs anyway? Does the masking then not do the same thing the right-shift does already, and so wouldn't it then be unneccessary? Would it not technically be impossible for the model anyway, at least for the testing phase, to see future outputs as they would not yet have been generated?

- The authors use scaled dot-product attention. Could be perhaps talk about how the scaling works in the course? (...) As far as I can see they do not report results on their system with additive attention, so how do they know whether the scaled dot-product attention *does* actually perform better than additive attention? Isn't it possible that scaling reduces the effect described, but that additive attention might still yield better results? Am I overlooking something here?