Self-Attention For Generative Models

Ashish Vaswani and Anna Huang

Learning Representations of Variable Length Data

Basic building block of sequence-to-sequence learning

Neural machine translation, summarization, QA, …
Recurrent Neural Networks

Model of choice for learning variable-length representations.

Natural fit for sentences and sequences of pixels.

LSTMs, GRUs and variants dominate recurrent models.
Recurrent Neural Networks

Let's represent this sentence,
But…

Sequential computation inhibits parallelization.

No explicit modeling of long and short range dependencies.

We want to model hierarchy.

**RNNs (w/ sequence-aligned states) seem wasteful!**
Convolutional Neural Networks?

Let’s represent this sentence,
Convolutional Neural Networks?

Trivial to parallelize (per layer).

Exploits local dependencies

‘Interaction distance’ between positions linear or logarithmic.

**Long-distance dependencies require many layers.**
Attention

Attention between encoder and decoder is crucial in NMT.

Why not use attention for representations?
Self-Attention

Let’s represent this sentence,
Text generation
Self-Attention

Constant ‘path length’ between any two positions.

Gating/multiplicative interactions.

Trivial to parallelize (per layer).

Can replace sequential computation entirely?
Previous work

**Classification & regression with self-attention:**
Parikh et al. (2016), Lin et al. (2016)

**Self-attention with RNNs:**
Long et al. (2016), Shao, Gows et al. (2017)

**Recurrent attention:**
Sukhbaatar et al. (2015)
The Transformer

Let's represent this sentence,

Representieren wir diesen Satz,
Encoder Self-Attention

\[ A(Q, K, V) = \text{softmax} \left( \frac{QK^T}{\sqrt{d_k}} \right) V \]
Decoder Self-Attention

Value

Key

Query

matmul_\text{K}

matmul_\text{Q}

matmul_\text{V}

d_1

d_2

d_3

d_4

d_2'

...
Attention is Cheap!

<table>
<thead>
<tr>
<th>Method</th>
<th>FLOPs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self-Attention</td>
<td>$O(\text{length}^2 \cdot \text{dim})$</td>
</tr>
<tr>
<td>RNN (LSTM)</td>
<td>$O(\text{length} \cdot \text{dim}^2)$</td>
</tr>
<tr>
<td>Convolution</td>
<td>$O(\text{length} \cdot \text{dim}^2 \cdot \text{kernel_width})$</td>
</tr>
</tbody>
</table>
Attention is Cheap!

<table>
<thead>
<tr>
<th>Method</th>
<th>Complexity</th>
<th>FLOPs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self-Attention</td>
<td>$O(\text{length}^2 \cdot \text{dim})$</td>
<td>$4 \cdot 10^9$</td>
</tr>
<tr>
<td>RNN (LSTM)</td>
<td>$O(\text{length} \cdot \text{dim}^2)$</td>
<td>$16 \cdot 10^9$</td>
</tr>
<tr>
<td>Convolution</td>
<td>$O(\text{length} \cdot \text{dim}^2 \cdot \text{kernel_width})$</td>
<td>$6 \cdot 10^9$</td>
</tr>
</tbody>
</table>

length=1000  dim=1000  kernel_width=3
Attention: a weighted average
I kicked the ball. Who did what? To whom?
Self-Attention

I kicked the ball

Who Did what? To whom?

I kicked the ball
Parallel attention heads

I kicked the ball

Who did what?  
To whom?
Attention head: Who

Who
Did what?
I kicked
the
ball
To whom?
Parallel attention heads

Who did what?

I kicked the ball
Parallel attention heads

I kicked the ball

Who Did what? To whom?

I kicked the ball
Parallel attention heads

I

kicked

the

ball

Who

Did what?

To whom?
Self-Attention: Averaging

I kicked the ball. Who did what? To whom?
Attention head: Who

I kicked the ball.
Attention head: Did What?

Who did what? I kicked the ball.
Attention head: To Whom?

Who kicked the ball to whom?
Multihead Attention

Who kicked the ball to whom?
Convolution:
Different linear transformations by relative position.

The cat stuck out its tongue and licked its owner.
Attention: a weighted average

The cat stuck out its tongue and licked its owner.
Multi-head Attention
Parallel attention layers with different linear transformations on input and output.

The cat stuck out its tongue and licked its owner.
Results
## Machine Translation: WMT-2014 BLEU

<table>
<thead>
<tr>
<th></th>
<th>EN-DE</th>
<th>EN-FR</th>
</tr>
</thead>
<tbody>
<tr>
<td>GNMT (orig)</td>
<td>24.6</td>
<td>39.9</td>
</tr>
<tr>
<td>ConvSeq2Seq</td>
<td>25.2</td>
<td>40.5</td>
</tr>
<tr>
<td>Transformer*</td>
<td><strong>28.4</strong></td>
<td><strong>41.8</strong></td>
</tr>
</tbody>
</table>

*Transformer models trained >3x faster than the others.

Frameworks:

tensor2tensor

Sockeye
Importance of residuals

Figure 1: The Transformer - model architecture.
Importance of Residuals

Residuals carry positional information to higher layers, among other information.
Training Details

ADAM optimizer with a learning rate warmup (warmup + exponential decay)

Dropout during training at every layer just before adding residual

Layer-norm

Attention dropout (for some experiments)

Checkpoint-averaging

Label smoothing

Auto-regressive decoding with beam search and length biasing

...
## What Matters?

<table>
<thead>
<tr>
<th>N</th>
<th>$d_{\text{model}}$</th>
<th>$d_{\text{ff}}$</th>
<th>$h$</th>
<th>$d_k$</th>
<th>$d_v$</th>
<th>$P_{\text{drop}}$</th>
<th>$\epsilon_{18}$</th>
<th>train steps</th>
<th>PPL (dev)</th>
<th>BLEU (dev)</th>
<th>params $\times 10^6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>base</td>
<td>6</td>
<td>512</td>
<td>2048</td>
<td>8</td>
<td>64</td>
<td>64</td>
<td>0.1</td>
<td>0.1</td>
<td>100K</td>
<td>4.92</td>
<td>25.8</td>
</tr>
<tr>
<td>(A)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5.29</td>
<td>24.9</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>512</td>
<td></td>
<td>128</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5.00</td>
<td>25.5</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>32</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4.91</td>
<td>25.8</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5.01</td>
<td>25.4</td>
</tr>
<tr>
<td>(B)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5.16</td>
<td>25.1</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5.01</td>
<td>25.4</td>
</tr>
<tr>
<td>(C)</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6.11</td>
<td>23.7</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5.19</td>
<td>25.3</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4.88</td>
<td>25.5</td>
</tr>
<tr>
<td></td>
<td>256</td>
<td></td>
<td></td>
<td>32</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5.75</td>
<td>24.5</td>
</tr>
<tr>
<td></td>
<td>1024</td>
<td></td>
<td></td>
<td>128</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4.66</td>
<td>26.0</td>
</tr>
<tr>
<td></td>
<td>1024</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5.12</td>
<td>25.4</td>
</tr>
<tr>
<td></td>
<td>4096</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4.75</td>
<td>26.2</td>
</tr>
<tr>
<td>(D)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.0</td>
<td>24.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.2</td>
<td>25.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4.95</td>
<td>25.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4.67</td>
<td>25.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5.47</td>
<td>25.7</td>
</tr>
<tr>
<td>(E)</td>
<td>positional embedding instead of sinusoids</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4.92</td>
<td>25.7</td>
</tr>
<tr>
<td>big</td>
<td>6</td>
<td>1024</td>
<td>4096</td>
<td>16</td>
<td></td>
<td></td>
<td>0.3</td>
<td>300K</td>
<td></td>
<td>4.33</td>
<td>26.4</td>
</tr>
</tbody>
</table>
## Generating Wikipedia by Summarizing Long Sequences

msaleh@ et al. submission to ICLR’18

<table>
<thead>
<tr>
<th>Method</th>
<th>ROUGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>seq2seq-attention</td>
<td>12.7</td>
</tr>
<tr>
<td>Transformer-ED (L=500)</td>
<td>34.2</td>
</tr>
<tr>
<td>Transformer-DMCA (L=11000)</td>
<td><strong>36.2</strong></td>
</tr>
</tbody>
</table>
Self-Similarity, Image and Music Generation
Self-similarity in images

Self-Similarity in Images

Starry Night (Van Gogh, June 1889)
Self-similarity in music

Motifs repeat, immediately and also at a distance
Probabilistic Image Generation

Model the joint distribution of pixels

Turning it into a sequence modeling problem

Assigning probabilities allows measuring generalization
Probabilistic Image Generation

RNNs and CNNs are state-of-the-art (PixelRNN, PixelCNN)

**CNNs incorporating gating now match RNNs in quality**

**CNNs are much faster due to parallelization**

A Oord et al. (2016), Salimans et al. (2017), Kalchbrenner et al. (2016)
Probabilistic Image Generation

Long-range dependencies matter for images (e.g. symmetry)

Likely increasingly important with increasing image size

Modeling long-range dependencies with CNNs requires either

- **Many layers** likely making training harder
- **Large kernels** at large parameter/computational cost
Texture Synthesis with Self-Similarity

Texture Synthesis by Non-parametric Sampling (Efros and Leung, 1999)
Non-local Means

Figure 1. Scheme of NL-means strategy. Similar pixel neighborhoods give a large weight, $w(p,q1)$ and $w(p,q2)$, while much different neighborhoods give a small weight $w(p,q3)$. 

BCM 2005
Non-local Means

A Non-local Algorithm for Image Denoising (Buades, Coll, and Morel. CVPR 2005)

Non-local Neural Networks (Wang et al., 2018)
Previous work

**Self-attention:**

Parikh et al. (2016), Lin et al. (2016), Vaswani et al. (2017)

**Autoregressive Image Generation:**

A Oord et al. (2016), Salimans et al. (2017)
Self-Attention

\[ A(Q, K, V) = \text{softmax} \left( \frac{QK^T}{\sqrt{d_k}} \right) V \]
Decoder Self-Attention

Value

Key

Query
Attention is Cheap!

<table>
<thead>
<tr>
<th></th>
<th>FLOPs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self-Attention</td>
<td>$O(\text{length}^2 \cdot \text{dim})$</td>
</tr>
<tr>
<td>RNN (LSTM)</td>
<td>$O(\text{length} \cdot \text{dim}^2)$</td>
</tr>
<tr>
<td>Convolution</td>
<td>$O(\text{length} \cdot \text{dim}^2 \cdot \text{kernel_width})$</td>
</tr>
</tbody>
</table>
Attention is Cheap if length $<<$ dim!

<table>
<thead>
<tr>
<th></th>
<th>FLOPs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self-Attention</td>
<td>$O(length^2 \cdot dim)$ (length=3072 for images)</td>
</tr>
<tr>
<td>RNN (LSTM)</td>
<td>$O(length \cdot dim^2)$</td>
</tr>
<tr>
<td>Convolution</td>
<td>$O(length \cdot dim^2 \cdot kernel_width)$</td>
</tr>
</tbody>
</table>
Combining Locality with Self-Attention

Restrict the attention windows to be local neighborhoods

Good assumption for images because of spatial locality
Local 1D Attention

Memory Block

Query Block

q
Image Transformer Layer
Tasks

Super-resolution

Unconditional and Conditional Image generation
Results

Image Transformer
Parmar*, Vaswani*, Uszkoreit, Kaiser, Shazeer, Ku, and Tran. ICML 2018
## Unconditional Image Generation

<table>
<thead>
<tr>
<th>Model</th>
<th>Cifar-10 (Test)</th>
<th>Imagenet (Validation)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PixelRNN</td>
<td>3.00</td>
<td>3.86</td>
</tr>
<tr>
<td>Gated PixelCNN</td>
<td>3.03</td>
<td>3.83</td>
</tr>
<tr>
<td>PixelCNN++</td>
<td>2.92 (dmol)</td>
<td>-</td>
</tr>
<tr>
<td>PixelSNAIL</td>
<td><strong>2.85</strong></td>
<td>3.8</td>
</tr>
<tr>
<td>Image Transformer, 1D local</td>
<td>2.9 (xent)</td>
<td><strong>3.77</strong></td>
</tr>
<tr>
<td>Image Transformer, 1D local</td>
<td>2.9 (dmol)</td>
<td>3.78</td>
</tr>
</tbody>
</table>

Cross entropy of various models on CIFAR-10 and Imagenet datasets.
Cifar10 Samples
CelebA Super Resolution

<table>
<thead>
<tr>
<th>Input</th>
<th>Local 1D Γ=0.8</th>
<th>Local 1D Γ=0.9</th>
<th>Local 1D Γ=1.0</th>
<th>Local 2D Γ=0.8</th>
<th>Local 2D Γ=0.9</th>
<th>Local 2D Γ=1.0</th>
<th>Truth</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

![Image of CelebA Super Resolution with input and local 1D/2D and truth images for different values of Γ (0.8, 0.9, 1.0)].
## CelebA Super Resolution

<table>
<thead>
<tr>
<th>Method</th>
<th>% Fooled</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\Gamma = \text{n/a}$</td>
</tr>
<tr>
<td>ResNet</td>
<td>4.0</td>
</tr>
<tr>
<td>srez GAN (Garcia, 2016)</td>
<td>8.5</td>
</tr>
<tr>
<td>Pixel Recursive (Dahl et al., 2017)</td>
<td>-</td>
</tr>
<tr>
<td>Image Transformer, 1D local</td>
<td>$35.94 \pm 3.0$</td>
</tr>
<tr>
<td>Image Transformer, 2D local</td>
<td>$36.11 \pm 2.5$</td>
</tr>
</tbody>
</table>

Human Eval performance for the Image Transformer on CelebA. The fraction of humans fooled is significantly better than the previous state of art.
Cifar10 SuperResolution
Conditional Image Completion
Music generation using relative self-attention


Blog post: https://magenta.tensorflow.org/music-transformer
Raw representations in music and language

Language
- text
- speech

Music
- score
- performance
- sound
- composer
- performer
- instrument
- listener

(Image from Simon & Oore, 2016)
Music Language model:
Prior work Performance RNN (Simon & Oore, 2016)
Continuations to given initial motif

Given motif

RNN-LSTM

Transformer

Music Transformer
Continuations to given initial motif

Given motif
Continuations to given initial motif

Given motif
Continuations to given initial motif

Given motif

RNN-LSTM
Continuations to given initial motif

Given motif

RNN-LSTM
Continuations to given initial motif

Given motif

RNN-LSTM

Transformer
Continuations to given initial motif

Given motif

RNN-LSTM

Transformer
Continuations to given initial motif

Given motif

RNN-LSTM

Transformer

Music Transformer
Continuations to given initial motif

Given motif

- RNN-LSTM
- Transformer
- Music Transformer
Self-Similarity in Music
Sample from Music Transformer
Attention: a weighted average
Attention: a weighted average
Convolution:
Different linear transformations by relative position.
Relative attention (Shaw et al, 2018)
Multihead attention + convolution?
Closer look at attention

$$softmax(QK^\top)$$
Closer look at relative attention

$$\text{softmax}(QK^T + Qf(E_{rel}))$$

Modulated by relative positions
<table>
<thead>
<tr>
<th>Model</th>
<th>Position Representation</th>
<th>BLEU En-De</th>
<th>BLEU En-Fr</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transformer Big</td>
<td>Absolute</td>
<td>27.9</td>
<td>41.3</td>
</tr>
<tr>
<td>Transformer Big</td>
<td>Relative</td>
<td>29.2</td>
<td>41.5</td>
</tr>
</tbody>
</table>

Machine Translation (Shaw et al, 2018)
Previous work $O(L^2 D)$: 8.5 GB per layer (Shaw et al, 2018)

Per layer, $L=2048$, $D=512$

$$softmax(QK^\top + Qf(E_{rel}))$$

Relative embeddings $E_{rel}$

Relative distances

-2 -1 0

Multiply by Q
Our formulation $O(LD): 4.2 \text{ MB per layer}$

$$\text{softmax}(QK^\top + \text{skew}(QE_{rel}^\top))$$

Per layer, $L=2048$, $D=512$

**Absolute by relative**

$QE^\top$

**Absolute by absolute**

$\text{Pad}$

$\text{Reshape}$

$\text{Skew}$

$\text{Slice}^{i_q}$
Goal of skewing procedure

Indexed by

absolute by relative

absolute by absolute
Skewing to reduce relative memory from $O(L^2D)$ to $O(LD)$

Per layer, $L=2048$, $D=512$

Previous work

$O(L^2D)$: 8.5 GB

Our work

$O(LD)$: 4.2 MB

Relative embeddings $E_r$

Multiply by $Q$

Pad

Reshape

Slice

Directly multiply by $Q$

$QE^{r^T}$

Skew

$S^{rel}$
A Jazz sample from Music Transformer
A Jazz sample from Music Transformer
Convolutions and Translational Equivariance
Relative positions Translational Equivariance
Relative Attention And Graphs
Relative Attention And Graphs

Relational inductive biases, deep learning, and graph networks. (Battaglia et al., 2018)

Self-Attention With Relative Position Representations (Shaw et al., 2018)
Message Passing Neural Networks

\[
m^{t+1}_v = \sum_{w \in N(v)} M_t(h^t_v, h^t_w, e_{vw})
\]

\[
h^{t+1}_v = U_t(h^t_v, m^{t+1}_v)
\]

\[
\hat{y} = R(\{h^T_v | v \in G\})
\]
Multiple Towers

- Run k smaller copies of the MPNN in parallel.
- Mix node states after each message pass.
- Offers a factor of k speedup for the same node dimension d (> 2x speedup when d=200).
- Also helped improve performance when used with matrix multiply message function.

Slide credit: Justin Gilmer
Self-Attention

Constant ‘path length’ between any two positions.

Unbounded memory.

Trivial to parallelize (per layer).

Models Self-Similarity.

Relative attention provides expressive timing, equivariance, and extends naturally to graphs.
Active Research Area

Non autoregressive transformer (Gu and Bradbury et al., 2018)

Deterministic Non-Autoregressive Neural Sequence Modeling by Iterative Refinement (Lee, Manismov, and Cho, 2018)


Towards a Better Understanding of Vector Quantized Autoencoders Roy, Vaswani, Parmar, Neelakantan, 2018

Blockwise Parallel Decoding For Deep Autogressive Models (NeurIPS 2019) Stern, Shazeer, Uszkoreit,
Transfer learning
Improving Language Understanding by Generative Pre-Training (Radford, Narsimhan, Salimans, and Sutskever)

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (Devlin, Chang, Lee, and Toutanova)
Optimization and Large Models


Self-attention in Other Work.


A Time-Restricted Self-Attention Layer for ASR (ICASSP 2018). Povey, Hadian, Gharemani, Li, Khudanpur.

Ongoing and Future Work
Ongoing

Self-supervision and classification for images and video

Understanding Transfer
Future

Multitask learning

Long-range attention