On the Relationship between Self-Attention & Convolutional Layers

Jean-Baptiste Cordonnier, Andreas Loukas, Martin Jaggi

Presented by Yenson Lau — Layer 6 AI

August 11th, 2021

Motivation

Lots of intuition available for convolutions

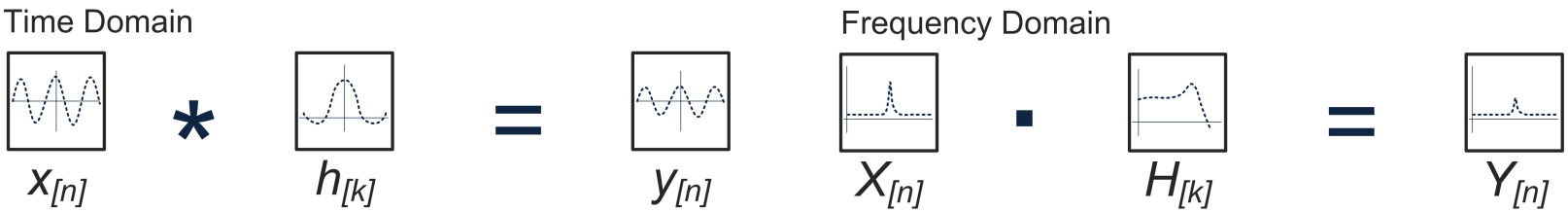
- Great for modeling functions on signals with *translational symmetries*
  - Equivariance / invariance:

$$\mathrm{Conv}(W, \mathcal{T}_\delta X) = \mathcal{T}_\delta \mathrm{Conv}(W, X)$$

  - Associativity / commutativity:

$$\mathrm{Conv}(W', \mathrm{Conv}(W, X)) \;=\; \mathrm{Conv}(\mathrm{Conv}(W', W), X) \;=\; \mathrm{Conv}(\mathrm{Conv}(W, W'), X)$$

  - Diagonalized by the Fourier Transform (source)

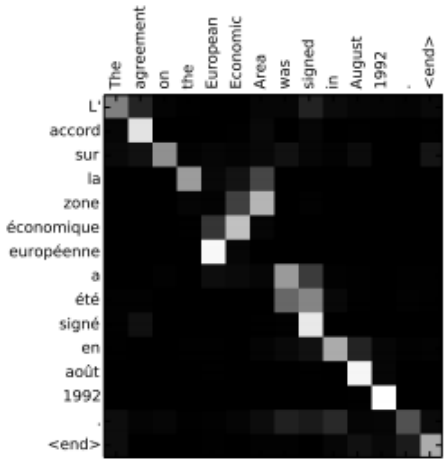


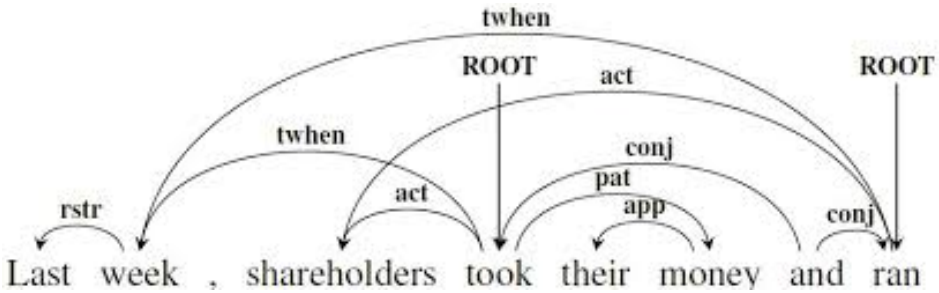
- Great at detecting local patterns in some fixed proximity – as long as order is preserved. Naturally a core component of deep learning models for vision tasks.
  - e.g. LeNet, AlexNet, ResNet, etc.

- For language tasks, however, a "sliding-window" approach alone is insufficient for identifying patterns.

Motivation

- Text does have a loose sequential component but long-range context matters, ordering of words can change in different settings, etc. (source)





- Unlike convolutions, attention can simultaneously attend to every word in an input sequence and produces the best language models, e.g. Transformer, BERT, etc.
- In fact, NN architectures using self-attention (SA) only (without convolution) can compete with SA + convolutional architectures on vision tasks.
  - Stand-Alone Self-Attention in Vision Models, Ramachandran et. al. 2018.
- Can attention be interpreted as a "generalized convolution", and does this interpretation provide meaningful insight into how the mechanism works?

Contributions of the paper

- A constructive theoretical proof that SA can express convolutional layers using relative positional encoding:

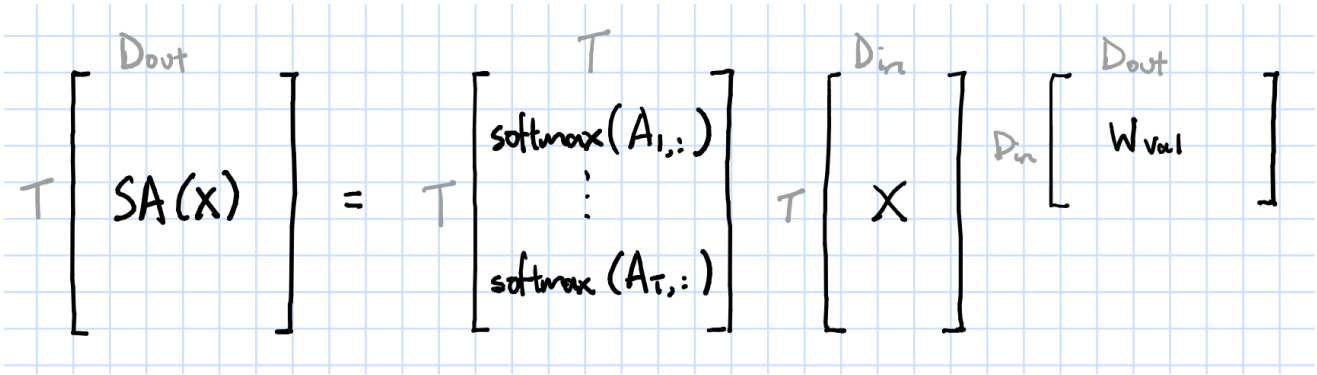
> **Main theorem.** *A* multi-head self-attention *(MHSA) layer with N_h heads of dimension D_h each, final output dimension D_{out}, and a* relative positional encoding *of dimension $D_p \geq 3$ can express any 2D convolutional layer of kernel size $\sqrt{N_h} \times \sqrt{N_h}$ and* $\min(D_h, D_{out})$ *output channels.*

- Experiments showing that (arguably) the first few layers of DNNs using SA do behave similarly to the theoretical construction. Attention probabilities may possibly be interpretable from the lens of "generalized convolution".

Self-Attention

- Let $\boldsymbol{X} \in \mathbb{R}^{T \times D_{in}}$ be an input matrix consisting of T tokens in D_{in} dimensions. For an output token t, the self-attention operation $\mathrm{SA} : T \times D_{in} \rightarrow D_{out}$ is expressed as

$$\mathrm{SA}(\boldsymbol{X})_{t,:} \doteq \mathrm{softmax}(\boldsymbol{A}_{t,:})\boldsymbol{X}\boldsymbol{W}_{val}, \tag{1}$$



- We refer to the the elements of the $T \times T$ matrix

$$\boldsymbol{A} \doteq \boldsymbol{X}\boldsymbol{W}_{qry}\boldsymbol{W}_{key}^{T}\boldsymbol{X}^{T} \tag{2}$$

the *attention scores*, and the softmax outputs as *attention probabilities*.

Self-Attention

- So far the learnable parameters are the *query*, *key*, and *value* matrices

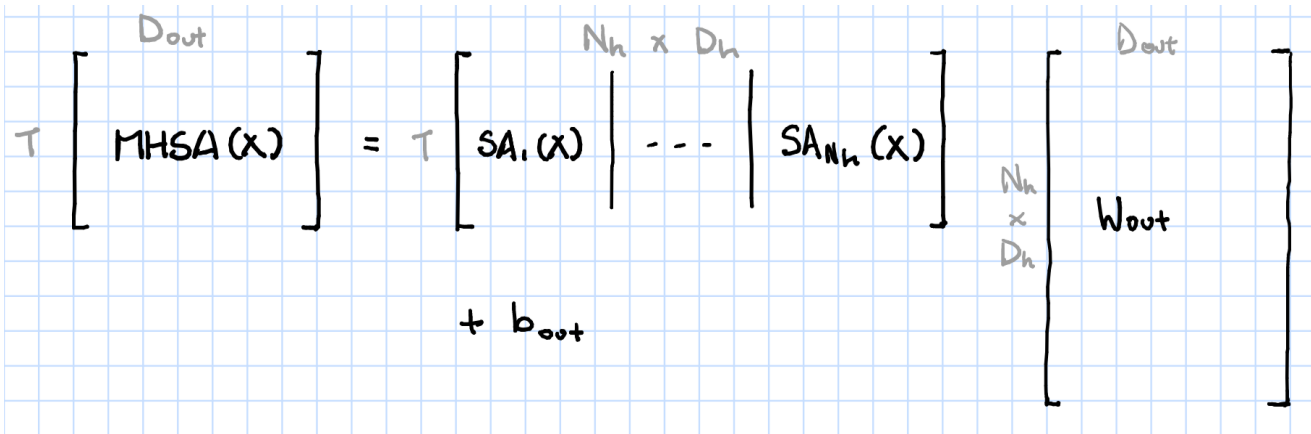
$$\boldsymbol{W}_{qry} \in \mathbb{R}^{D_{in} \times D_k}, \quad \boldsymbol{W}_{key} \in \mathbb{R}^{D_{in} \times D_k}, \quad \text{and} \quad \boldsymbol{W}_{val} \in \mathbb{R}^{D_{in} \times D_{out}}.$$

  For simplicity, ignore residual connections, normalization or constant factors.

- In practice it is beneficial to replicate the SA mechanism into multiple heads, by concatenating the output of N_h heads of output dimension D_h and projecting it to dimension D_{out}.

$$\mathrm{MHSA}(\boldsymbol{X}) \doteq \mathrm{hcat}_{h \in [N_h]} \left[\mathrm{SA}_h(\boldsymbol{X}) \right] \boldsymbol{W}_{out} + \boldsymbol{b}_{out}. \tag{3}$$

  Here $\boldsymbol{W}_{out} \in \mathbb{R}^{N_h D_h \times D_{out}}$ is the projection matrix and $\boldsymbol{b}_{out} \in \mathbb{R}^{D_{out}}$ is a bias term.



Positional Encoding

- Note that $\boldsymbol{A}$ is *permutation equivariant* – shuffling the order of the tokens (rows) in $\boldsymbol{X}$ shuffles the token scores in $\boldsymbol{A}$. *This is not desired behavior.*
  - e.g. "the cat ate the fish" is very different from "the fish ate the cat"

- To overcome this, we can add a (fixed or learned) *positional encoding* $\boldsymbol{P}_{t,:}$, for each token in the sequence, to the input matrix for computing attention scores

$$A \doteq (\boldsymbol{X} + \boldsymbol{P})\boldsymbol{W}_{qry}\boldsymbol{W}_{key}^{T}(\boldsymbol{X} + \boldsymbol{P})^{T}, \tag{4}$$

  where the encoding matrix $\boldsymbol{P}$ has size $T \times D_{in}$.
  - A well-known example for NLP is sinusoidal positional encoding.

Relative positional encoding

- Writing out the positional encoding for a pair of query (the token to compute the representation of) and key tokens (the pixel we are attending to) q, k,

$$\begin{aligned} \boldsymbol{A}^{abs}_{q,k} &= (\boldsymbol{X}_{q,:} + \boldsymbol{P}_{q,:})\boldsymbol{W}_{qry}\boldsymbol{W}^T_{key}(\boldsymbol{X}_{q,:} + \boldsymbol{P}_{q,:})^T \\ &= \boldsymbol{X}_{q,:}\boldsymbol{W}_{qry}\boldsymbol{W}^T_{key}\boldsymbol{X}^T_{k,:} + \boldsymbol{X}_{q,:}\boldsymbol{W}_{qry}\boldsymbol{W}^T_{key}\boldsymbol{P}^T_{k,:} \\ &\quad + \boldsymbol{P}_{q,:}\boldsymbol{W}_{qry}\boldsymbol{W}^T_{key}\boldsymbol{X}^T_{k,:} + \boldsymbol{P}_{q,:}\boldsymbol{W}_{qry}\boldsymbol{W}^T_{key}\boldsymbol{P}^T_{k,:} \end{aligned} \tag{5}$$

- Absolute positional encoding is too strong. *Relative positional encoding* instead considers the positional difference between the query and key tokens:

$$\boldsymbol{A}^{rel}_{q,k} = \boldsymbol{X}^T_{q,:}\boldsymbol{W}_{qry}\boldsymbol{W}^T_{key}\boldsymbol{X}_{k,:} + \boldsymbol{X}^T_{q,:}\boldsymbol{W}^T_{qry}\hat{\boldsymbol{W}}_{key}\boldsymbol{r}_\delta + \boldsymbol{u}^T\boldsymbol{W}_{key}\boldsymbol{X}^T_{k,:} + \boldsymbol{v}^T\hat{\boldsymbol{W}}_{key}\boldsymbol{r}_\delta. \tag{6}$$

  - The (fixed or learned) vectors $\boldsymbol{u}$, $\boldsymbol{v}$ are unique for each head
  - The relative positional encoding $\boldsymbol{r}_\delta \in \mathbb{R}^{D_p}$ depends only on the shift $\delta \doteq k - q$, and is shared by all layers and heads.
  - Key weights are split into $\boldsymbol{W}_{key}$ for the input and $\hat{\boldsymbol{W}}_{key}$ for positional encoding.

- The attention scores are now *translation equivariant* rather than permutation equivariant. This is the desired behavior for convolutional / vision tasks.

- Replace query and key tokens with pixels $\boldsymbol{q}, \boldsymbol{k} \in [W] \times [H]$, and the input with $\boldsymbol{X} \in \mathbb{R}^{W \times H \times D}$. Each attention score now associates a query and key pixel.
- For a pixel $\boldsymbol{p} \in (i, j)$, $\boldsymbol{X}_{\boldsymbol{p},:} \doteq \boldsymbol{X}_{i,j,:}$ and $\boldsymbol{A}_{\boldsymbol{p},:} \doteq \boldsymbol{A}_{i,j,:,:}$.
- Then, analogously to the 1D case,

$$\mathrm{SA}(\boldsymbol{A})_{\boldsymbol{q},:} = \sum_{\boldsymbol{k}} \mathrm{softmax}(\boldsymbol{A}_{\boldsymbol{q},:})_{\boldsymbol{k}} \, \boldsymbol{X}_{\boldsymbol{k},:} \, \boldsymbol{W}_{val}, \tag{7}$$

and MHSA retains the same form as Equation (3). Similarly, relative positional encoding retains the same form as Equation (6) .

- Given an image tensor $\boldsymbol{X} \in \mathbb{R}^{W \times H \times D_{in}}$ and kernel tensor $\boldsymbol{W} \in \mathbb{R}^{K \times K \times D_{in} \times D_{out}}$,

$$\mathrm{Conv}(\boldsymbol{X})_{i,j,:} \doteq \sum_{(\delta_1,\delta_2) \in \Delta_K} \boldsymbol{X}_{i-\delta_1, j-\delta_2,:} \, \boldsymbol{W}_{\delta_1,\delta_2,:,:} + \boldsymbol{b} \in \mathbb{R}^{D_{out}}, \tag{8}$$

  where

$$\Delta_K \doteq \left[-\left\lfloor \frac{K}{2} \right\rfloor, \ldots, \left\lfloor \frac{K}{2} \right\rfloor \right] \times \left[-\left\lfloor \frac{K}{2} \right\rfloor, \ldots, \left\lfloor \frac{K}{2} \right\rfloor \right]$$

  is the set of all shifts represented by a $K \times K$ kernel.

- We depart from the original notation slightly by using the *unflipped* convolution. In the paper, the summand is flipped to $\boldsymbol{X}_{i+\delta_1, j+\delta_2,:} \boldsymbol{W}_{\delta_1,\delta_2,:,:}$. The theorem in the paper is not changed by this.

- **Main theorem.** *A* multi-head self-attention *(MHSA) layer with* N_h *heads of dimension* D_h *each, final output dimension* D_{out}*, and a* relative positional encoding *of dimension* $D_p \geq 3$ *can express any 2D convolutional layer of kernel size* $\sqrt{N_h} \times \sqrt{N_h}$ *and* $\min(D_h, D_{out})$ *output channels.*

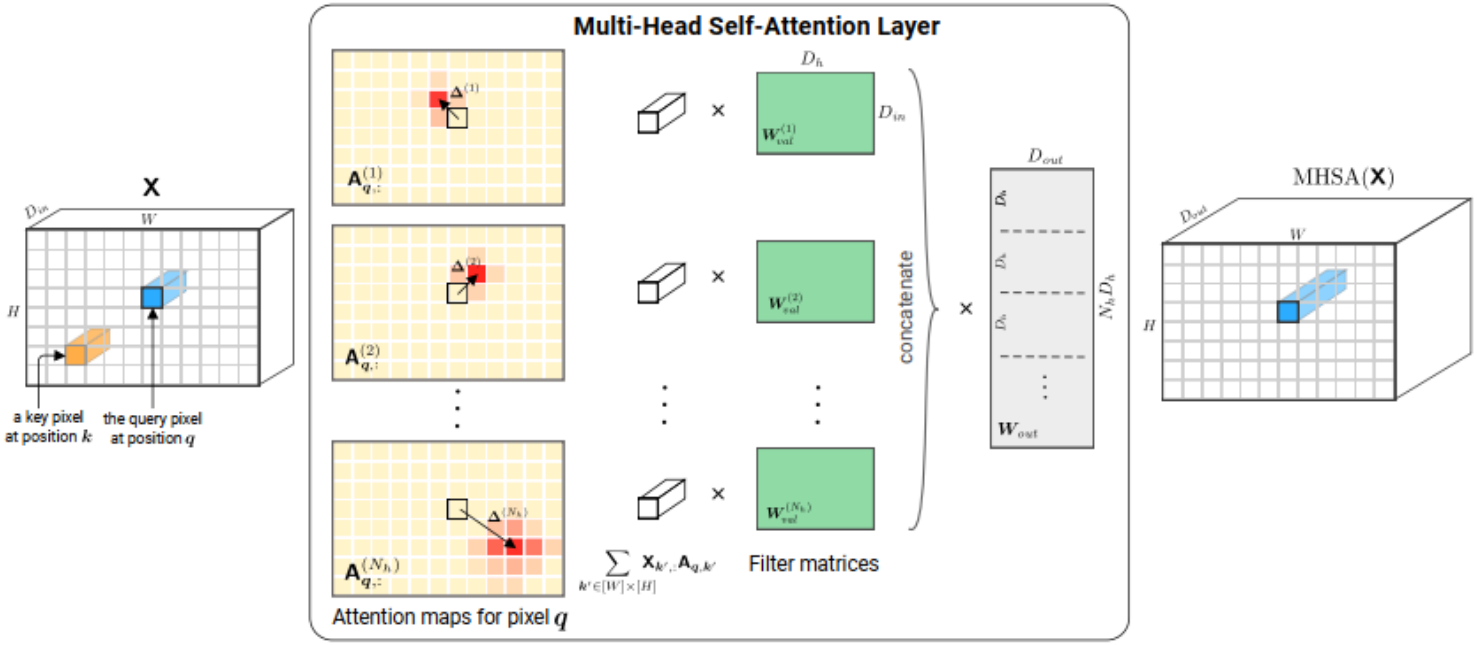


Figure 1: Illustration of a Multi-Head Self-Attention layer applied to a tensor image $\mathbf{X}$. Each head h attends pixel values around shift $\boldsymbol{\Delta}^{(h)}$ and learn a filter matrix $\boldsymbol{W}_{val}^{(h)}$. We show attention maps computed for a query pixel at position $\boldsymbol{q}$.

- The theorem is a consequence of two lemmas.

- **Lemma 1.** *Consider a MHSA layer consisting of $N_h = K^2$ heads, $D_h \geq D_{out}$ and let $\boldsymbol{f} : [N_h] \to \Delta_K$ be a bijective mapping of heads onto shifts. Further, suppose that for every head*

$$\text{softmax}(\boldsymbol{A}^{(h)}_{\boldsymbol{q},:})_{\boldsymbol{k}} = \begin{cases} 1 & \text{if } \boldsymbol{f}(h) = \boldsymbol{q} - \boldsymbol{k}, \\ 0 & \text{otherwise.} \end{cases} \tag{9}$$

Then for any convolutional layer with a $K \times K$ kernel and D_{out} output channels, there exists $\{\boldsymbol{W}^{(h)}_{val}\}_{h \in [N_h]}$ such that $\text{MHSA}(\boldsymbol{X}) = \text{Conv}(\boldsymbol{X})$, for any kernel tensor $\boldsymbol{W} \in \mathbb{R}^{K \times K \times D_{in} \times D_{out}}$, and for all $\boldsymbol{X} \in \mathbb{R}^{W \times H \times D_{in}}$.

- *Proof of Lemma 1.* Rewrite Equation (3) as

$$\text{MHSA}(\boldsymbol{X}) = \boldsymbol{b}_{out} + \sum_{h \in [N_h]} \text{softmax}(\boldsymbol{A}^{(h)})\ \boldsymbol{X}\ \underbrace{\boldsymbol{W}^{(h)}_{val}\ \boldsymbol{W}^{(h)}_{out}}_{\boldsymbol{W}^{(h)}} . \tag{10}$$

Here $\boldsymbol{W}^{(h)}_{out} \in \mathbb{R}^{D_h \times D_{out}}$ is the output matrix for head h, or in MATLAB notation,

$$\boldsymbol{W}^{(h)}_{out} = \boldsymbol{W}[(h-1) * D_h + 1 : hD_h\ ,\ :\ ] \quad \in \quad \mathbb{R}^{D_h \times D_{out}}.$$

- *Proof of Lemma 1 (cont).* Consider a single "query" pixel from $\mathrm{MHSA}(\boldsymbol{X})$:

$$\mathrm{MHSA}(\boldsymbol{X})_{\boldsymbol{q},:} = \boldsymbol{b}_{out} + \sum_{h \in [N_h]} \left(\sum_{\boldsymbol{k} \in [T]} \mathrm{softmax}(\boldsymbol{A}^{(h)}_{\boldsymbol{q},:})_{\boldsymbol{k}} \; \boldsymbol{X}_{\boldsymbol{k},:} \right) \boldsymbol{W}^{(h)}. \tag{11}$$

  The summand $\sum_{\boldsymbol{k}} \mathrm{softmax}(\boldsymbol{A}^{(h)}_{\boldsymbol{q},:})_{\boldsymbol{k}} \; \boldsymbol{X}_{\boldsymbol{k},:}$ is a weighted average on the rows of $\boldsymbol{X}$.

- Applying the assumption from Equation (9), each of the softmax weights pick out a single pixel $\boldsymbol{k} = \boldsymbol{q} - \boldsymbol{f}(h)$:

$$\mathrm{MHSA}(\boldsymbol{X})_{\boldsymbol{q},:} = \boldsymbol{b}_{out} + \sum_{h \in [N_h]} \boldsymbol{X}_{\boldsymbol{q}-\boldsymbol{f}(h),:} \boldsymbol{W}^{(h)}. \tag{12}$$

  If we set $K = \sqrt{N_h}$, then it is clearly possible to index all the shifts using $h \in [N_h]$, i.e. $\boldsymbol{f}([N_h]) = \Delta_K$. Therefore we can rewrite the expression as

$$\mathrm{MHSA}(\boldsymbol{X})_{\boldsymbol{q},:} = \boldsymbol{b}_{out} + \sum_{h \in [N_h]} \boldsymbol{X}_{\boldsymbol{q}-\boldsymbol{f}(h),:} \boldsymbol{W}_{\boldsymbol{f}(h),:,:} = \mathrm{Conv}(\boldsymbol{X})_{\boldsymbol{q},:}. \qquad \square \tag{13}$$

- Lemma 1 says that, with the right choice of relative positional encoding (and a very restricted set of attention parameters), *a MHSA layer is exactly equivalent to a convolutional layer*.

- The number of linearly independent filters expressible by $\boldsymbol{W}$ is clearly limited by $\min(D_h, D_{out})$. Therefore, to express D_h convolutional filters, it is best to simply let $D_{out} = N_h D_h$ and have $\boldsymbol{W}^{(h)} \in \mathbb{R}^{D_h \times D_h}$.

- There is a generalized version of this lemma in the Appendix of the paper that delves into the space of filters spannable for a given set of dimensions D_h, N_h, D_{out}, etc., for interested readers.

- **Lemma 2.** *There exists a relative encoding scheme* $\{\boldsymbol{r}_{\boldsymbol{\delta}} \in \mathbb{R}^{D_p}\}_{\boldsymbol{\delta} \in \mathbb{Z}^2}$ *with* $D_p \geq 3$ *and parameters* $\boldsymbol{W}_{qry}$, $\boldsymbol{W}_{key}$, $\hat{\boldsymbol{X}}_{key}$, $\boldsymbol{u}$ *with* $D_p \leq D_k$ *such that, for every* $\boldsymbol{\tau} \in \Delta_K$ *there exists some vector* $\boldsymbol{v}(\boldsymbol{\tau})$ *that yields* $\operatorname{softmax}(\boldsymbol{A}_{\boldsymbol{q},:})_{\boldsymbol{k}} = 1$ *if* $\boldsymbol{k} - \boldsymbol{q} = \boldsymbol{\tau}$, *and zero otherwise.*

- *Proof of Lemma 2.* Start with the relative positional encoding of Equation (6):

$$\boldsymbol{A}^{rel}_{\boldsymbol{q},\boldsymbol{k}} = \boldsymbol{X}^T_{\boldsymbol{q},:} \boldsymbol{W}_{qry} \boldsymbol{W}^T_{key} \boldsymbol{X}_{k,:} + \boldsymbol{X}^T_{\boldsymbol{q},:} \boldsymbol{W}^T_{qry} \hat{\boldsymbol{W}}_{key} \boldsymbol{r}_{\boldsymbol{\delta}} + \boldsymbol{u}^T \boldsymbol{W}_{key} \boldsymbol{X}^T_{k,:} + \boldsymbol{v}^T \hat{\boldsymbol{W}}_{key} \boldsymbol{r}_{\boldsymbol{\delta}}.$$

  Since the desired attention probabilities (9) from Lemma 1 are independent of the input $\boldsymbol{X}$, set $\boldsymbol{W}_{key} = \boldsymbol{W}_{qry} = 0$.

  This leaves only the final term. Setting $\hat{\boldsymbol{W}}_{key} = [\ \boldsymbol{I}_{D_p}\ ;\ 0\ ] \in \mathbb{R}^{D_k \times D_p}$ (recall $D_p \leq D_k$) yields

$$\boldsymbol{A}_{\boldsymbol{q},\boldsymbol{k}} = \boldsymbol{v}^T_{1:D_p} \boldsymbol{r}_{\boldsymbol{\delta}}.$$

- *Proof of Lemma 2 (cont).* Recall that $\boldsymbol{\delta} \doteq \boldsymbol{k} - \boldsymbol{q}$. Now suppose some $\boldsymbol{v}, \boldsymbol{r}_{\boldsymbol{\delta}}$ could be found such that

$$\boldsymbol{A}_{\boldsymbol{q},\boldsymbol{k}} = -\alpha(\|\boldsymbol{\delta} - \boldsymbol{\tau}\|^2 + c) \tag{14}$$

so that the maximum score $-\alpha c$ is achieved when $\boldsymbol{\delta} = \boldsymbol{\tau}$, then

$$\begin{aligned}\lim_{\alpha\to\infty} \text{softmax}(\boldsymbol{A}_{\boldsymbol{q},:})_{\boldsymbol{k}} &= \lim_{\alpha\to\infty} \frac{e^{-\alpha(\|\boldsymbol{\delta}-\boldsymbol{\tau}\|^2+c)}}{\sum_{\boldsymbol{\delta}'\in\Delta_K} e^{-\alpha(\|\boldsymbol{\delta}'-\boldsymbol{\tau}\|^2+c)}} \\ &= \frac{1_{\boldsymbol{\delta}=\boldsymbol{\tau}}}{1+\lim_{\alpha\to\infty}\sum_{\boldsymbol{\delta}'\neq\boldsymbol{\delta}} e^{-\alpha\|\boldsymbol{\delta}'-\boldsymbol{\tau}\|^2}} = 1_{\boldsymbol{\delta}=\boldsymbol{\tau}}.\end{aligned}$$

- Choosing $\boldsymbol{v}_{1:D_p} = -\alpha(1, -2\tau_1, -2\tau_2)$ and $\boldsymbol{r}_{\boldsymbol{\delta}} = (\|\boldsymbol{\delta}\|^2, \delta_1, \delta_2)$, yields

$$\begin{aligned}\boldsymbol{A}_{\boldsymbol{q},\boldsymbol{k}} &= \boldsymbol{v}_{1:D_p}^T \boldsymbol{r}_{\boldsymbol{\delta}} \\ &= -\alpha(\|\boldsymbol{\delta}\|^2 - 2\tau_1\delta_1 - 2\tau_2\delta_2) \\ &= -\alpha(\|\boldsymbol{\delta} - \boldsymbol{\tau}\|^2 - \|\boldsymbol{\tau}\|^2).\end{aligned}$$

Setting $c = -\|\boldsymbol{\tau}\|^2$ recovers Equation (14) and completes the proof. □

- The encoding scheme satisfying Lemma 2 is a *quadratic encoding scheme*, and is achieved by setting

$$\begin{aligned} \boldsymbol{v}_{1:D_p}^{(h)} &\doteq -\alpha^{(h)}(1, -2\tau_1^{(h)}, -2\tau_2^{(h)}), \\ \boldsymbol{r}_{\boldsymbol{\delta}} &\doteq (\|\boldsymbol{\delta}\|^2,\ \delta_1,\ \delta_2), \\ \boldsymbol{W}_{qry} &= \boldsymbol{W}_{key} \doteq 0, \\ \hat{\boldsymbol{W}}_{key} &\doteq [\ \boldsymbol{I}_{D_p}\ ;\ 0\ ]. \end{aligned} \tag{15}$$

  Here the learned parameters $\boldsymbol{\tau}^{(h)} = (\tau_1^{(h)}, \tau_2^{(h)})$ and $\alpha^{(h)}$ determine the center and width of attention of each head, and $\boldsymbol{\delta} = (\delta_1, \delta_2)$ and therefore $\boldsymbol{r}_{\boldsymbol{\delta}}$ is fixed and expresses the relative shift between query and key pixels.

- Although the proof requires $\alpha \to \infty$ to satisfy the assumption (9) of Lemma 1, finite precision arithmetic performs hard attention with sufficiently large enough α. E.g. for `Float32`, set $\alpha \geq 46$.

- The lemma, and thus the theorem, can be extended in a straightforward manner to cover K-dimensional convolutions, with $D_p = K + 1$.

Experimental Setup

- **Baseline.** Standard ResNet18 on CIFAR-10
- **Challenger models.**
  - 6 MHSA layers. Each layer consists of a MHSA step, followed by dense, dropout and LayerNorm sublayers.
  - Use 2x2 invertible downsampling to reduce image size (attention coefficients scale quadratically to image size)
  - The last layer representation is average pooled and fed to a linear classifier
- **MHSA variations.** Different types of relative positional encoding

$$\boldsymbol{A}^{rel}_{\boldsymbol{q},\boldsymbol{k}} = \boldsymbol{X}^T_{\boldsymbol{q},:} \boldsymbol{W}_{qry} \boldsymbol{W}^T_{key} \boldsymbol{X}_{k,:} + \boldsymbol{X}^T_{\boldsymbol{q},:} \boldsymbol{W}^T_{qry} \hat{\boldsymbol{W}}_{key} \boldsymbol{r}_{\boldsymbol{\delta}} + \boldsymbol{u}^T \boldsymbol{W}_{key} \boldsymbol{X}^T_{k,:} + \boldsymbol{v}^T \hat{\boldsymbol{W}}_{key} \boldsymbol{r}_{\boldsymbol{\delta}}.$$

  - **SA with quadratic embedding:** Retain final term only and fix the variables using Equation (15). The attention widths $\alpha^{(h)}$ and centers $\tau^{(h)}$ are still learnt.
  - **SA with learned embedding:** Retain final term only but learn v, $\hat{\mathrm{W}}_{key}$, $\mathrm{r}_{\boldsymbol{\delta}}$, with $D_p = D_{out} = 400$. Set $D_h = D_{out}$.
  - **SA with content-based attention:** All terms retained (might actually be just first two terms) and all variables learnable. Same dimensions as above.

Convergence / Testing Accuracy

- **ResNet converges faster:** Probably because SA's inductive bias is not as strong as ResNet, but may also be due to different optimization setup.
- SA models with more learnable parameters converge slower and ends up with lower testing accuracy.

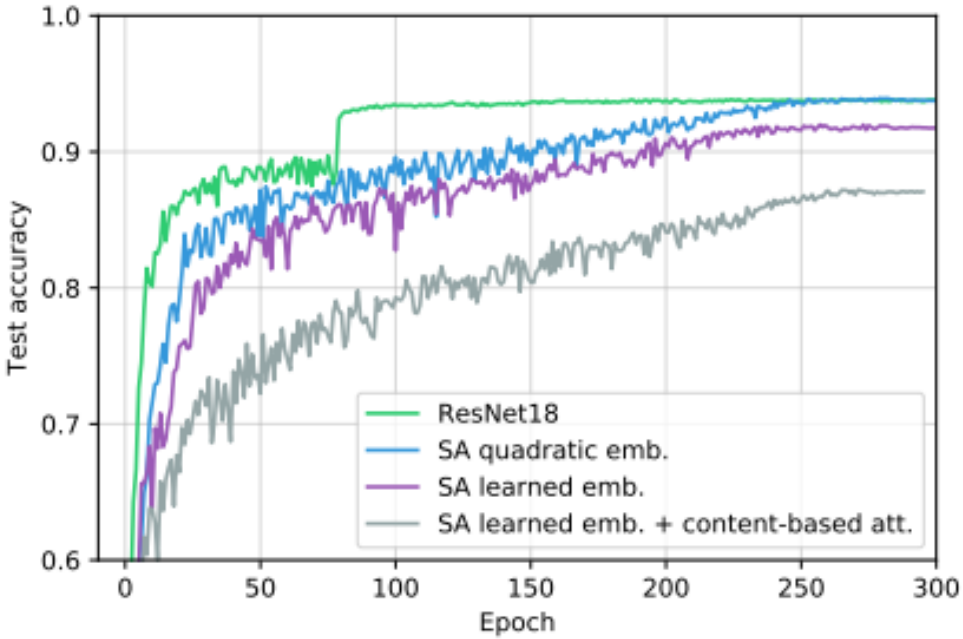


Figure 2: Test accuracy on CIFAR-10.

Do Self-Attention Layers Actually Learn Convolutions? Maybe.

- Recall that from Lemma 1, the critical assumption is

$$\text{softmax}(\boldsymbol{A}^{(h)}_{\boldsymbol{q},:})_{\boldsymbol{k}} = 1_{\boldsymbol{f}(h)=\boldsymbol{q}-\boldsymbol{k}}, \quad \forall h.$$

  If $N_h = |\Delta_K|$, mapping each head to a single shift $\boldsymbol{f}(h) = \boldsymbol{\tau} \in \Delta_K$ yields

$$\text{softmax}(\boldsymbol{A}^{\boldsymbol{\tau}}_{\boldsymbol{q},:})_{\boldsymbol{k}} = 1_{\boldsymbol{\tau}=\boldsymbol{q}-\boldsymbol{k}}, \quad \forall \boldsymbol{\tau}.$$

- MHSA expresses a single convolution operation when attention probabilities are all elements of a *translation group*. For instance, this would also suffice:

$$\text{softmax}(\boldsymbol{A}^{\boldsymbol{\tau}}_{\boldsymbol{q},:})_{\boldsymbol{k}} \propto 1_{|(\boldsymbol{q}-\boldsymbol{k})-\boldsymbol{\tau}|\leq R}, \quad \forall \boldsymbol{\tau}.$$

- Alternatively, if we remove some of the shifts from Δ_K, the heads are still elements of a translation group and we just end up with *sparse* convolution.

- And if there were multiple distribution profiles in the attention probabilities, then MHSA can be seen as expressing a *mixture* of convolution operators.

- **Takeaway:** *It's difficult to judge whether a MHSA layer has a preference for learning convolutional operators or not, especially when N_h is small.*

- Here the attention probabilities are Gaussians shifted from origin $\boldsymbol{q}$, so a single convolution is expressed if they have the same width and different centers.
- The attention probabilities for each of the $N_h = 9$ heads after training:

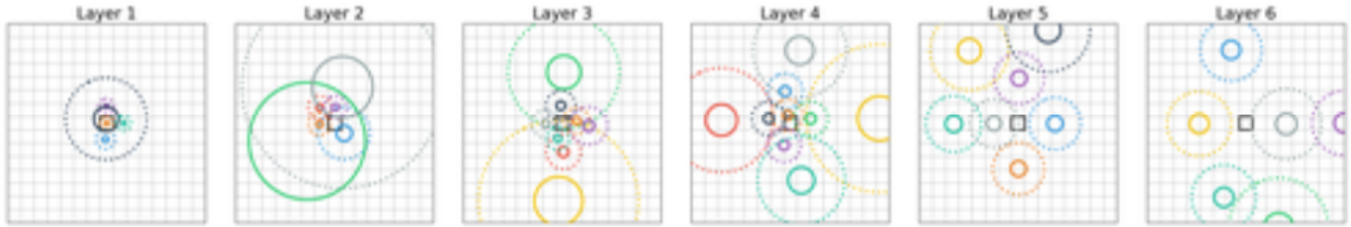


Figure 4: Centers of attention of each attention head (different colors) for the 6 self-attention layers using quadratic positional encoding. The central black square is the query pixel, whereas solid and dotted circles represent the 50% and 90% percentiles of each Gaussian, respectively.

- A significant portion of the heads have similar widths, esp. at lower layers. Higher layers have more varied widths. Notice that the centers tend to separate.
- Again, should we interpret this as a mixture of convolutions? Sparse convolutions? Convolutions with strides? Hard to say.

- $N_h = 16$ heads. Similar to behavior with 9 heads, but now lower layers have groups of heads with small widths + a few heads with widely varying widths.

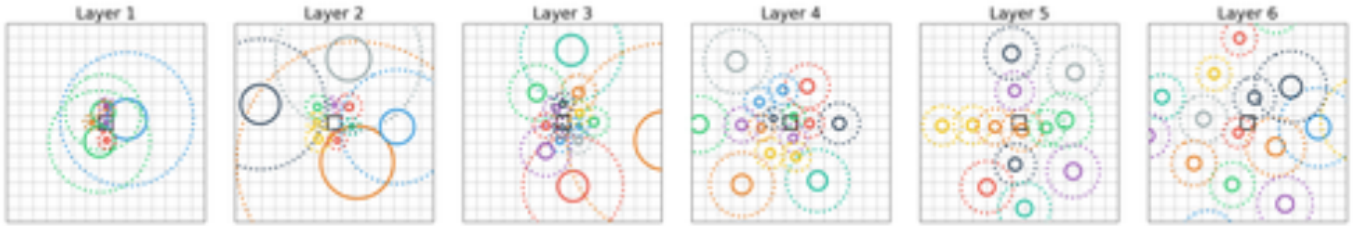


Figure 14: Centers of attention for 16 attention heads (different colors) for the 6 self-attention layers using quadratic positional encoding. The central black square is the query pixel, whereas solid and dotted circles represent the 50% and 90% percentiles of each Gaussian, respectively.

- Non-isotropic Gaussians ($N_h = 9$). Large variations in shapes and sizes, but perhaps useful for representing rotations?

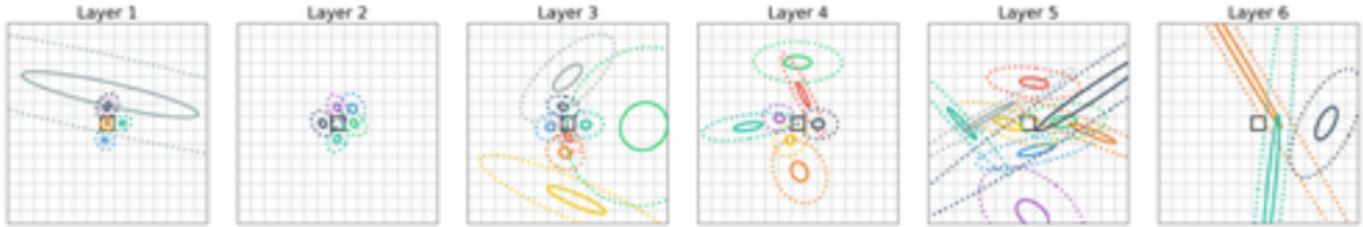


Figure 12: Centers of attention of each attention head (different colors) for the 6 self-attention layers using non-isotropic Gaussian parametrization. The central black square is the query pixel, whereas solid and dotted circles represent the 50% and 90% percentiles of each Gaussian, respectively.

Do Self-Attention Layers Actually Learn Convolutions? Maybe.

Learned relative positional encoding (w/o content)

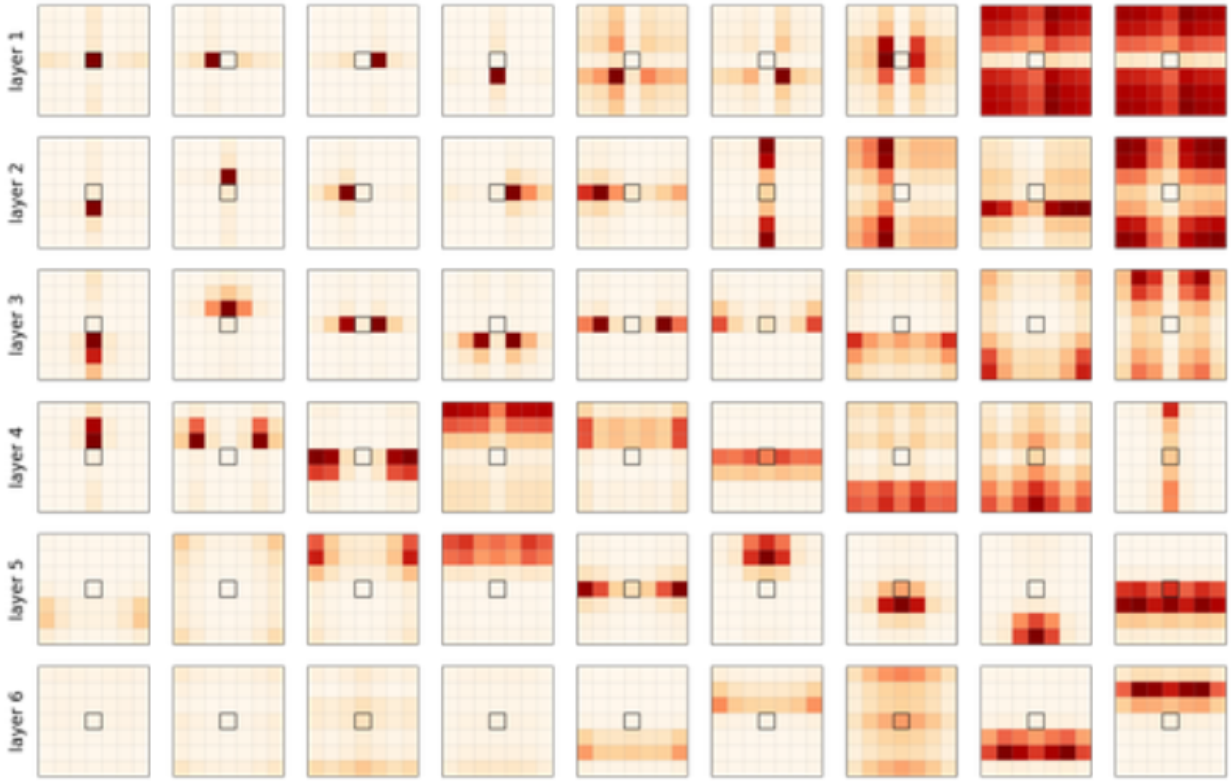


Figure 5: Attention probabilities of each head (*column*) at each layer (*row*) using learned relative positional encoding without content-based attention. The central black square is the query pixel. We reordered the heads for visualization and zoomed on the 7x7 pixels around the query pixel.

- Each layer seems to mostly learn one or two attention profiles.

Do Self-Attention Layers Actually Learn Convolutions? Maybe.

Learned relative positional encoding with content

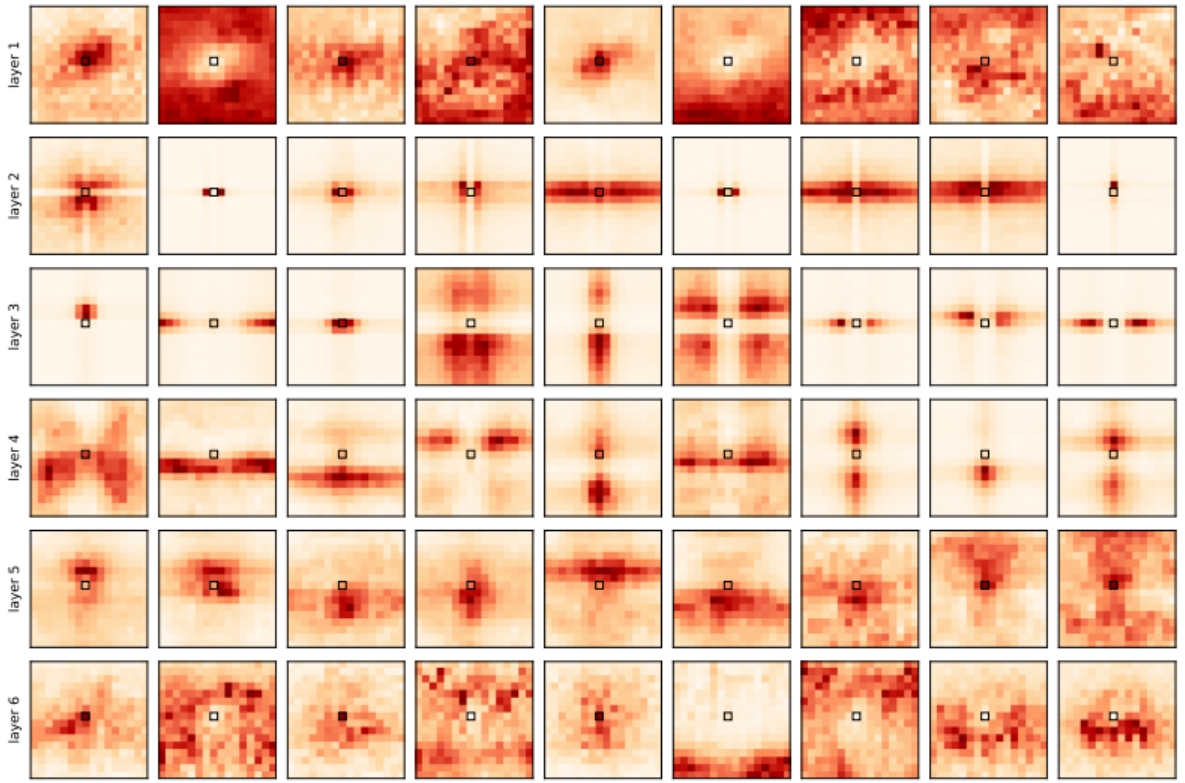


Figure 6: Attention probabilities for a model with 6 layers (*rows*) and 9 heads (*columns*) using learned relative positional encoding and content-content based attention. Attention maps are averaged over 100 test images to display head behavior and remove the dependence on the input content. The black square is the query pixel. More examples are presented in Appendix A.

- Attention probabilities are averaged over images. Still demonstrates some of the behavior shown for the learned embeddings without content.
- Attention profiles do seem to shift with the query pixel $\boldsymbol{q}$ (see Appendix).

Thanks for listening!